Github: https://github.com/Ber-geb/solid-octo-tribble.git

CPE 403 ADV EMB SYS DES F 2019

TITLE: TIRTOS TIVAC Assignment

GOAL:

- Create ADC task to run every 10th instance of HWI
- Create UART diplay task to run every 20th instance of HWI
- Create Switch/Read Task to run every 30th instance of HWI
- Repeat the process above every 30 ms

DELIVERABLES:

The project will show an LED that is affected by the PWM signal that takes the ADC value generated every 10th instance of the HWI. Also, a terminal will be shown that is connected to the same port as the TIVAC TM4C123GH6PM MCU to show the UART signals being transmitted/received. Every time SW0/SW1 is pressed, the duty cycle will change. Since the period of the PWM is very small, a logic analyzer will be shown which will convey when the switch is pressed to affect the PWM signal of the LED.

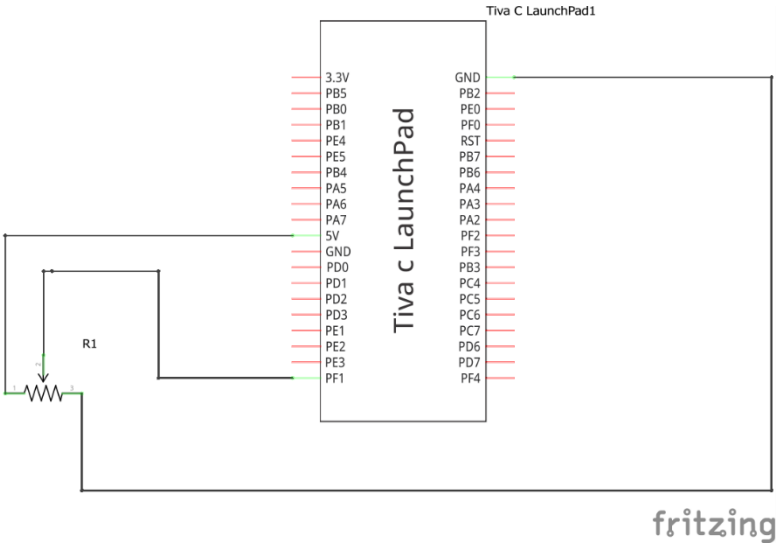
COMPONENTS:

TIVAC TM4C123GH6PM MCU

- Logic Analyzer
- Jumper Wires
- Potentiometer

SCHEMATICS:

Github: https://github.com/Ber-geb/solid-octo-tribble.git



IIMPLEMENTATION:

UART and GPIO will be initialized as well as the ADC. This is the major initializations made for the assignment. The code below will show these initializations.

Github: https://github.com/Ber-geb/solid-octo-tribble.git

CODE:

