**Final Project**

Prepared by: Mohamed El Azhary   
Date: 15-10-2022

**Introduction**

In this project, we are implementing and analyzing a dynamic priority based preemptive scheduling algorithm known as the Earliest Deadline First “EDF” algorithm. Using the previously published thesis “Implementation and test of EDF and LLREF schedulers in FreeRTOS”, we are implementing the algorithm as documented in the thesis and providing a report explaining the work done and the manual and automated analysis done on the proposed taskset.

**Dynamic priority scheduling in brief**

It’s the type of scheduling algorithms in which the priorities are calculated during the execution of the system. The goal of dynamic priority scheduling is to adapt to dynamically changing progress and to form an optimal configuration in a self-sustained manner. It can be very hard to produce well-defined policies to achieve the goal depending on the difficulty of a given problem. Earliest deadline first scheduling and least slack time scheduling are examples of Dynamic priority scheduling algorithms.

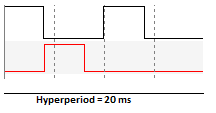
**EDF in brief**

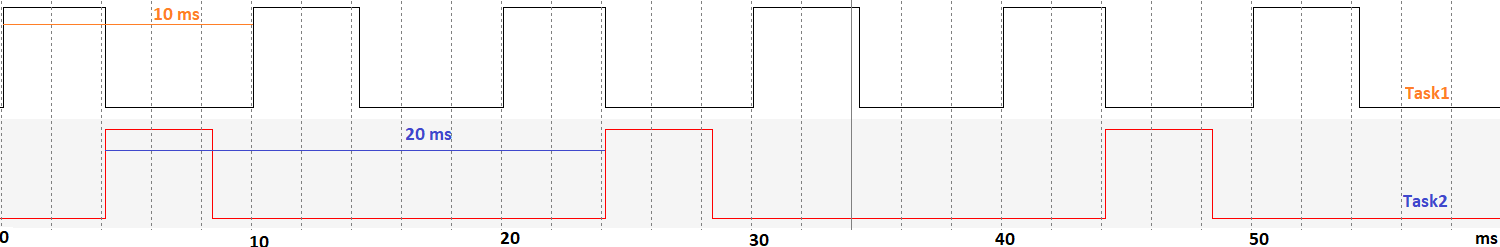
EDF adopts a dynamic priority-based preemptive scheduling policy, meaning that the priority of a task can change during its execution, and the processing of any task is interrupted by a request for any higher priority task. The algorithm assigns priorities to tasks in a simple way: the priority of a task is inversely proportional to its absolute deadline; In other words, the highest priority is the one with the earliest deadline.

**The Proposed taskset**

* The proposed taskset is as follows,
  + TaskA (Periodicity=10, Deadline=10, WCET=4.3056) ms.
  + TaskB (Periodicity=10, Deadline=20, WCET=4.32195) ms.
* In runtime priority is dynamic and is based on the earliest deadline first.
* Initially TaskA has higher priority than TaskB as its deadline is nearer in occurrence than TaskB’s deadline.
* In EDF, the priority is inversely proportional to the deadline of the task, this means that initially TaskA with Deadline = 10 has higher priority than that of TaskB having a Deadline = 20, but this is only initially as in EDF the priorities are dynamic.
* Hyperperiod = LCM [10,20] = 20 ms.
* Assumptions made during implementation,
  + Tasks are periodic.
  + Deadline of a task is equal to its period.

**Implementation Graph**

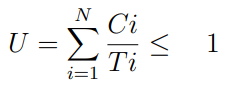




**Implementation Results - Notes**

* Earliest Deadline First is executed successfully
* Tasks are meeting their deadlines successfully.
* Taskset is schedulable using EDF.

**EDF Manual Schedulability Analysis**



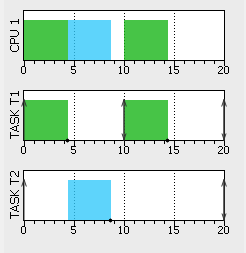
* U =
* Taskset is schedulable using the EDF scheduling algorithm.

**System Manual CPU Load Calculation**

=

* CPU load is within accepted limit, system is healthy.

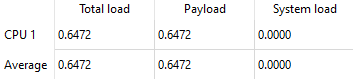
**SIMSO Simulation**



**SIMSO Simulation – Notes**

* Taskset is schedulable using EDF as no task exceeds its deadline.
* SIMSO simulation aligns with the practical implementation.

**SIMSO CPU Load Calculation**

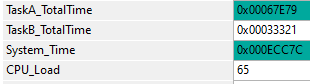


**SIMSO CPU Load Calculation – Notes**

* CPU load calculated with SIMSO aligns with manual calculations.

**Run Time Calculations**

**Run Time - Manual CPU Load Calculation**

* Using trace hooks in switch out and switch in instances along with timer1.
* CPU load was calculated approximately as 65%, aligning with previous calculated measurements.  
    
  

**Run Time – RUN Stats CPU Load Calculation**

* Using RUN Stats FreeRTOS feature, the CPU Load was calculated and sent over UART  
   
* In the above screenshot, partial CPU load of tasks is displayed and the total CPU load will be their summation which is equal to 64% and this is more accurate than manual CPU load calculations.