Tutorial 1: RT Services on Linux

Miguel Cabral 93091 Diogo Vicente 93262

Código Desenvolvido

Primeiramente foi alterado o valor da variável `subinterval` na função `Heavy_Load,` para 48000 para obter valores de tempo de execução perto de 20ms. Foram feitos 3 programas um para cada alínea da secção 2.2 e alterado o makefile de modo a que seja possível correr cada um separadamente.

Serviços não RT vs Serviços RT (A1)

Para estes testes foi lançada a *task* e posteriormente foram executados outros processos com operações de I/O intensivas. Nos testes de serviços não RT o Inter-Arrival-Time máximo subiu de **99 ms** para **108 ms** e o mínimo desceu de **99 ms** para **94 ms**. Nos testes de serviços RT a *task* recebeu uma prioridade fixa de 42, neste caso o Inter-Arrival-Time máximo e mínimo mantiveram-se em **100 ms.** Isto justifica-se pelo facto de nos testes em serviços não RT ser utilizado o escalonador *default* do linux e nos testes RT ser usado um escalonador RT do tipo FIFO que prioriza as threads com prioridades fixas, então comos os outros processos usam o escalonador *standard* de Linux a thread RT lançada não sofre alterações no Inter-Arrival-Time.

Serviços RT usando todos os CPU's vs usando CPU0 (A2 e A3)

Seguidamente foram lançadas 8 threads com diferentes prioridades. Analisando os valores obtidos foi verificado que os valores de Inter-Arrival-Time máximo e mínimo não sofreram grandes alterções, estando o IAT de todas as *tasks* no intervalo de **95 ms** (min) e **105ms** (max). Este resultado justifica-se pois as *threads* executadas foram repartidas pelos vários CPU 's presentes, não havendo assim necessidade de escalonamento entre elas.

De seguida realizamos o mesmo teste mas, desta vez, foram lançadas de forma a serem executadas exclusivamente pelo CPU0. Os resultados obtidos encontram-se na tabela abaixo:

Prio/IAT	10	20	30	40	50	60	70	80
min(ms)	34,42	0,15	15,53	35,52	18,72	86,01	99,96	99,97
max(ms)	999,65	999,88	389,72	181,12	180,10	114,48	100,03	100,01

Com os resultados apresentados acima podemos facilmente verificar que os valores máximo e mínimo de IAT obtidos em *threads* com maior prioridade sofrem muito menos variação e quando maior é a prioridade mais baixo é o IAT máximo.