ESET 369 LAB 3 REPORT

Student name	Kyle Rex and William Parks
Group number	36
Lab Section	501
Lab session date	February 10, 2025
Lab instructor	Xin Zhuang

INTRODUCTION

The objective of this lab is to develop a functional embedded system using the C/C++ programming language, Code Composer Studio, and the MSP430FR5944 microcontroller. The system is designed to achieve two primary tasks. The first task is to implement a six-color sequence on an RGB LED controlled by a button press on the Launchpad, ensuring that the LED cycles through red, green, blue, cyan, magenta, and yellow while the button is held. The second task involves configuring and programming a keypad matrix such that pressing a button on the keypad correctly illuminates the corresponding LED. This lab provides hands-on experience with embedded system design principles, including GPIO configuration, microcontroller interfacing, and software development for real-time control applications.

SYSTEM A

The purpose of System A is to correctly configure the launchpad to create a 6 color (Red, green, blue, cyan, magenta, and yellow) blinking sequence on a single RGB LED on the BH board that is being controlled by a button on the launchpad itself using a C/C++ code sequence. The button that is used to control the RGB LED is the “S2” button on the MSP430FR5944 Launchpad board. While this button is pressed, the RGB LED on the BH EDU board should keep changing colors with the sequence of red, green, and blue, cyan, magenta, and yellow. When the button is released, the LED should be turned off. The duration of the color change must be between 0.5 to 1 second.

The first step in addressing this problem is making the physical board connections between the BH board and the MSP430FR5994 microcontroller. This can be done by connecting three specific pins together using male-to-female jumper wires, checking that all the DIP switches on the BH board are in the correct configuration, and ensuring the microcontroller itself is properly seated on the BH board. Pin P6.0 controls the color red of the RGB LED, pin P6.1 controls the color green of the RGB LED, and pin P6.2 controls the color blue of the RGB LED. These are all active low meaning the LED color is on when the input is 0. Pin P5.5 is for the S2 button on the launchpad and does not require any connection like the previous three pins because it’s on the launchpad itself, not the BH board. The button, like the RGB LED individual colors, is active low and can be enabled the same as any button. To get the other 3 colors required in the sequence it involved having two of the three original colors (Red, green, and blue) on together at the same time. To get the color cyan the colors green and blue were on together. To get the color magenta the colors red and blue were on together. To get the color yellow the colors green and red were on together. This combination of colors is represented in Table 1.

Table 1: Color Combinations (0 = LED on, 1 = LED off)

	P6.0 (Red)	P6.1 (Green)	P6.2 (Blue)
Cyan	1	0	0
Magenta	0	1	0
Yellow	0	0	1

Using the code sequence for system B of Lab 1 as a reference the code sequence used for this system was created. The program is designed to control an RGB LED cycling through different colors when a button is pressed. The LED colors are active-low, meaning they turn on when their corresponding output is set to 0 and off when set to 1. The program begins by disabling the watchdog timer to prevent unintended resets and unlocking the GPIO pins. The RGB LED pins (P6.0 for Red, P6.1 for Green, and P6.2 for Blue) are configured as outputs and initialized to an off state by setting them high. The button (S2) is connected to P5.5, which is configured as an input with an internal pull-up resistor to ensure a stable high signal when the button is not pressed. Inside the infinite loop, the program continuously checks the state of the button. When the button is pressed (P5.5 reads low), the RGB LED cycles through different colors in sequence, with a 500ms delay between each transition. The sequence includes Red, Green, Blue, Cyan (Green + Blue), Magenta (Red + Blue), and Yellow (Red + Green). When the button is released, all LEDs are turned off by setting their outputs high. This implementation demonstrates button-controlled LED color cycling, effectively utilizing GPIO configuration and input handling with an MSP430 microcontroller.

SYSTEM B

The purpose of System B is to correctly configure the launchpad to control 6 corresponding buttons and LEDs on the BH board using a C/C++ code sequence. When the buttons of the keypad matrix shown in Figure 1 are pressed, the proper LED should be turned on. When the button is released, the proper LED should be turned off.

