CPE403 – Advanced Embedded Systems

# Design Assignment 1

Name: Elmer Mejia

Email: mejiae4@unlv.nevada.edu

Github Repository link (root): [assignments](https://github.com/cpemejia/Launchpad_assignments.git)

Youtube Playlist link (root): [Tiva\_C](https://www.youtube.com/playlist?list=PL6PbL2NpuYCIJs2P3WZJMcGl-yEjS-LJJ)

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

1. Block diagram and/or Schematics showing the components, pins used, and interface.

A picture containing text, electronics, circuit

Description automatically generated



1. Screenshots of the IDE, physical setup, debugging process - Provide screenshot of successful compilation, screenshots of registers, variables, graphs, etc.

Graphical user interface, text, application

Description automatically generated

**#include** <stdint.h>

**#include** <stdbool.h>

**#include** "inc/hw\_memmap.h"

**#include** "inc/hw\_types.h"

**#include** "driverlib/gpio.h"

**#include** "driverlib/pin\_map.h"

**#include** "driverlib/sysctl.h"

**#include** "driverlib/uart.h"

**#include** "inc/hw\_ints.h"

**#include** "driverlib/interrupt.h"

**#include** "driverlib/timer.h"

**#include** "driverlib/debug.h"

**#include** "driverlib/adc.h"

**#include** "utils/uartstdio.h"

**#include** "driverlib/pwm.h"

**#include** "driverlib/pin\_map.h"

**#include** "inc/hw\_gpio.h"

**#include** "driverlib/rom.h"

**#include** "driverlib/rom\_map.h"

**#include** <string.h>

**#define** PWM\_FREQUENCY 100

**#ifdef** DEBUG

void\_\_error\_\_(**char** \*pcFilename, uint32\_t ui32Line)

{

}

**#endif**

// Globals

uint32\_t ui32Period;

**char** buffer[4];

uint32\_t ui32ADC0Value[4];

**volatile** uint32\_t ui32TempAvg;

**volatile** uint32\_t ui32TempValueC;

**volatile** uint32\_t ui32TempValueF;

**volatile** uint32\_t ui32Load;

**volatile** uint32\_t ui32PWMClock;

**volatile** uint8\_t ui8Adjust;

// UART0 ISR

**void** **UARTIntHandler**(**void**)

{

uint32\_t current;

current = **UARTIntStatus**(UART0\_BASE, true);

**UARTIntClear**(UART0\_BASE, current);

//RED on

**if** (**UARTCharGet**(UART0\_BASE) == 'R')

{

**UARTprintf**("Red ON\n");

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1, 2); // Red led

**SysCtlDelay**(10000000);

}

//RED off

**if** (**UARTCharGet**(UART0\_BASE) == 'r')

{

**UARTprintf**("Red OFF\n");

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1, 0);

**SysCtlDelay**(10000000);

}

//GREEN on

**if** (**UARTCharGet**(UART0\_BASE) == 'G')

{

**UARTprintf**("GREEN ON\n");

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_3, 8); //Green LED

**SysCtlDelay**(10000000);

}

//GREEN off

**if** (**UARTCharGet**(UART0\_BASE) == 'g')

{

**UARTprintf**("GREEN OFF\n");

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_3, 0);

**SysCtlDelay**(10000000);

}

//BLUE on

**if** (**UARTCharGet**(UART0\_BASE) == 'B')

{

**UARTprintf**("Blue ON\n");

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, GPIO\_PIN\_2); //Blue LED

**SysCtlDelay**(10000000);

}

//BLUE off

**if** (**UARTCharGet**(UART0\_BASE) == 'b')

{

**UARTprintf**("Blue OFF\n");

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 0);

**SysCtlDelay**(10000000);

}

**if** (**UARTCharGet**(UART0\_BASE) == 'T')

{

**ADCIntClear**(ADC0\_BASE,2);

**ADCProcessorTrigger**(ADC0\_BASE,2);

**while**(!**ADCIntStatus**(ADC0\_BASE, 2, false))

{

}

**ADCSequenceDataGet**(ADC0\_BASE, 2, ui32ADC0Value);

ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value[3] + 2)/4;

ui32TempValueC = (1475 - ((2475 \* ui32TempAvg)) / 4096)/10;

**UARTprintf**("C %3d\t",ui32TempValueC );

**UARTprintf**("\n");

}

**if** (**UARTCharGet**(UART0\_BASE) == 't')

{

**ADCIntClear**(ADC0\_BASE,2);

**ADCProcessorTrigger**(ADC0\_BASE,2);

**while**(!**ADCIntStatus**(ADC0\_BASE, 2, false))

{

}

**ADCSequenceDataGet**(ADC0\_BASE, 2, ui32ADC0Value);

ui32TempAvg = (ui32ADC0Value[0] + ui32ADC0Value[1] + ui32ADC0Value[2] + ui32ADC0Value[3] + 2)/4;

ui32TempValueC = (1475 - ((2475 \* ui32TempAvg)) / 4096)/10;

ui32TempValueF = ((ui32TempValueC \* 9) + 160) / 5;

