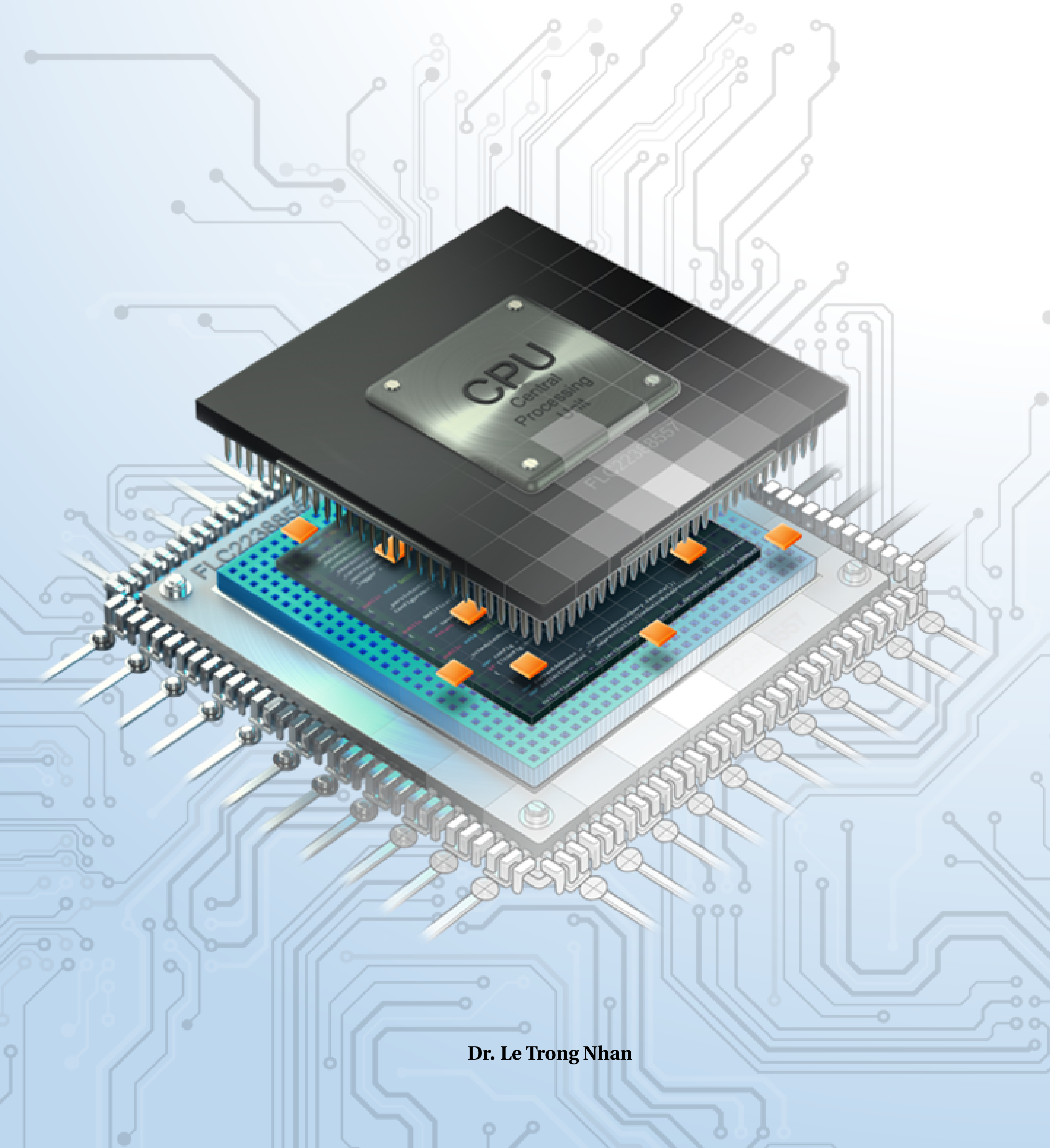
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
COMPUTER ENGINEERING

