ECE 2049 A23
Lab 0 Report
Ryan Bendremer, Christian Piper

Introduction:

The objective of this lab was to familiarize ourselves with the Code Composer Studio development environment and gain practical experience in programming and debugging our custom MSP430F5529 Launchpad-based Lab Board. This hands-on session aimed to equip us with the foundational knowledge and skills required for successful embedded system programming and development. Throughout the lab, we followed a series of structured steps to assemble the lab board, connect it to our computer, and utilize Code Composer Studio 10.4 to program and control the board's functions. The key accomplishment of this lab was achieving our first embedded program which made the red LED on the lab board blink. This simple task demonstrated our ability to write code, build projects, and debug them effectively using Code Composer Studio. In this report, we will detail the specific steps taken, provide insights into the techniques employed to achieve our goal, and offer explanations of the underlying principles that made our success possible.

Discussion and Results:

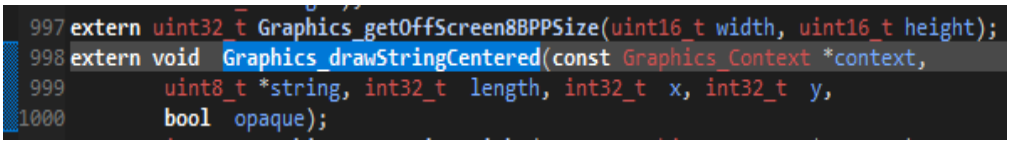
At the beginning of this lab, our initial focus was on ensuring the setup and assembly of the lab IO board and the MSP430F5529 Launchpad. Once we completed these one-time formatting tasks, we were ready to work on our first program.

In the 'blink.c' code for the MSP430F5529 lab board, the behavior of the red LED is controlled by writing a 1 or 0 to the third bit of the port 6 output register. This is achieved through an XOR operation (^) between the Port 6 Out register (P6OUT) and the constant BIT2, which is defined in 'msp430f5529.h' as 0x0004 (00000100b). This XOR operation essentially

toggles the state of pin 2 by flipping the respective bit in the output register. When a 1 is written to the pin's bit, it turns on the LED, and when a 0 is written, it turns off. This mechanism allows us to control the LED's state in the code, making it blink as desired.

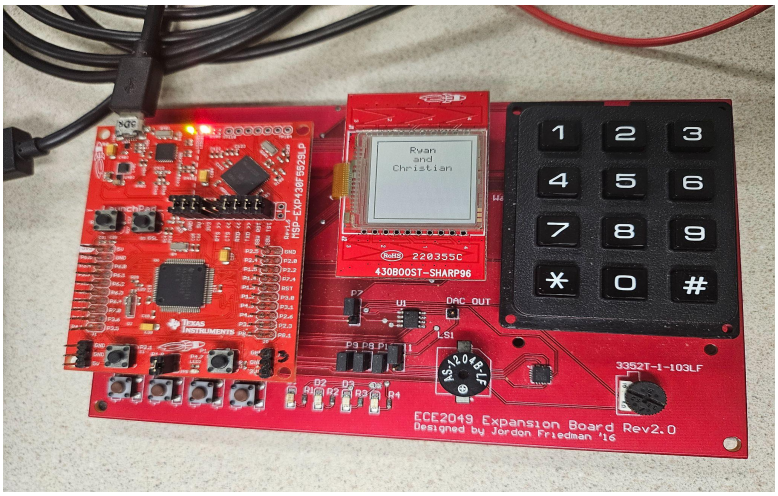
In 'blink.c,' there are two commented lines of code below the XOR statement. By uncommenting and experimenting with these lines, we can determine whether a 1 or 0 lights the LED. Through testing, it was observed that the LED lights when a 1 is written to pin 2's bit in the port 6 output register. This result aligns with our understanding of the XOR operation's behavior, where the third bit is flipped on and off by comparing it with BIT2 (00000100b).

In the demo project, we explored and modified the 'main.c' file to perform various tasks. First, we manipulated the LCD write commands ('Graphics_drawStringCentered') to reposition text on the screen. This allowed us to control the position of the text and change what was displayed.



LCD write commands

Additionally, we utilized the 'expand declaration' tool by right-clicking on functions, which proved to be a valuable asset during debugging and understanding code flow. We declared an array of characters, initialized it with our name, and wrote our name to the LCD. To answer a critical question about the array, we confirmed that including a NULL terminator ('\n') as the last element is essential, as it signifies the end of the character array.



Writing our names on the LCD

Furthermore, we investigated variable sizes used by the Code Composer MSP430 Compiler for different variable types, including float (32 bits), int (16 bits), long integer (32 bits), and char (8 bits). We found that the ASCII code for "A" is 65 (value stored in myGrade) and the new value of test is 1536. The variable myGrade is implicitly cast to an integer before the subtraction happens. We also determined that the buzzer sounds when the asterisk key on the keypad is pressed and turns off when the pound key is pressed. Lastly, when a number key is pressed, we established that the 4 colored LEDs display the binary representation of the key's value.

Summary and Conclusion:

In conclusion, this lab has been significant in our introduction to embedded system programming and development using the Code Composer Studio environment. We began with

the fundamental task of safely handling and setting up our MSP430F5529 Launchpad-based Lab Board. Subsequently, we delved into the intricacies of controlling the red LED on the board by toggling a 1 and 0 in the port 6 output register, gaining a practical understanding of how Code Composer Studio facilitates embedded programming.

Our exploration extended to the demo project, where we not only modified code to reposition text on the LCD but also examined variable sizes and grasped the relationships between the keypad and colored LEDs. Through these hands-on experiences, we honed our skills in debugging and manipulating embedded systems, setting a solid foundation for more complex tasks ahead. This lab's key achievement, making the red LED blink as our first embedded program, exemplifies our initial success in applying Code Composer Studio and understanding its interaction with hardware components. As we move forward, the knowledge and skills gained in this lab will undoubtedly be invaluable in tackling more advanced challenges.