LAB #3—Alarm clock

Kizito NKURIKIYEYEZU

October 9, 2021

1 LAB OBJECTIVES

- Understand the embedded software design process from requirements to implementation
- Understand how to translate business requirements/logic into code?
- Understand the necessary steps to design a system that uses a multitasking operating system.
- Implementation and testing of a medium scale embedded system project

2 LAB PROJECT

In this lab[1], you will develop a 6-digit alarm clock with the following features:

- 6-digit LED display, showing hours : minutes : seconds. The hours can be in either a 12 hour or 24 hour format. In the 12 hour format a single LED indicator specifying AM / PM is included.
- 6 controls, FAST_SET, SLOW_SET, TIME_SET, ALARM_ SET, ALARM_ON, SNOOZE.
- The alarm shall both flash the display, and emit a AM modulated audio tone.

3 Implementation details

- The project will be developed in C. Start with the this lab's code template.
- The details on how to implement the code is discussed in details in *Curtis, K. E. (2006). Embedded Multitasking. Newnes.*

[1] based on Curtis, K. E. (2006). Embedded Multitasking. Newnes.

- Please read this book's chapters 3, 4 and 5.
  - Chapter 3 discusses the system-level design, including definition of the tasks, layout of the communications, determination of the overall system timing, and the high-level definition of the priority structure. It also discussest the system requirements document, and the functions required, their timing, their communications needs, and their priorities.
  - Chapter 4 continues the design process, translating the system-level design from the last chapter into the individual software components that will make up the final system. This includes the design of the state machines, timing controls, and priority handler, as well as defining the variables used for communications.
  - Chapter 5 concludes the design process, translating the component level design from the last chapter into the actual software that will make up the final system. This chapter will cover not only the writing of the software but also individual module tests and integration testing of the complete system. When we are finished, we will have a complete, tested software solution for the design specified in the requirements document.
- All switches should use internal pull-ups
- The alarm's display should use an LCD display
- The alarm shall be generated using a buzzer

4 Lab grading criteria

- **FUNCTIONALITY (80%)**
  - Code does not compile ..... **deduct 60%**
  - The simulator does not run (e.g, missing the .hex file) ..... **deduct 60%**
  - The device does not work as intended ..... **deduct 20%**
  - If the LED's bounce when the switch is pressed ..... **deduct 20%**
  - Any missing/non-implemented specification ..... **deduct 20%**
- **CODE QUALITY(40%)**
  - Code's tasks not organized as discussed in the book ..... deduct 10%
  - Poor coding style ..... deduct 5%
  - Poor code organization and modularization ..... deduct 10%
  - Using magic numbers ..... deduct 5%

  - Messy, unreadable Code ..... deduct 5%
  - Violates major embedded C coding standard ..... deduct 5%
  - Does not use proper naming convention ..... deduct 5%
  - Does not use meaningfull variable names ..... deduct 5%
  - Use God functions ..... deduct 5%
- Any cheating or copying someone else's code ..... **deduct 100%**
- **TOTAL** **100%**

5 Lab submission

The lab is due on **October 17, 2021**. The submission shall go as follows:

- You should submit the report, all your code file (c or cpp files), the proteus simulation file and the .hex file in one .zip file and submit them through the e-learning platform no later than midnight on October 17, 2021.
- Before submission, please make sure that, when the simulation file is properly linked with the .hex file. **I should be able to run your simulator without compiling your code.**
- Please submit your work before the deadline. I will not accept any submission through my email no matter the reasons