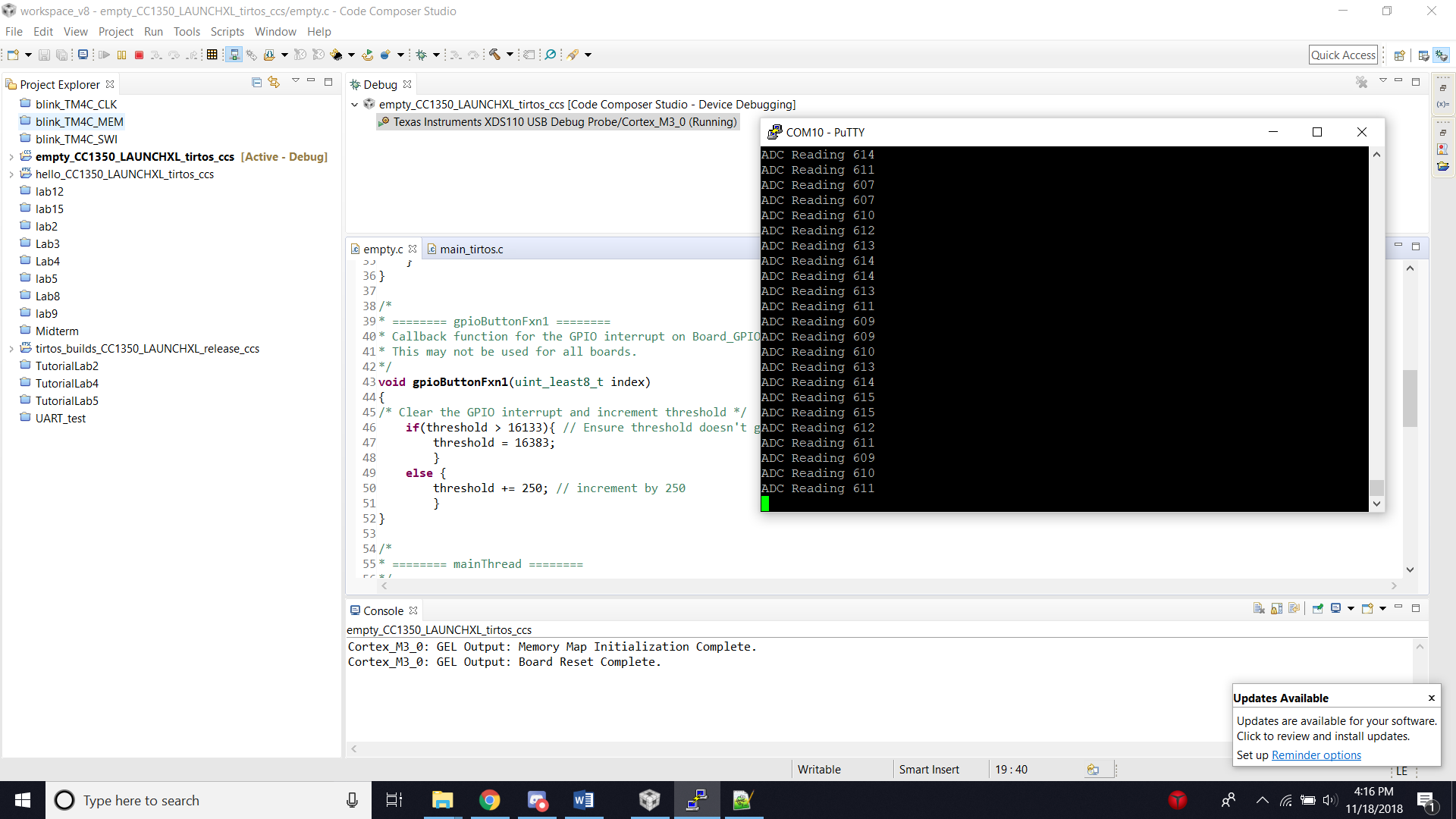
Adrian Ruiz

Cpe 403

CC1350 Lab 5

Screen Shot:



Objective: Read an ADC value from a pin and display it on the terminal. Buttons 0 and 1 are used to increase a threshold which turns on the LED if the ADC value is greater than the threshold.

