Tutorial 1: RT Services on Linux

Bernardo Falé[1 [93331]] and Diogo Maduro[2 [80233]]

University of Aveiro, Aveiro, Portugal

1 Introduction

Using the concepts given in the classes the authors were able to go through the tutorial with ease; This challenge aimed at creating periodic processes and observe its temporal behavior, and to acquaint the students with the real-time services of the Linux OS.

2 Analysis

2.1 Real time Scheduler vs. Normal Scheduler

We can observe in the table below the data obtained by running our periodicTask program,

N	Priority	Periodicity (ms)	Difference (us)
$Old_Version$	20	100	±15000
$New_Version$	20	100	±100

with a set of applications, in this case, 4 firefox youtube tabs and 10 cpu stress hogs, that the inter-arrival difference time will grow, owing to the fact that the linux scheduler is not optimized to perform real-time tasks.

We can also see that the difference is almost 0 in the newer version, after setting the real-time optimizations, . This new version was running concurrently with the old version, with the same priority and periodicity, which is 20 and 100ms respectively.

2.2 The impact of priorities in a Real Time Scheduler

The Linux real time scheduler priority is static defined between the values 1 to 99, the threads that have the highest value of priority are the first to be running in the CPU, and we tested this by running the script 7 times in parallel with affinity set to the CPU0, each one with different priorities, this way we overloaded the CPU0 with real time task.

N	Priority	Periodicity (ms)	Difference (us)
1	10	98	±448560
2	30	99	±363720
3	40	100	±372353
4	50	101	±490508
5	60	102	±210484
6	70	103	±156276
7	99	104	±95708

It is perceived in the table that the differences are not fully compatible with the theory behind the fixed priorities. This can be caused by other tasks that are running in real-time, like audio applications. The phasing of the periodicity, although it is a small interval, can also impact the running tasks, and so, the results are not always trivial. However, we can conclude that there's a huge decrease in difference in the last 3 tasks; This means that the higher the priority, the more are the chances that our task will be ran on time, thus more stable.