LAB 1

Introduction to the TI Launchpad and Boosterpack

Haritha Gnanasegar

Professor Leyla Nazhand-Ali

ECE 2534-82953

Project Description

This project develops and implements a program on the TI Launchpad and boosterpack. The commands for the buttons on the launchpad and the booster pack are followed by the user and are displayed on the LCD screen. The inputs are read from the four buttons, and the output is shown by the Red-Green-Blue (RGB) LED's on the launchpad and the boosterpack.

The program should manipulate the two buttons on the Launchpad (L1 and L2), and the two buttons on the boosterpack (S1 and S2) should light the RGB Led on the booster pack. When the program reloads after one cycle, the initials and the reset state are the same.

The conditions are: When no buttons are pressed

- The Left LED on the launchpad should be on
- The Right LED on the Launchpad should have its Red LED on with the Blue and Green off
- The RGB LED on the booster pack should stay off

The program should light the following colors in the given order on the RGB LED: Red, Blue, Green, Purple, Orange, White.

Hardware Abstraction Layer (HAL)

For the application developed to achieve the output described in the problem description, the HAL model of code was extensively used. This model was obtained to write the application as it gave a better readability of code. In the application, the operations to be achieved by the bit manipulation of the Registers are separated into different small methods, and these methods are called when required for operation. Hence, this makes the main() and the key methods clear and concise to read.

For example: Below is a snippet of the code where the RED LED on the RGB LED is turned on if L1 is pressed and L2 is not pressed.

```
if((SwitchStatus_Launchpad_Button1() == PRESSED) &&
(SwitchStatus_Launchpad_Button2()!=PRESSED))

{
        TurnOn_Booster_LEDR();
        TurnOff_Launchpad_LED1();
        pick->LEDColor = Red;
}
```

If the HAL Method was not obtained for writing the application and if the bit manipulation is done in the if condition, it would look like the below code.

```
if (((P1IN & LEFT_BUTTON) == PRESSED) && ((P1IN & RIGHT_BUTTON)!= PRESSED))
```

This decreases the code readability because a third person reading the code would not be able to know what is happening with the bit manipulation. It is not necessary to know the hardware details, but it can be referred if needed. But by using the HAL method and naming the methods after their functionality, it becomes easier to know what each line of code is looking to achieve.

Additionally, by using HAL methods, sections of code become more portable. If a similar bit manipulation is to be achieved in a different program, all one needs to do is copy the methods and paste it along with the function declaration.

There are not a lot of disadvantages of using HAL, but one disadvantage can be that it is more time consuming than writing just the hardware details. HAL involves declaring multiple methods, defining them, and having the right calling statement for each of them. Hence, it can be a disadvantage for someone who has an approaching deadline to meet.

Driverlib functions versus Manipulating registers

Advantages of Driverlib functions:

- They are easy to use once their functionality is known because they take away the tedious work of register manipulation. Hence, the register manipulation that is not written by the coder makes it easier for debugging.
- Since a major part of the details are already written, it reduces the length of an application, making it more concise to read.
- Also using Driverlib functions masks away the harder part of application development. So, the programmer is only confined to writing for the problem description and does not have to worry how the information is manipulated in the hardware.

Disadvantages of Driverlib functions:

- By using only driverlib function, one is not aware of the details that build the program.