João Castanheira (97512) • *Nuno Fahla* (97631) • TP3

Real-time Operating Systems

Xenomai introduction (2022/2023)

We were given code that through a real-time API, created and executed a task that caused the processor's load to increase. Using other unrelated I/O operations to increase the CPU usage, we noticed that with the given code the task executed on time, that being said, it had a 1 second period to execute a ~30ms task, so we started to make the appropriate changes to get some meaningful results.

As indicated, we started by adding information about the maximum and minimum inter-arrival time. After that, we created 2 more tasks using the Xenomai API's methods and changed their affinity to the same CPU, in this case CPU 0. It is important to note that the tasks' periodicities are not multiple of each other, as to avoid harmonic task scheduling.

When running the 3 tasks with different priorities, we could watch live the impact of real time priorities by running multiple tasks assigned to the same CPU. Our findings are reported below (*figure 1*) and as we can see the higher the task priority the lower is the difference between the maximum and minimum time of successive jobs of the same task, in other words, tasks that need more <u>strict deadlines</u> should have <u>higher priorities</u> to cause <u>less fluctuation</u> in min/max <u>execution times</u>.

Task	Priotity	Period(ms)	Min(µs)	Max(µs)	Diff(ms)
A	12	113	50758.922	176813.578	126.054656
B	25	97	66392.164	124113.086	57.720922
C	60	121	120994.242	121002.078	0.007836

Figure 1

In a more extreme environment, where CPU availability was scarce, low priority tasks could even be <u>overrun</u>, as evidenced below (*figure 2*), and as expected, the difference between min/max inter-arrival times increased.

Task	Priotity	Period(ms)	Min(µs)	Max(µs)	Diff(µs)
A	12	113	-	-	overrun
B	25	97	50997.746	121039.281	70041.535
C	60	121	120990.031	121016.391	26.36

Figure 3