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
/*
 * Copyright (c) 2015, Texas Instruments Incorporated
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
// *
// * *  Redistributions of source code must retain the above copyright
// *    notice, this list of conditions and the following disclaimer.
// *
// * *  Redistributions in binary form must reproduce the above copyright
// *    notice, this list of conditions and the following disclaimer in the
// *    documentation and/or other materials provided with the distribution.
// *
// * *  Neither the name of Texas Instruments Incorporated nor the names of
// *    its contributors may be used to endorse or promote products derived
// *    from this software without specific prior written permission.
// *
// * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
// * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
// * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
// * PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
// * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
// * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
// * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS;
// * OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
// * WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
// * OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
// * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
// */
//
///*
// *  ======== empty.c ========
// */
//
////-------------------------------------
//// Prototypes
////-------------------------------------
//void hardware_init(void);
//
//
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
//#include <stdbool.h>
//
////-----------------------------------------
//// BIOS header files
////-----------------------------------------
//#include <xdc/std.h>                        //mandatory - have to include first, for BIOS types
//#include <ti/sysbios/BIOS.h>                //mandatory - if you call APIs like BIOS_start()
//#include <xdc/runtime/Log.h>                //needed for any Log_info() call
//#include <xdc/cfg/global.h>                 //header file for statically defined objects/handles
//
//
///* XDCtools Header files */
//#include <xdc/runtime/System.h>
//
///* BIOS Header files */
//#include <ti/sysbios/knl/Task.h>
//
///* TI-RTOS Header files */
//#include <ti/drivers/GPIO.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>
// #include <ti/drivers/WiFi.h>
//
///* Board Header file */
//#include "Board.h"
//
//#include "driverlib/adc.h"
//#include "inc/hw_memmap.h"
//#include "driverlib/sysctl.h"
//#include "driverlib/timer.h"
//#include "driverlib/interrupt.h"
//
//
//#define TASKSTACKSIZE   512
//
//Task_Struct task0Struct;
//Char task0Stack[TASKSTACKSIZE];
//
///*
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
85  // *  ======== heartBeatFxn ========
86  // *  Toggle the Board_LED0. The Task_sleep is determined by arg0 which
87  // *  is configured for the heartBeat Task instance.
88  // */
89  //Void heartBeatFxn(UArg arg0, UArg arg1)
90  //{
91  //    while (1) {
92  //        Task_sleep((UInt)arg0);
93  //        GPIO_toggle(Board_LED0);
94  //    }
95  //}
96  //
97  ///*
98  // *  ======== main ========
99  // */
100
101 //---------------------------------------
102 // BIOS header files
103 //---------------------------------------
104 #include <xdc/std.h>                        //mandatory - have to include first, for BIOS types
105 #include <ti/sysbios/BIOS.h>                //mandatory - if you call APIs like BIOS_start()
106 #include <xdc/runtime/Log.h>                //needed for any Log_info() call
107 #include <xdc/cfg/global.h>                 //header file for statically defined objects/handles
108
109
110 //-----------------------------------------
111 // TivaWare Header Files
112 //-----------------------------------------
113 #include <stdint.h>
114 #include <stdbool.h>
115
116 #include "inc/hw_types.h"
117 #include "inc/hw_memmap.h"
118 #include "driverlib/sysctl.h"
119 #include "driverlib/gpio.h"
120 #include "inc/hw_ints.h"
121 #include "driverlib/interrupt.h"
122 #include "driverlib/timer.h"
123 #include "driverlib/adc.h"
124 #include "utils/uartstdio.h"
125 #include "driverlib/uart.h"
126 #include "driverlib/pin_map.h"
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
127  #include "driverlib/pwm.h"
128
129  //*****************************************
130  // Prototypes
131  //*****************************************
132  void hardware_init(void);
133  void ledToggle(void);
134  void TIMER_ISR(void);
135  void TIMER2INT(void);
136  void ADCfun(void);
137  void SRfun(void);
138  void UARTfun(void);
139  void reverse(char[], int);
140  char* itoa(int,char*,int);
141  void InitConsole(void);
142
143  #define PWM_FREQUENCY 55
144
145  volatile int16_t i16ToggleCount;
146  uint32_t ui32ADC0Value[4];
147  volatile bool buttonPressed;
