Geovanni Portillo
Github root directory: https://github.com/portig1/submissions_E2

Date Submitted: **November 12, 2019**

Task 01 (ADC Gui):

Youtube Link: https://youtu.be/0uyVUwZCRqo

Modified Code:

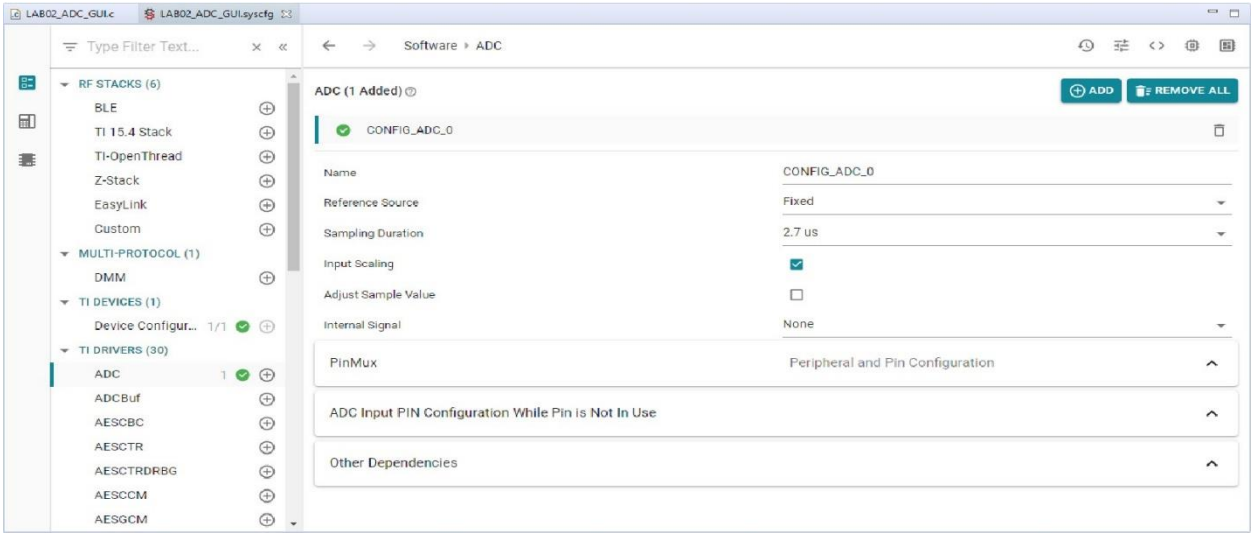
```
/* For usleep() */
  #include <unistd.h>
  #include <stdint.h>
  #include <stddef.h>

  /* Driver Header files */
  #include <ti/drivers/GPIO.h>
  #include <ti/drivers/ADC.h>
  // #include <ti/drivers/I2C.h>
  // #include <ti/drivers/SPI.h>
  // #include <ti/drivers/UART.h>
  // #include <ti/drivers/Watchdog.h>

  /* Driver configuration */
  #include "ti_drivers_config.h"

  /* global variableS FOR GUI COMPOSER */
  uint16_t adcValue = 0;
  uint16_t threshold = 100;
  uint16_t alert = 0;

  /*
   *  ======== mainThread ========
   */

  void *mainThread(void *arg0)
  {
      /* ~10 loops/second */
      uint32_t time = 100000;  // update ~10/second

      /* Call driver init functions */
      GPIO_init();
      ADC_init();
      // I2C_init();
      // SPI_init();
      // UART_init();
      // Watchdog_init();

      /* Open ADC Driver */
      ADC_Handle adc;
      ADC_Params params;
      ADC_Params_init(&params);
      adc = ADC_open(CONFIG_ADC_0, &params);
      if (adc == NULL) {
```

```
        // Error initializing ADC channel 0
        while (1);
    }

    while (1) {
        int_fast16_t res;

        res = ADC_convert(adc, &adcValue);
        if (res == ADC_STATUS_SUCCESS) {
            if(adcValue >= threshold) {
                GPIO_write(CONFIG_GPIO_LED_0, CONFIG_GPIO_LED_ON);
                alert = 1;
            } else{
                GPIO_write(CONFIG_GPIO_LED_0, CONFIG_GPIO_LED_OFF);
                alert = 0;
            }
        }

        usleep(time);
    }
}
```

