Memo to: Mr. Cory Mettler
From: Julian Rechsteiner
Date: January 19, 2021
Regarding: EELE 465, Lab 1 – LED Heartbeat

Summary:

The purpose of this lab was to use an MSP430-FR2355 microprocessor to interface with an LED bar such that it blinks in different patterns when a matrix keypad button is pressed. This task was accomplished by interfacing first with the keypad to check what button was pressed, then interface with the LED bar to provide the correct patterns.

Setup:

This project required a fair amount of wiring to set up. Ports 1.0 – 1.7 were used to interface with the keypad and ports 6.0 through 6.7 (excluding 6.5 and 6.7) as well as port 2.5 and port 3.7 were used to work with the LED bar. In between the LED bar and the microprocessor sat a 101-ohm resistor. Figure 1 depicts a circuit diagram for this experiment.

Step 1 – Interfacing with the keypad:

The first part of this project was to correctly interface with the keypad. This was done by creating a subroutine called "CheckKeypad" in Figure 2, which confirmed which button was pressed. During the first portion of the subroutine, ports 1.0-1.3 were set as inputs with a pull up resistor and ports 1.4-1.7 were set as enabled outputs. When a button was pressed on the keypad, this portion of the subroutine computed which column was pressed, as the column would return a logic high. Then in the next portion of the subroutine, each input row was checked to determine the actual button pressed. For reference, button 1 was determined as 0x88 and button D was determined as 0x11. This was stored in register R6 such that other subroutines have access to it.

Step 2 – Interfacing with the LED bar:

The second part of the project was to connect the LED bar to the microprocessor and interface with the keypad. Producing the first pattern was straightforward as it required to set every other port to a logic high. Because the button 'A' had to be pressed to produce this pattern, the "CheckKeypad" subroutine was called prior and a compare was made to check whether 'A' was pressed. The same logic was used to check if the second pattern was chosen, by pressed the button 'B'. This pattern consisted of a binary counter and each LED bar would correspond to the current count in binary form; the LED on signified that it was a logic 1 and off was a logic 0. It was crucial to use ports from Px.0 to Px.7 because the number was sent to the output ports, which is why P2.5 and P3.7 were used. Each count was separated by a 1s delay. A port 1 interrupt was also implemented such that upon clicking other buttons the counter would either reset or finish. This was accomplished by initializing and enabling a port 1 interrupt inside of the LED pattern subroutine "LED2" in Figure 3, which evidently got disabled at the end of the subroutine to not interfere with the other patterns.

Step 3 – Creating a passcode for the LED bar

The last part of this project was to implement a passcode to unlock the LED pattern abilities. If the correct keypad button was selected in the correct order, the user was able to enable the patterns. This was accomplished inside the "KeyPass" subroutine, Figure 4, where the program waited in an infinite loop until the user correctly pressed on the right button. The passcode used here was 4351.

Conclusion:

This project introduced the keypad and the concept behind how initializing the part correctly using pull up resistors. This was then connected to an LED bar where the patterns can be modified by clicking on different buttons. These patterns were static (pattern XOXOXOXO) and dynamic (binary counter). The binary counter needed an interrupt subroutine to determine if another button was pressed to whether continue the counter, reset it, or terminate it. Finally, a mild security system was put in place where a keycode of 4351 needed to be entered to get access to the LED patterns.

Lessons Learned:

- Keypad uses pull up resistors
- Get more wires for next project, only had 1 spare
- Resistors work in magnitudes – 100 ohm resistor is pretty similar to 300 ohm – for LEDs

