**Round Robin Rule of thumb**

|  |  |  |
| --- | --- | --- |
| **Time slice** | **Number of Timeouts** | **CPU Utilisation in %** |
| 1 | 44023 | 19.99 |
| 300 | 135 | 98.47 |
| 2000 | 12 | 99.57 |
| 2300 | 9 | 99.60 |
| 2500 | 8 | 99.60 |
| 2750 | 5 | 99.63 |
| 3000 | 3 | 99.64 |
| 4000 | 1 | 99.67 |
|  |  |  |

To carry out the experiment the program generator was used to come up with the test programs that are shown in the test directory of this folder. For the programs, the minimum CPU burst was 300 while the maximum was 5000. The varying of the time slice was chosen from 2000 going up to 4000.

Table 1 showing the results obtained from varying the time slice

As seen above increasing the number of time outs will only increase CPU utilisation due the fact that the number of interrupts occurring due to time outs have been reduced. As seen above a small time out value of 1 which is lower than all the time bursts of the operating system. This translated to a 19.99% CPU utilisation caused by the 44023 more interrupts caused by the time outs. However, as we increase the time out such that, the time slice for each process is greater than the CPU bursts of 80% of the process we see that the CPU utilization is increased as the number of timeouts is reduced. This starts at time slice between 2750 and 3000. At this time the number of time outs decreases, and the CPU utilization is increased to 99.64%. The optimum value for the time slice can then for an operating system running the type of programs as the generated ones, be between 2750 to 3000. This is so that we can allow to timeout CPU greedy processes at the same time reducing the number of timeouts, so we can then not interfere execution of the process every now and then. This way we are going to have an increased CPU utilization. This information is demonstrated in the graph below.

As seeen from the linear fit the number of timeouts could be seen to decrease with increase in time slice. Thus the optimum CPU utilization which allowing minimum time outs is then for greedy process can be seen to be between 2750 and 3000.

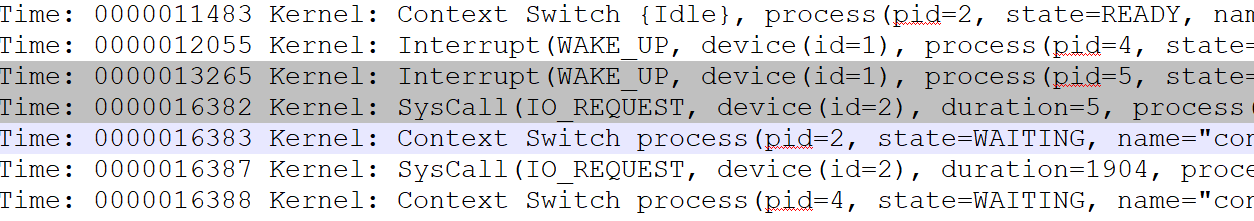
**Convoy effect**

Convey effect is a result of using the FCFS scheduling scheme where the first job to be received is going to be the first job to be processed. The whole operating system will be slowed down due to this convey effect as the scheduler has to wait for a greedy process to finish before calling the next process to come up.

To investigate the effect, programs were generated with varying CPU bursts in the ranges of 10 to 3118.

The results of this are that the instructions which have CPU burst of 10 will quickly process and leave the CPU so that the other processes can then come up and run also. However, there were cases were the CPU burst where 3118 (*which was the maximum registered CPU burst).* For this instruction, the operating system was left waiting for the process to finish before it could have performed a scheduling task. This means that all other process waiting to get CPU time had to wait for 3118-time units before a new process could be put into the queue. During this time the operating could also not do anything except when there was a wakeup time out which also did not disturb the process. This then leads to situation where resources are being misused.

For instance, there may be no processes that are using the IO devices. And the use for these are not optimized. Also, from the trace we can see that the CPU utilization is as low as 48.22%.



Above is a snip of the trace file to show the instruction that results in the convoy effect.