****

**Universidad Autónoma de Guadalajara**

Biomedical Engineering

Peripherals and Interfaces

Jesús Arnoldo Zerecero Nuñez, 2885993

Edwin Francisco Zamora Oros, 2874382

*“Lab 2: Scheduler (part 2)”*

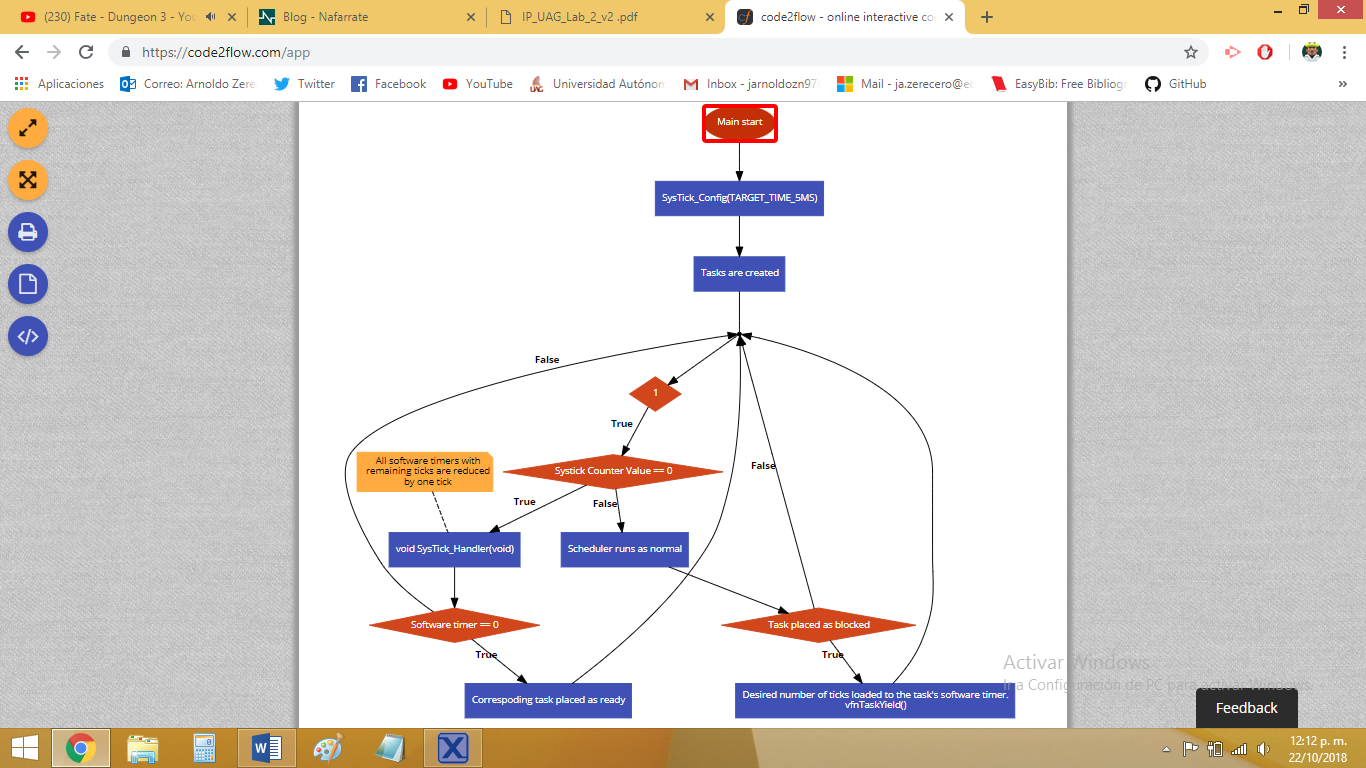
22/10/18

**Introduction:**

In this second lab practice, additional functions were added to the scheduler implemented in lab #1: a function to place a task as blocked for a certain amount of time (ticks) and the implementation of a system tick timer, which enters a handler every time base (or every “tick”).

The scheduler must keep track of the time left for every blocked task to come back as ready, so, a software timer must be implemented for each possible task that could be created. Every time the system tick generates an interrupt, each timer with remaining counts must reduce its value by one, and whenever that time is up, the corresponding task must be placed back as ready.

**Flowchart:**



**Development (issues):**

* **Realizing the PendSV had a higher priority than the System Tick interrupt.**
  + After implementing the system tick, we found out the code stopped entering the system tick handler after the scheduler had started.
  + We struggled finding the cause of this, as everything seemed to be initialized as it should, and everything else was working properly.
  + After researching, we found out the PendSV handler was overriding the system tick handler because if had a higher priority (-2, while system tick had -1).
  + The priorities were changed, and everything worked as it should.

**Questionnaire:**

**What’s the overhead added by the scheduler?**

The time consumed by the scheduler while it’s making scheduling decisions (which tasks shall run next, keeping track of blocked tasks’ software timer, etc.).

**What’s the importance of having an IDLE task in an RTOS?**

It’s of vital importance, as the scheduler must always have a task running, and if there is no task to run at a given time, bad things may happen.

**Conclusion:**

**Arnoldo:**

This second delivery of the Scheduler lab was much easier than lab #1, as the requirements were much lighter. Although, what we added for this delivery is also a core part of any scheduler, as now our scheduler can handle both ready and blocked tasks, running a system tick timer in the background.

**Edwin:**

Practice number 2 is the continuation of the previous one, so we only had to add this new part where we learned what the Systick is. This helps us to be able to give a specific time to each task so it can leaves the blocked queue and passes to the ready queue. This part of the practice was a bit easier than the last one but it helped us to better understand how a kernel works and the components it carries.