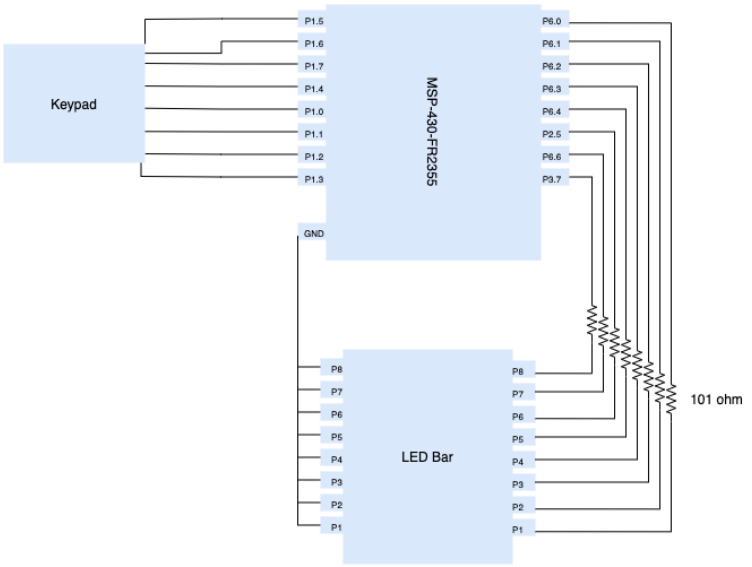
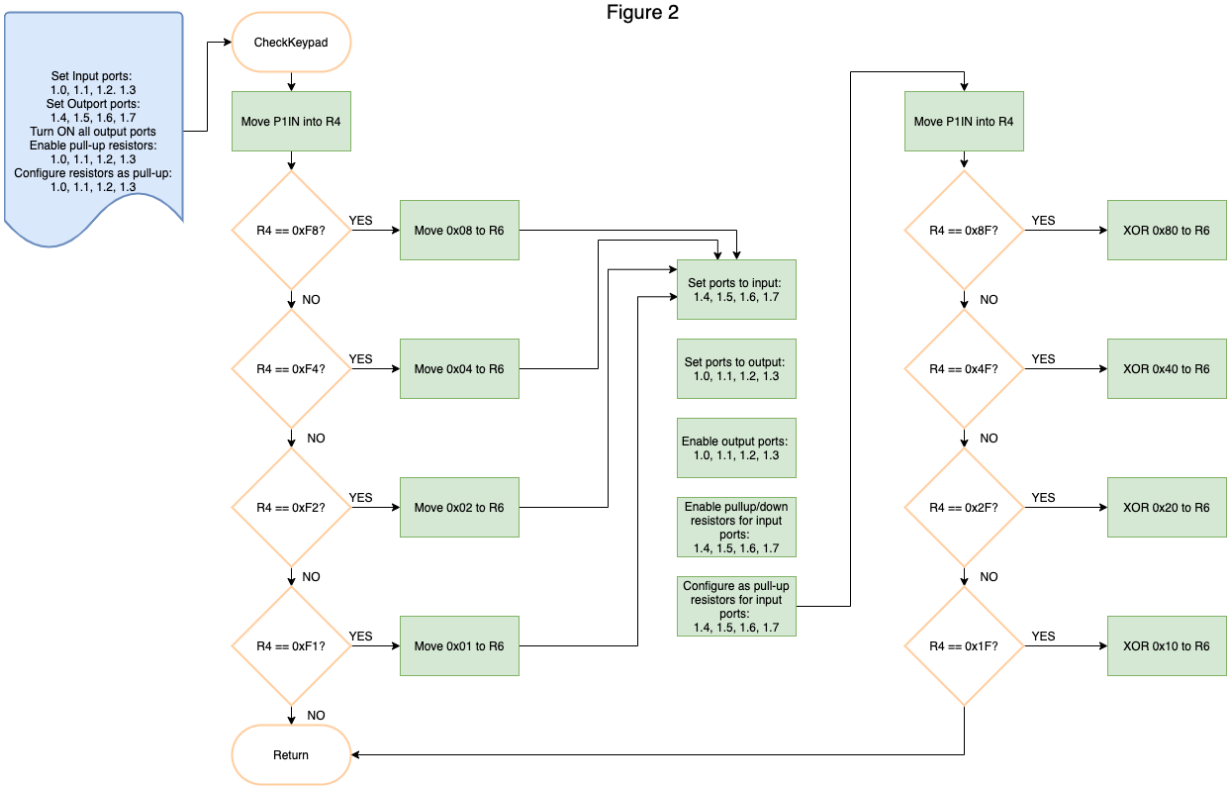