Screenshots of ROV view and SysCfg Modifications

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Geovanni Portillo
Github root directory: https://github.com/portig1/submissions_E2



Task 02 (RTOS Example):

Youtube Link: https://youtu.be/dQ99x-8o9Iw

Modified 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/SPI.h>
// #include <ti/drivers/UART.h>
// #include <ti/drivers/Watchdog.h>

/* Driver configuration */
#include "ti_drivers_config.h"

 /* global variableS FOR GUI COMPOSER */
  uint16_t adcValue = 0;
  uint16_t threshold = 100;
  uint16_t trigger = 0;


  /*
  * ======== gpioButtonFxn0 ========
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

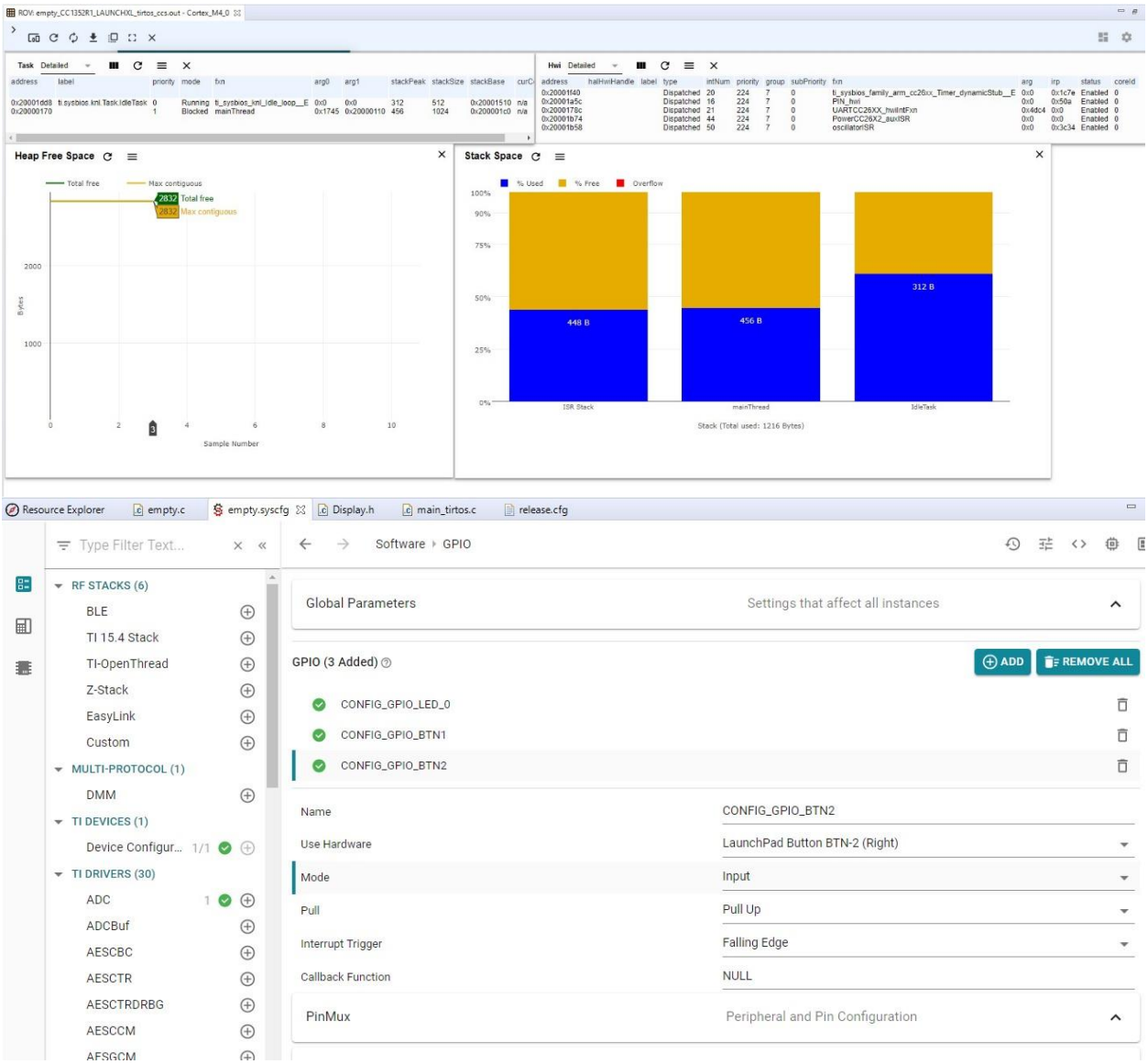
```
* Callback function for the GPIO interrupt on Board_GPIO_BUTTON0.
*/
void gpioButtonFxn0(uint_least8_t index)
{
    /* Clear the GPIO interrupt and decrement threshold */
    if(threshold < 250){ // Ensure threshold doesn't go below zero
        threshold = 0;
    }
    else {
        threshold -= 250; // decrement by 250
    }
}

void gpioButtonFxn1(uint_least8_t index)
{
    /* Clear the GPIO interrupt and increment threshold */
    if(threshold > 4095){ // Ensure threshold doesn't go above max ADC range
        threshold = 4095;
    }
    else {
        threshold += 250; // increment by 250
    }
}

/*
* ======== gpioButtonFxn1 ========
* Callback function for the GPIO interrupt on Board_GPIO_BUTTON1.
* This may not be used for all boards.
*/


/*
   *  ======== mainThread ========
   */

  void *mainThread(void *arg0)
  {
    /* ~10 loops/second */
    uint32_t time = 100000;  // update ~10/second

    /* 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);
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