Microcontroller

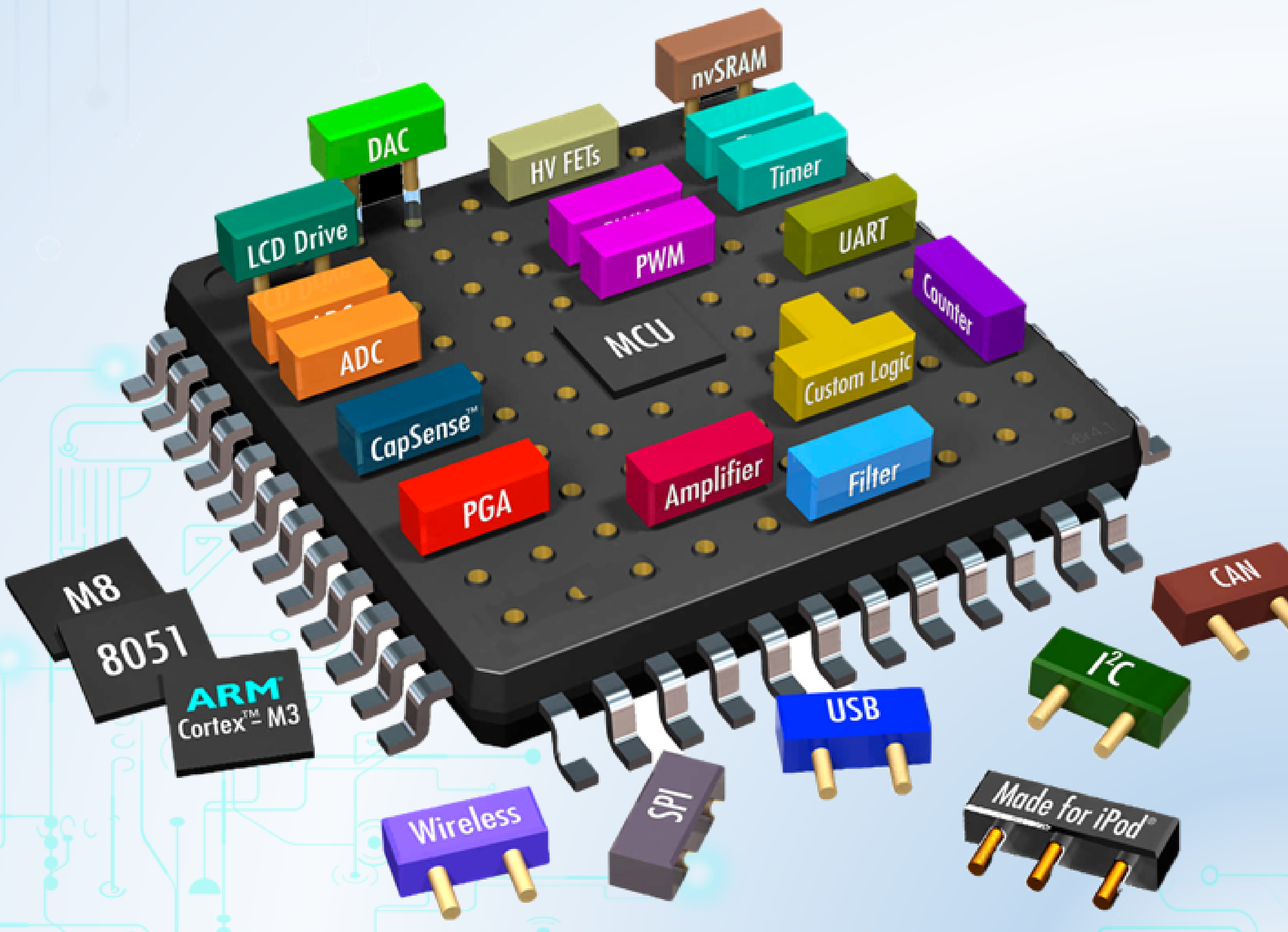


Dr. Le Trong Nhan

Contents

CHAPTER 1

A cooperative scheduler



1 Objectives

The aim of this lab is to design and implement a cooperate scheduler to accurately provide timeouts and trigger activities. You should add a file for the scheduler implementation and modify the main system call loop to handle timer interrupts.

2 Problem

- Your system should have at least four functions:

- **void SCH_Update(void)**:This function will be updated the remaining time of each tasks that are added to a queue. It will be called in the interrupt timer, for example 10 ms.
- **void SCH_Dispatch_Tasks(void)**: This function will get the task in the queue to run.
- **uint32_t SCH_Add_Task(void (* pFunction) (), uint32_t DELAY, uint32_t PERIOD)**: This function is used to add a task to the queue. It should return an ID that is corresponding with the added task.
- **uint8_t SCH_Delete_Task(uint32_t taskID)**: This function is used to delete the task based on its ID.

You should add more functions if you think it will help you to solve this problem. Your main program must have 5 tasks running periodically in 0.5 second, 1 second, 1.5 seconds, 2 seconds, 2.5 seconds.

3 Demonstration

You should be able to show some test code that uses all the functions specified in the driver interface.

Specifically set up and demonstrate:

- A regular 10ms timer tick.
- Register a timeout to fire a callback every 10ms.
- Then, print the value returned by get_time every time this callback is received.
- Note: Your timestamps must be at least accurate to the nearest 10ms.
- Register another timeout at a different interval in addition to the 500ms running concurrently (i.e. demo more than one timeout registered at a time).
- Before entering the main loop, set up a few calls to SCH_Add_Task. Make sure the delay used is long enough such that the loop is entered before these wake up. These callbacks should just print out the current timestamp as each delay expires.

Note this is not a complete list. The following designs are considered unsatisfactory:

- Only supporting a single timeout registered at a time.
- Delivering callbacks in the wrong order
- O(n) searches in the SCH_Update function.
- Interrupt frequencies greater than 10Hz, if your timer ticks regularly.

4 Submission

You need to

- Demonstrate your work in the lab class and then
- Submit your source code to the BKeL.