**UARTprintf**("F %3d\t",ui32TempValueF );

**UARTprintf**("\n");

}

**if** (**UARTCharGet**(UART0\_BASE) == 'P')

{

**UARTprintf**("Brighten");

**UARTprintf**("\n");

// Configure PWM

ui8Adjust = 1;

MAP\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_PWM1);

MAP\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

MAP\_GPIOPinTypePWM(GPIO\_PORTF\_BASE, GPIO\_PIN\_1);

MAP\_GPIOPinConfigure(GPIO\_PF1\_M1PWM5);

ui32PWMClock = **SysCtlClockGet**() / 64;

ui32Load = (ui32PWMClock / PWM\_FREQUENCY) - 1;

**PWMGenConfigure**(PWM1\_BASE, PWM\_GEN\_2, PWM\_GEN\_MODE\_DOWN);

**PWMGenPeriodSet**(PWM1\_BASE, PWM\_GEN\_2, ui32Load);

MAP\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, ui8Adjust \* ui32Load / 100);

MAP\_PWMOutputState(PWM1\_BASE, PWM\_OUT\_5\_BIT, true);

MAP\_PWMGenEnable(PWM1\_BASE, PWM\_GEN\_2);

**while**(ui8Adjust < 100){

ui8Adjust++;

MAP\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, ui8Adjust \* ui32Load / 100);

MAP\_SysCtlDelay(1000000);

}

ui8Adjust = 1;

MAP\_PWMOutputState(PWM1\_BASE, PWM\_OUT\_5\_BIT, false);

}

**if** (**UARTCharGet**(UART0\_BASE) == 'F')

{

**UARTprintf**("Flash Blue");

**UARTprintf**("\n");

**IntEnable**(INT\_TIMER0A);

**TimerIntEnable**(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);

**TimerEnable**(TIMER0\_BASE, TIMER\_A);

}

}

**void** **Timer0IntHandler**(**void**)

{

// Clear the timer interrupt

**TimerIntClear**(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);

// Read the current state of the GPIO pin and

// write back the opposite state

**if**(**GPIOPinRead**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2))

{

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, 0);

}

**else**

{

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 4);

}

}

**int** **main**(**void**) {

// Configure Clock

**SysCtlClockSet**(SYSCTL\_SYSDIV\_4 | SYSCTL\_USE\_PLL | SYSCTL\_OSC\_MAIN | SYSCTL\_XTAL\_16MHZ);

//MAP\_SysCtlClockSet(SYSCTL\_SYSDIV\_5|SYSCTL\_USE\_PLL|SYSCTL\_OSC\_MAIN|SYSCTL\_XTAL\_16MHZ);

MAP\_SysCtlPWMClockSet(SYSCTL\_PWMDIV\_64);

// Configure peripherals

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_UART0);

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOA);

// Configure Timer

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_TIMER0);

**TimerConfigure**(TIMER0\_BASE, TIMER\_CFG\_PERIODIC);

ui32Period = (**SysCtlClockGet**() / 10)/2;

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period -1);

// Configure ADC

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_ADC0);

**ADCHardwareOversampleConfigure**(ADC0\_BASE, 32);

**ADCSequenceConfigure**(ADC0\_BASE, 2, ADC\_TRIGGER\_PROCESSOR, 0); // Changed to sequencer #2

**ADCSequenceStepConfigure**(ADC0\_BASE, 2, 0, ADC\_CTL\_TS);

**ADCSequenceStepConfigure**(ADC0\_BASE, 2, 1, ADC\_CTL\_TS);

**ADCSequenceStepConfigure**(ADC0\_BASE, 2, 2, ADC\_CTL\_TS);

**ADCSequenceStepConfigure**(ADC0\_BASE, 2, 3, ADC\_CTL\_TS|ADC\_CTL\_IE|ADC\_CTL\_END);

//Configure LEDs

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOF);

**GPIOPinTypeGPIOOutput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3);

// Configure pins for UART

**GPIOPinConfigure**(GPIO\_PA0\_U0RX);

**GPIOPinConfigure**(GPIO\_PA1\_U0TX);

**GPIOPinTypeUART**(GPIO\_PORTA\_BASE, GPIO\_PIN\_0 | GPIO\_PIN\_1);

**UARTClockSourceSet**(UART0\_BASE, UART\_CLOCK\_PIOSC);

**UARTStdioConfig**(0, 115200, 16000000);

// Enable interrupts

**IntMasterEnable**();

**ADCSequenceEnable**(ADC0\_BASE, 2);

**ADCIntEnable**(ADC0\_BASE, 2);

**IntEnable**(INT\_UART0);

**UARTIntEnable**(UART0\_BASE, UART\_INT\_RX | UART\_INT\_RT);

// Initial message to terminal display

**UARTprintf**("Enter Valid cmd:\n");

**UARTprintf**("R : Red LED on. r: Red LED off.\n");

**UARTprintf**("B : Blue LED on. b: Blue LED off.\n");

**UARTprintf**("G : Green LED on. g: Green LED off.\n");

**UARTprintf**("T : Temp in C. t: Temp in F.\n");

**UARTprintf**("P : Brighten Red LED. F: Flash Blue LED.");

**UARTprintf**("\n");

**while** (1) // Wait forever

{

}

}

1. Declaration

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

“This assignment submission is my own, original work”.

-Elmer Mejia