Code:

/\*

\* ======== empty.c ========

\*/

/\* For usleep() \*/

#include <unistd.h>

#include <stdint.h>

#include <stddef.h>

/\* Driver Header files \*/

#include <ti/drivers/GPIO.h>

#include <ti/drivers/ADC.h>

#include <ti/display/Display.h>

// #include <ti/drivers/I2C.h>

// #include <ti/drivers/SDSPI.h>

// #include <ti/drivers/SPI.h>

// #include <ti/drivers/UART.h>

// #include <ti/drivers/Watchdog.h>

/\* Board Header file \*/

#include "Board.h"

/\* GLOBAL VARIABLES FOR GUI COMPOSER \*/

uint16\_t adcValue **=** 0**;**

uint16\_t threshold **=** 100**;**

uint16\_t trigger **=** 0**;**

/\*

\* ======== gpioButtonFxn0 ========

\* Callback function for the GPIO interrupt on Board\_GPIO\_BUTTON0.

\*/

void gpioButtonFxn0**(**uint\_least8\_t index**)**

**{**

/\* Clear the GPIO interrupt and decrement threshold \*/

**if(**threshold **<** 250**){** // Ensure threshold doesn't go below zero

threshold **=** 0**;**

**}**

**else** **{**

threshold **-=** 250**;** // decrement by 250

**}**

**}**

/\*

\* ======== gpioButtonFxn1 ========

\* Callback function for the GPIO interrupt on Board\_GPIO\_BUTTON1.

\* This may not be used for all boards.

\*/

void gpioButtonFxn1**(**uint\_least8\_t index**)**

**{**

/\* Clear the GPIO interrupt and increment threshold \*/

**if(**threshold **>** 16133**){** // Ensure threshold doesn't go above max ADC range

threshold **=** 16383**;**

**}**

**else** **{**

threshold **+=** 250**;** // increment by 250

**}**

**}**

/\*

\* ======== mainThread ========

\*/

void **\***mainThread**(**void **\***arg0**)**

**{**

/\* ~10 loops/second \*/

uint32\_t time **=** 100000**;**

/\* Call driver init functions \*/

GPIO\_init**();**

ADC\_init**();**

// I2C\_init();

// SDSPI\_init();

// SPI\_init();

// UART\_init();

// Watchdog\_init();

/\* Open Display Driver \*/

Display\_Handle displayHandle**;**

Display\_Params displayParams**;**

Display\_Params\_init**(&**displayParams**);**

displayHandle **=** Display\_open**(**Display\_Type\_UART**,** **NULL);**

/\* Open ADC Driver \*/

ADC\_Handle adc**;**

ADC\_Params params**;**

ADC\_Params\_init**(&**params**);**

adc **=** ADC\_open**(**Board\_ADC0**,** **&**params**);**

**if** **(**adc **==** **NULL)** **{**

// Error initializing ADC channel 0

**while** **(**1**);**

**}**

GPIO\_setCallback**(**Board\_GPIO\_BUTTON0**,** gpioButtonFxn0**);**

GPIO\_setCallback**(**Board\_GPIO\_BUTTON1**,** gpioButtonFxn1**);**

/\* Enable interrupts \*/

GPIO\_enableInt**(**Board\_GPIO\_BUTTON0**);**

GPIO\_enableInt**(**Board\_GPIO\_BUTTON1**);**

**while** **(**1**)** **{**

int\_fast16\_t res**;**

res **=** ADC\_convert**(**adc**,** **&**adcValue**);**

**if** **(**res **==** ADC\_STATUS\_SUCCESS**)** **{**

Display\_printf**(**displayHandle**,** 1**,** 0**,** "ADC Reading %d"**,** adcValue**);**

**if(**adcValue **>=** threshold**){**

GPIO\_write**(**Board\_GPIO\_LED0**,** Board\_GPIO\_LED\_ON**);**

trigger **=** 1**;**

**}**

**else{**

GPIO\_write**(**Board\_GPIO\_LED0**,** Board\_GPIO\_LED\_OFF**);**

trigger **=** 0**;**

**}**

**}**

usleep**(**time**);**

**}**

**}**