148  volatile uint32_t ui32Load;
149  volatile uint32_t ui32PWMClock;
150  volatile uint32_t ui8Adjust = 83;
151
152  int main(void)
153  {
154      //    Task_Params taskParams;
155      //
156      //    /* Call board init functions */
157      //    Board_initGeneral();
158      //    //Board_initGPIO();
159      //    // Board_initI2C();
160      //    // Board_initSDSPI();
161      //    // Board_initSPI();
162      //    //Board_initUART();
163      //    // Board_initUSB(Board_USBDEVICE);
164      //    // Board_initWatchdog();
165      //    // Board_initWiFi();
166      //
167      //    /* Construct heartBeat Task  thread */
168      //    Task_Params_init(&taskParams);
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
169     //    taskParams.arg0 = 1000;
170     //    taskParams.stackSize = TASKSTACKSIZE;
171     //    taskParams.stack = &task0Stack;
172     //    Task_construct(&task0Struct, (Task_FuncPtr)heartBeatFxn, &taskParams, NULL);
173     //
174     //    /* Turn on user LED */
175     //    GPIO_write(Board_LED0, Board_LED_ON);
176     buttonPressed = false;
177     hardware_init();
178     //
179     //    System_printf("Starting the example\nSystem provider is set to SysMin. "
180     //            "Halt the target to view any SysMin contents in ROV.\n");
181     //    /* SysMin will only print to the console when you call flush or exit */
182     //    System_flush();
183
184     /* Start BIOS */
185     BIOS_start();
186
187 }
188
189 //-------------------------------------------------------------------------
190 // hardware_init()
191 //
192 // inits GPIO pins for toggling the LED
193 //-------------------------------------------------------------------------
194 void hardware_init(void)
195 {
196     uint32_t ui32Period;
197
198     i16ToggleCount = 0;
199
200     // Board_initUART();
201
202     //Set CPU Clock to 40MHz. 400MHz PLL/2 = 200 DIV 5 = 40MHz
203     SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_XTAL_16MHZ|SYSCTL_OSC_MAIN);
204     SysCtlPWMClockSet(SYSCTL_PWMDIV_64);
205
206     SysCtlPeripheralEnable(SYSCTL_PERIPH_PWM1);
207     SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOD);
208
209     GPIOPinTypePWM(GPIO_PORTD_BASE, GPIO_PIN_0); //PD0 PWM pin
210     GPIOPinConfigure(GPIO_PD0_M1PWM0);
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
211
212     // ADD Tiva-C GPIO setup - enables port, sets pins 1-3 (RGB) pins for output
213     SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
214
215     GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);
216     GPIOPinTypeGPIOInput(GPIO_PORTF_BASE, GPIO_PIN_4|GPIO_PIN_0);
217
218     GPIOPadConfigSet(GPIO_PORTF_BASE, GPIO_PIN_4|GPIO_PIN_0 , GPIO_STRENGTH_2MA, GPIO_PIN_TYPE_STD_WPU);
219
220     ui32PWMClock = SysCtlClockGet() / 64;
221     ui32Load = (ui32PWMClock / PWM_FREQUENCY) - 1;
222
223     PWMGenConfigure(PWM1_BASE, PWM_GEN_0, PWM_GEN_MODE_DOWN);
224     PWMGenPeriodSet(PWM1_BASE, PWM_GEN_0, ui32Load);
225
226     PWMPulseWidthSet(PWM1_BASE, PWM_OUT_0, (ui32ADC0Value[0]/200 * ui32Load)/1000);
227     PWMOutputState(PWM1_BASE, PWM_OUT_0_BIT, true);
228     PWMGenEnable(PWM1_BASE, PWM_GEN_0);
229
230     // Turn on the LED
231     GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 4);
232
233     //initialize ADC
234     SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC0);
235     ADCHardwareOversampleConfigure(ADC0_BASE, 64);
236     ADCSequenceConfigure(ADC0_BASE, 1, ADC_TRIGGER_PROCESSOR, 0);
237
238     ADCSequenceStepConfigure(ADC0_BASE, 1, 0, ADC_CTL_TS);
239     ADCSequenceStepConfigure(ADC0_BASE, 1, 1, ADC_CTL_TS);
240     ADCSequenceStepConfigure(ADC0_BASE, 1, 2, ADC_CTL_TS);
241     ADCSequenceStepConfigure(ADC0_BASE, 1, 3, ADC_CTL_TS | ADC_CTL_IE | ADC_CTL_END);
242
243     ADCSequenceEnable(ADC0_BASE, 1);
244
245     // Timer 2 setup code
246     SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER2);           // enable Timer 2 periph clks
247     TimerConfigure(TIMER2_BASE, TIMER_CFG_PERIODIC);        // cfg Timer 2 mode - periodic
248
249     ui32Period = (SysCtlClockGet() / 1000);                   // period = CPU clk div 1000 (1ms)
250     TimerLoadSet(TIMER2_BASE, TIMER_A, ui32Period);         // set Timer 2 period
251
252     TimerIntEnable(TIMER2_BASE, TIMER_TIMA_TIMEOUT);        // enables Timer 2 to interrupt CPU
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
253
254     TimerEnable(TIMER2_BASE, TIMER_A);                      // enable Timer 2
255
256     InitConsole();
257     UARTprintf("WORKING!");
258 }
259
260
261
262 void ADCfun(void){
263     while(1){
264
265         ADCIntClear(ADC0_BASE, 1);
266         ADCProcessorTrigger(ADC0_BASE, 1);
267
268         while (!ADCIntStatus(ADC0_BASE, 1, false)){}
269
270         ADCSequenceDataGet(ADC0_BASE, 1, ui32ADC0Value);
271         //ui32ADC0Value holds the ADC value...choose what to do with it...
272         Semaphore_pend (ADCSem, BIOS_WAIT_FOREVER);