```
    /* Open ADC Driver */
    ADC_Handle adc;
    ADC_Params params;
    ADC_Params_init(&params);
    adc = ADC_open(CONFIG_ADC_0, &params);
    if (adc == NULL) {
        // Error initializing ADC channel 0
        while (1);
    }

    /* install Button callback */

    GPIO_setCallback(CONFIG_GPIO_BTN1, gpioButtonFxn0);
    GPIO_setCallback(CONFIG_GPIO_BTN2, gpioButtonFxn1);



    /* Enable interrupts */
    GPIO_enableInt(CONFIG_GPIO_BTN1);
    GPIO_enableInt(CONFIG_GPIO_BTN2);

    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(CONFIG_GPIO_LED_0, CONFIG_GPIO_LED_ON);
          trigger = 1;
        } else{
          GPIO_write(CONFIG_GPIO_LED_0, CONFIG_GPIO_LED_OFF);
          trigger = 0;
        }
      }

      usleep(time);
    }
  }
```

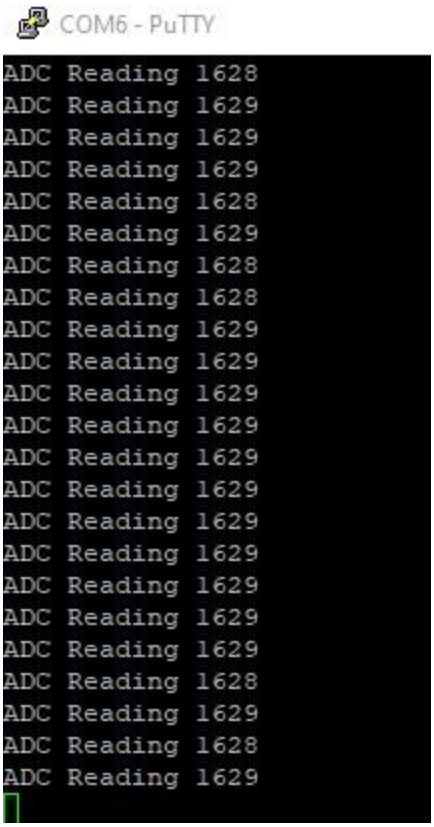
Screenshots of ROV view and SysCfg Modifications

Geovanni Portillo
Github root directory: https://github.com/portig1/submissions_E2



Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Geovanni Portillo
Github root directory: https://github.com/portig1/submissions_E2



Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.