Milestone 6

New ESOS32 Service

"User interface is like a joke; if you have to explain it, then you've done it wrong."
– *Anonymous*

PRACTICAL embedded systems that interact directly with users need an ergonomic and efficient user interface. Users need to telegraph their intentions to the systems and the embedded systems needs to provide feedback to the user. In this milestone, you will develop the hardware-specific portions for a new "user-interface service" for ESOS32 to provide to ESOS32 application developers with a clean and direct method by which to interace with users. The systems' interaction with users is done via provided hardware devices[1], *e.g.* the pushbuttons, LEDs, the potentiometer, and the liquid crystal display.

6.1 Repo preparation

1. Be sure that you are working from the latest ESOS32 tree.
2. Using the tree structure and the repository location provided by the TA or Dr. Bruce, start populating your team repo.
3. Configure your client to access and manage your team git repo.
4. Maintain all code and documentation for the remainder of your lab in this repo. The TA will provide instructions for each milestone as to the location of the files.

6.2 ESOS32 UI Service

User-interface components are often platform-specific, and the ECE 4140/5140 hardware board is no different. You will develop an UI service in ESOS32 to ease the developer's burden in configuring and reading the UI devices on your target board. The latest version of ESOS32 code

[1]The LCD and the potentiometer are quite different devices and more involved than the simple buttons and LEDs. We will address these two devices individually in a subsequent design lab.

contains a "simple user interface" (SUI) service. The hardware independent code is contained in `esos_sui.c` and `esos_sui.h`. These files will contain the data structures, macros, and code to implement the user interface without regard to the underlying processor and hardware. Your job is to use the `generic_hwxxx` files as a template and create the ARM-Cortex hardware support files for the ESOS32 SUI service. These files will be named `esos_stm32l4_edub_sui.c` and `esos_stm32l4_edub_sui.h`. You will use these files throughout the rest of the semester. There should only be one copy of these files in your repo for ease of maintainability and can be placed in your ESOS32 tree. Any future pulls from the public ESOS32 repo should leave these files untouched.

Be sure that you fully and properly document how to use your new service for any developers that might come along later. This will also help you and your team in subsequent design milestones as all of your future UI development will be done with your implementation of the SUI on the Edubase board.

6.3 Test Applications

In testing your SUI code and verifying its functionality, you will likely need to create series of very simple demonstration applications. Be sure these are well-documented and cover the full range of the SUI.

6.4 ESOS application

In a file named `m6_app.c`, implement the same functionality as `m5_app`. Your application in this design should utilized the ARM-Cortex SUI service with the hardware-specific code your team developed.

6.5 Check Off

Demonstrate the following things to the TA:

1. Commit your project to your repository to your "trunk" folder in accordance with the procedure provided by the TA. Your github repository should all files required to produce your "build", including the appropriate build file(s).

2. A "tag" release in your team's git repository called `m6` that can be recalled at any time to build this milestone's deliverables.

3. Demonstrate the key features of the SUI service and the watchdog service via your test applications.

4. Using the target board specified by the TA, demonstrate the proper operation of the your user application.

6.6 Submission

Each team should submit the following items to their repo. Send a notification email to the TA as to the location of your submission.

- the hardware support files for the simple user interface targeting the PTB: `esos_stm32l4_edub_sui.c` and `esos_stm32l4_edub_sui.h`
- API and use documentation for your ESOS SUI services for the STM32L4 and Edubase.
- well written, well commented ESOS32 application code for `m6_app.c`
- any unit test or test bench files used to verify your code created for this milestone
- updated design review forms and software metrics