REAL TIME EMBEDDED SYSTEM FINAL ASSIGNEMENT

IPSA
ÉCOLE D'INGÉNIEURS
DE L'AIR, DE L'ESPACE ET DE LA MOBILITÉ DURABLE

MARIAM SHAABAN

ANDREIA DE ALMEIDA

Abstract

This study aims to analyze the real-time performance of a set of five tasks within a time-sensitive system. Our objective is to measure the worst-case execution time (WCET) of each task, select an appropriate period for each task, and assess their schedulability. By employing Fixed Priority (FP) scheduling in FreeRTOS, we seek to understand the impact of task execution time on system performance and determine whether the task set meets given deadlines.

Introduction

Real-time operating systems (RTOS) are crucial in ensuring tasks are completed within strict timing constraints. RTOS guarantees critical operations are executed timely, meeting deadlines.

Our study revolves around analysing the real-time performance of tasks within a time-sensitive system. Our primary goal is to measure the WCET of each task and determine suitable periods for scheduling. To achieve this, we utilize the Fixed Priority (FP) scheduling algorithm within the FreeRTOS environment. FP scheduling assigns priorities to tasks based on their characteristics, allowing higher priority tasks to preempt lower priority ones when necessary.

We will elaborate on the methods employed and present our findings in the next parts.

Methodology

To achieve the objectives outlined in the introduction, our methodology consists of the following.

Firstly, we begin by designing the RTOS with the set of tasks .

In our system, we have four periodic tasks and one aperiodic task. The first periodic task periodically prints "It's working" to indicate normal system operation. The second periodic task is responsible for converting a fixed Fahrenheit temperature to Celsius degrees. Following that, the third periodic task performs multiplication on two integers. The fourth periodic task executes

a binary search algorithm on a list of 50 elements. Additionally, we have an aperiodic task that completes a specific operation within 100 milliseconds.

Then, we write the code of each periodic task in a separate c file. We execute them and We use the executed files in a python code to find the Worst execution time. To do that, we run each code 1000 times and then we choose the maximum time of execution.

Finally, We can choose a period and we make sure that the set of files are schedulable by running posix_demo file with each time we change the period to test.

Furthermore, we conduct extensive experiments to explore the impact of task execution time on system performance. By systematically varying task parameters, we evaluate the schedulability of the designed system.

In summary, our methodology encompasses task design, execution time measurement and schedulability analysis**.**

Results

When running the Python code, we found that the Worst-Case Execution Time (WCET) for each periodic task is respectively 41 ms, 34 ms, 43 ms, and 48 ms. To find the WCET, we run the code for each task 1000 times and then we pick the maximum value.

To choose a period for each task, we can use the WCET as the execution time. Therefore, we should set a period for each task greater than the sum of all the WCETs calculated above. This ensures that each task can execute before its deadline.

The sum of the Worst-Case Execution Times (WCETs) for the tasks is 166 ms. Therefore, to ensure that each task can execute within its deadline, we should set the period of each task to be equal to or greater than 166 ms.

Therefore, we don’t have freedom over the choice of the period; we must choose at least 166 ms.

Since we see that the code is running, we can conclude that the tasks are indeed schedulable and execute before their deadlines.

If we do lower the period, the tasks have higher risk of completely missing their period, therefore not executing properly.

Conclusion

Through FreeRTOS, we were able to schedule the tasks in a timely fashion and then we run each task 1000 times, we could increase this to get a better estimation, but we run the risk of a having a WCET that is too pessimistic.

We use the WCET to fix the period for the different tasks and we then run the code to ensure it is schedulable and respects the deadline.