Hochschule Esslingen
University of Applied Sciences

Echtzeitsysteme

Lab Exercise Tea-Timer with Switches

Prof. Dr. Jörg Friedrich

Table of Contents

1 Introduction

The Tea Timer functions as a little embedded application that helps a user to have his tea come out just right. As part of this project we will not build the hardware for this, but rather use the existing Dragon12 board in the lab. It provides all features we need for this application.

2 Requirements

The user can enter a time between 0 and 5 minutes with a precision of a second. After the timer is started by the user the timer counts down the time and sounds a beeper when the entire time has elapsed.

To avoid having to enter the preferred time every time the timer is used, the last time entered is being stored and being recalled upon push of a special button.

Input Elements

Switch 1 and 2: set time

Switch 3: recall last time used

Switch 4: start countdown, pause countdown

Output Elements

Main output is on the LCD. Output shall be designed as follows.

Display while being set:

First line: “Preset: 03:00”
Second line: “Actual: --:--“

The preset changes while it is being set. Display during countdown

First line: “Preset: 03:00”
Second line: “Actual: 01:34“

The actual value changes during countdown. After expiration of the time interval the beeper shall sound five times for one second each, with a break of one second in between beeps.

3 Suggested Implementation

The suggested implementation consists of two tasks. One task takes care of switch input and computes the preset. The other task executes the countdown and writes to the display. The scheduler shall be very simple and call the tasks in a round robin fashion.

4 Implementation Procedure

Design

Before any software is implemented a design is required. The design should consist of UML state charts and message sequence diagrams.

Implementation

You are required to follow the procedure described in the lecture notes, with most of the software development taking place on the host computer.

5 Pass Criteria

You may fail at most two of the following criteria; otherwise you will not be given credit for the lab exercise:

1. Have a proper directory layout, as described in the lecture, and have cleanly separated HALs, IDEs, and application software.
2. Have a proper UML design which can easily be related to the code (same names, etc.).
3. Can show all relevant functionality demonstrated on the host PC with test cases stored in files. Demonstrate that contact debouncing works properly in the simulation.
4. Have used source code control (not just one version). Must have the most up to date version checked in.
5. Show proper functionality on the target system. Make sure debouncing works properly.
6. Each team member must have programmed at least 30% of the system himsef. It does not suffice to watch how the other has programmed. Must prove to the instructor which software parts have been programmed.