273         //Semaphore_reset(ADCSem, 0);
274         //Semaphore_pend (UARTSem, BIOS_WAIT_FOREVER);
275         //Semaphore_post(UARTSem);
276     }
277 }
278
279 void SRfun(void){
280
281     while(1){
282         if(GPIOPinRead(GPIO_PORTF_BASE, GPIO_PIN_4|GPIO_PIN_0)==0x00)
283         {
284             buttonPressed = true;
285             PWMPulseWidthSet(PWM1_BASE, PWM_OUT_0, (ui32ADC0Value[0]));
286         }
287         //Semaphore_reset(SRSem, 0);
288         //Semaphore_pend (SRSem, BIOS_WAIT_FOREVER);
289         Semaphore_pend (SRSem, BIOS_WAIT_FOREVER);
290
291     }
292 }
293
294 void UARTfun(void){
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
295
296
297
298         while(1){
299             UARTprintf("ADC Value[0]: %d\n", ui32ADC0Value[0]);
300             UARTprintf("ADC Value[1]: %d\n", ui32ADC0Value[1]);
301             UARTprintf("ADC Value[2]: %d\n", ui32ADC0Value[2]);
302             UARTprintf("ADC Value[3]: %d\n", ui32ADC0Value[3]);
303             Semaphore_pend (UARTSem, BIOS_WAIT_FOREVER);
304             //Semaphore_reset(UARTSem, 0);
305             //Semaphore_pend (SRSem, BIOS_WAIT_FOREVER);
306         }
307     }
308
309     void TIMER2INT(void){
310         TimerIntClear(TIMER2_BASE, TIMER_TIMA_TIMEOUT);         // must clear timer flag FROM timer
311         i16ToggleCount = i16ToggleCount + 1; //increment every time HWI occurs
312         //    System_printf("Timer 2 interrupt occurred\n");
313         //    System_flush();
314
315         if (i16ToggleCount == 10){
316             //count = Semaphore_getCount(ADCSem);
317             Semaphore_post (ADCSem);
318         }
319
320         else if (i16ToggleCount == 20){
321             Semaphore_post (UARTSem);
322         }
323
324         else if (i16ToggleCount == 30){
325             Semaphore_post (SRSem);
326             i16ToggleCount = 0;
327         }
328
329         //Semaphore_post(ADCSem);
330     }
331
332     void TIMER_ISR(void){
333         TimerIntClear(TIMER0_BASE, TIMER_TIMA_TIMEOUT);         // must clear timer flag FROM timer
334         if (!buttonPressed){
335             if(GPIOPinRead(GPIO_PORTF_BASE, GPIO_PIN_2))
336             {
337                 GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
338             }
339             else
340             {
341                 GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);
342             }
343         }
344         else {
345 
346             if(GPIOPinRead(GPIO_PORTD_BASE, GPIO_PIN_0))
347             {
348                 GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3, 0);
349             }
350             else
351             {
352                 GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 4);
353             }
354         }
355     }
356 
357     void reverse(char str[], int len){
358         int start, end;
359         char temp;
360         for (start=0, end=len-1; start < end; start++, end--){
361             temp = *(str+start);
362             *(str+start) = *(str+end);
363             *(str+end) = temp;
364         }
365     }
366 
367     char* itoa( int num, char* str, int base){
368         int i = 0;
369         bool isNegative = false;
370 
371         if (num==0){
372             str[i] = '0';
373             str[i+1] = '\0';
374             return str;
375         }
376 
377         if (num < 0 && base == 10) {
378             isNegative = true;
379             num = -num;
380         }
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

```
381
382     while (num!=0){
383         int rem = num % base;
384         str[i++] = (rem > 9) ? (rem - 10) + 'A' : rem + '0';
385         num = num/base;
386     }
387
388     if (isNegative){
389         str[i++] = '-';
390     }
391
392     str[i] = '\0';
393     reverse(str,i);
394     return str;
395 }
396
397 void InitConsole(void){
398     //Enable GPIO port A which is used for UART0 pins
399     SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA);
400     //Configure the pin muxing for UART 0 functions on port A0 and A1
401     //This step is not necessary if your part does not support pin muxing
402     //TODO: change this to select the port/pin you are using.
403     GPIOPinConfigure(GPIO_PA0_U0RX);
404     GPIOPinConfigure(GPIO_PA1_U0TX);
405     //ENable UART0 so that we can configure the clock.
406     SysCtlPeripheralEnable(SYSCTL_PERIPH_UART0);
407     //Use the internal 16MHz oscillator as the UART clock source.
408     UARTClockSourceSet(UART0_BASE, UART_CLOCK_PIOSC);
409     //Select the alternate (UART) function for these pins.
410     GPIOPinTypeUART(GPIO_PORTA_BASE, GPIO_PIN_0 | GPIO_PIN_1);
411     //Initialize the UART for console I/O.
412     UARTStdioConfig(0,115200,16000000);
413 }
```

Github: https://github.com/Ber-geb/solid-octo-tribble.git

Github: https://github.com/Ber-geb/solid-octo-tribble.git

Name: Serak Gebremedhin