CPE403 – Advanced Embedded Systems

Design Assignment #04

DO NOT REMOVE THIS PAGE DURING SUBMISSION:

Name: Cameron Kirk

Email: kirkc1@unlv.nevada.edu

Github Repository link (root): https://github.com/kirkster96/DqF514-not-embedded

Youtube Playlist link (root):

https://www.youtube.com/playlist?list=PLiqmqQ7XKuf7ArV7lO6b20D1ES5SUp0Yk

Follow the submission guideline to be awarded points for this Assignment.

Submit the following:

- Demo (Video link text file), Document, and Code.
- Documentation: Submit the midterm report with 1) Goal, 2) Detailed Implementation, 3) Schematics, 4) Video links, 5) Screenshots, and 6) Conclusions (tasks completed).

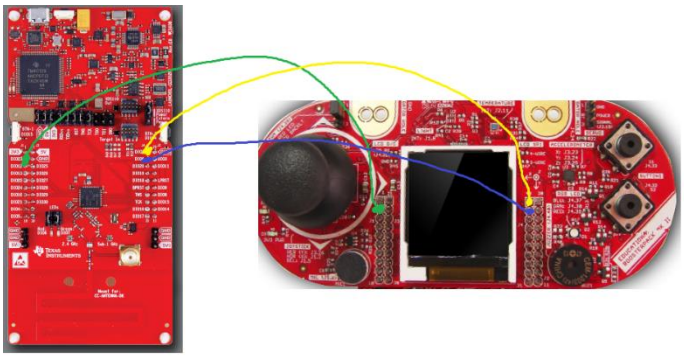
1. Goal

The goal of this assignment was to combine the CC1352R with the Educational BoosterPack MKII and to implement TI-RTOS and SysConfig to interface the two devices. There is to be a task for the ADC 0 to read from the Analog stick, a task to print the ADC value to the terminal via UART, a for the PWM, and a timer that posts the ADC and UART and also adjusts the duty cycle of the PWM. These tasks should be synchronized using a timer Hardware Interrupt every 5ms. There should also be a heartbeat idle task that blinks the LED throughout the execution of the program.

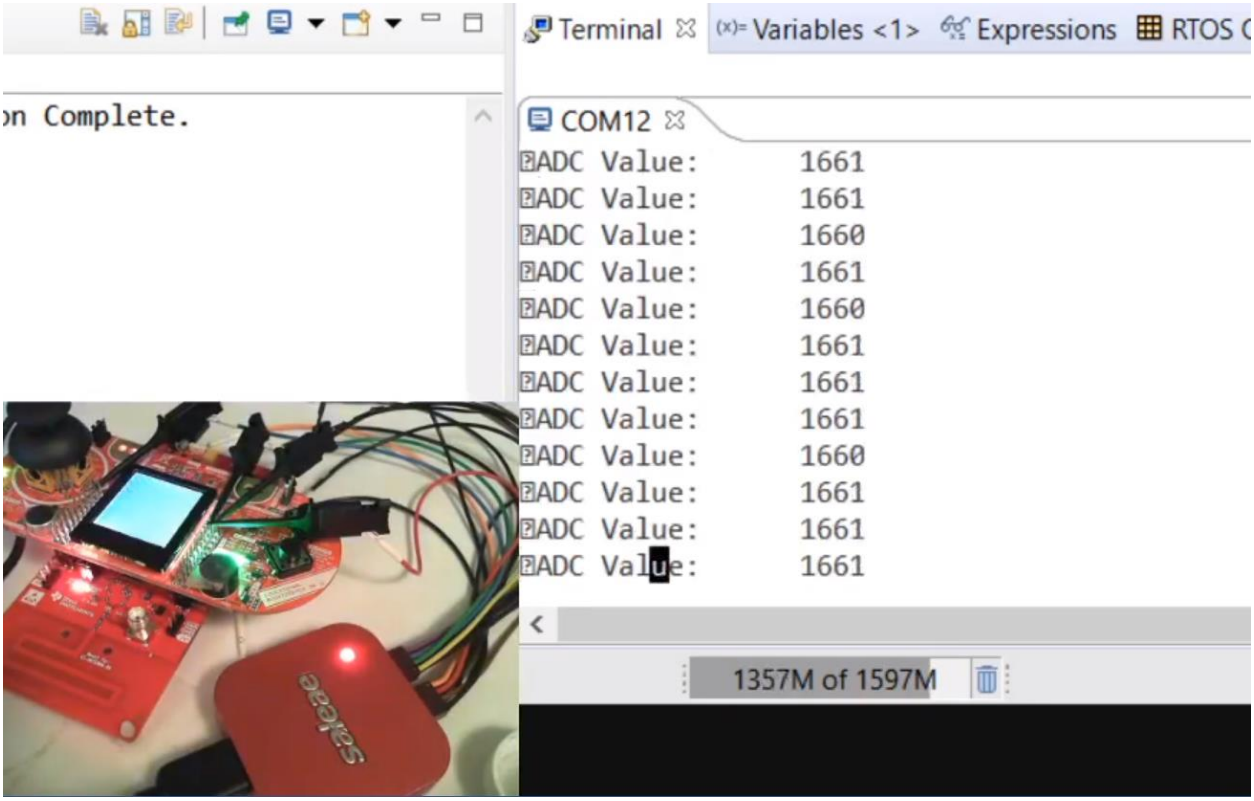
2. Detailed Implementation

Educational BoosterPack MKII Launchpad configuration with the CC1352R. ADC0 is used on pin DIO23. The red LED is controlled on pin DIO6. The green LED is controlled on DIO7. The UART code was developed using the TI uartecho.c example code and the UART_write API. The ADC task code was developed using the TI adcsinglechannel project from the resource explorer. The heartbeat function was created from the TI empty.c example project. The timer HWI code was developed using the timerled project from the resource explorer. The semaphores are defined in the asgn4.cfg GUI.

3. Schematics



4. Screenshots



5. Video Links

https://youtu.be/nbymOxet0ys

6. Conclusion

In conclusion, TI-RTOS can significantly speed up the development process for complicated applications and allow for seamless integration and orchestration of multiple tasks to be performed by a microcontroller CPU. The important thing to recognize when using a RTOS is that your tasks are now each going to be scheduled and share execution time on the CPU. This means that the context switching and priority of tasks need to be taken into consideration. There are circumstances where a task can enter a critical section and if this is overlooked then the program will fail.

The ADC, UART, PWM, Timer HWI, and Heartbeat task have been successfully implemented.

I understand the Student Academic Misconduct Policy -
http://studentconduct.unlv.edu/misconduct/policy.html

"This assignment submission is my own, original work".
Cameron Kirk