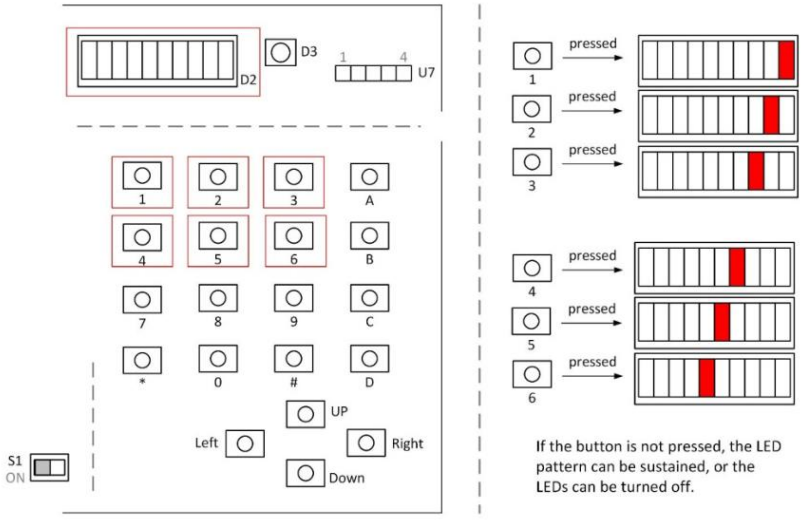

Figure 1: LED & Button Configuration

The first step in addressing this problem is making the physical board connections between the BH board and the MSP430FR5994 microcontroller. This can be done by connecting twelve specific pins together using male-to-female jumper wires, checking that all the DIP switches on the BH board are in the correct configuration, and ensuring the microcontroller itself is properly seated on the BH board. The P3 pins control the button columns and the P7 pins control the button rows while the P5 and the P8 pins control the LEDs. Pin P5.3 controls LED 6 (Far left red LED as seen in the figure), pin P5.1 controls LED 5, pin P5.0 controls LED 4, pin P8.2 controls LED 3, pin P8.1 controls LED 2 and pin P8.0 controls LED 1 (Far right red LED as seen in the figure). Pin P3.0 is a lateral connection used for buttons 1, 4, 7, * and its port direction must be configured as an input. Pin P3.1 is a lateral connection used for buttons 2, 5, 8, 0 and its port direction must be configured as an input. Pin P3.2 is a lateral connection used for buttons 3, 6, 9, # and its port direction must be configured as an input. Pin P3.3 is a lateral connection used for buttons A, B, C, D and its port direction must be configured as an input. Pin P7.0 is a horizontal connection used for buttons 1, 2, 3, A and its port direction must be configured as an output. Pin P7.1 is a horizontal connection used for buttons 4, 5, 6, B and its port direction must be configured as an output. With these configurations the buttons and LEDs relevant for this system (Buttons and LEDs 1, 2, 3, 4, 5, and 6) are all set up. In this setup the button number directly correlates with its adjacent LED number therefore when button 1 is pressed LED 1 is on, when button 2 is pressed LED 2 is on, and so on. This setup follows the configuration that is required for this system that could previously be seen in Figure 1.

Using a code sequence labeled 5.1, found in: *Learning Embedded Systems with MSP430 FRAM microcontrollers* by B. Hur, as a reference the code sequence used for this system was created. One button and one LED were coded and tested at a time to ensure that there were no issues with the code or the hardware. The program, in its most basic sense, interfaces a 3x2 keypad to control six LEDs. The LEDs operate on an active-low logic, meaning they turn on when their corresponding output is set to 0 and off when set to 1. The program first disables the watchdog timer to prevent unintended resets and unlocks the GPIO pins. The LED pins (P5.3, P5.1, P5.0, P8.2, P8.1, P8.0) are configured as outputs and initialized to an off state. The keypad rows (P7.0, P7.1) are also set as outputs, while the columns (P3.0, P3.1, P3.2) are configured as inputs with pull-up resistors to detect key presses. The program continuously scans the keypad to determine which button is pressed. To do this, it first sets all rows high, then lowers the first row (P7.0) to check for key presses. If a column input reads 0, it indicates that the corresponding button is pressed, and the associated LED is turned on. This process is repeated for the second row (P7.1), activating the corresponding LEDs for buttons in that row. A short delay is introduced after each row scan to allow for debouncing and prevent false readings due to electrical noise. By iterating through this scanning process in an infinite loop, the system provides real-time feedback where pressing a button illuminates the corresponding LED. This implementation effectively demonstrates keypad scanning and LED control using an MSP430 microcontroller.

CONCLUSION

The lab successfully demonstrated the ability to interface an MSP430FR5944 microcontroller with external peripherals, including an RGB LED and a keypad matrix. Through System A, the proper

configuration and control of the RGB LED were achieved, ensuring a sequential color change upon button press while maintaining the required timing constraints. System B effectively integrated a keypad matrix with LED outputs, allowing individual LEDs to respond accurately to corresponding button presses. The implementation reinforced critical embedded system concepts such as GPIO configuration, pull-up resistors, and microcontroller-based input/output control. Overall, the lab met its objectives, providing valuable insights into hardware-software integration and microcontroller programming for real-time applications.

REFERENCES

B. Hur, *Learning Embedded Systems with MSP430 FRAM microcontrollers*, 2nd ed., 2023.

K. Rex, *ESET 369 Lab Report 2*, 2025.

K. Rex, *ESET 369 Lab Report 1*, 2025.