Figure 1

Figure 2



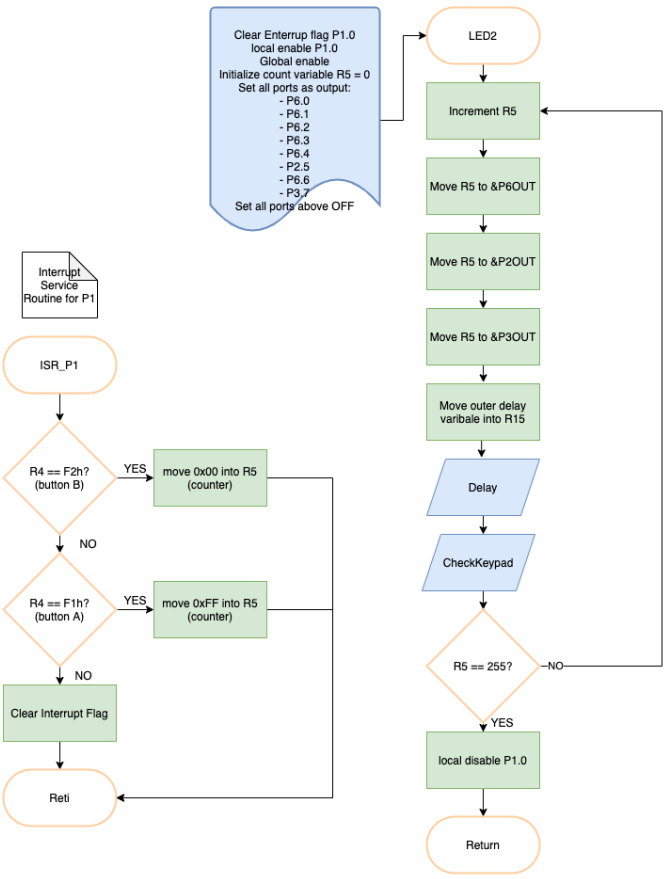


Figure 3

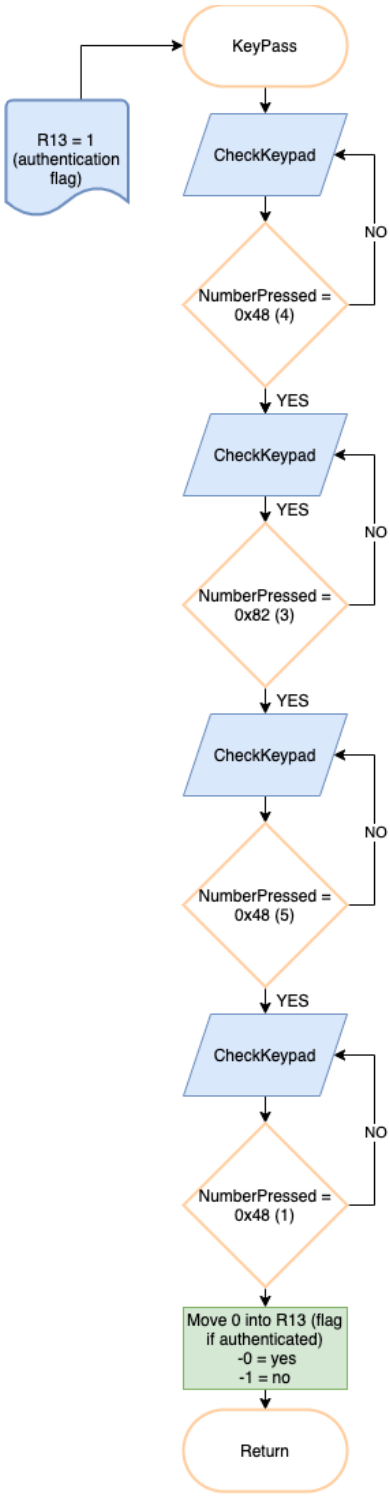


Figure 4