**Date Submitted: 10/07/18**

**------------------------------------------------------------------------------------**

**Task 01:**

Youtube Link: https://www.youtube.com/watch?v=Ctht9pyxAFc

**Modified Code:**

**// Insert code here**

**#include** <stdint.h>

**#include** <stdbool.h>

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

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

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

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

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

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

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

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

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

**#define** PWM\_FREQUENCY 55

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

{

**volatile** uint32\_t ui32Load;

**volatile** uint32\_t ui32PWMClock;

**volatile** uint8\_t ui8Adjust;

ui8Adjust = 83;

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

ROM\_SysCtlPWMClockSet(SYSCTL\_PWMDIV\_64);

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_PWM1);

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOD);

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

ROM\_GPIOPinTypePWM(GPIO\_PORTD\_BASE, GPIO\_PIN\_0);

ROM\_GPIOPinConfigure(GPIO\_PD0\_M1PWM0);

HWREG(GPIO\_PORTF\_BASE + GPIO\_O\_LOCK) = GPIO\_LOCK\_KEY;

HWREG(GPIO\_PORTF\_BASE + GPIO\_O\_CR) |= 0x01;

HWREG(GPIO\_PORTF\_BASE + GPIO\_O\_LOCK) = 0;

ROM\_GPIODirModeSet(GPIO\_PORTF\_BASE, GPIO\_PIN\_4|GPIO\_PIN\_0, GPIO\_DIR\_MODE\_IN);

ROM\_GPIOPadConfigSet(GPIO\_PORTF\_BASE, GPIO\_PIN\_4|GPIO\_PIN\_0, GPIO\_STRENGTH\_2MA, GPIO\_PIN\_TYPE\_STD\_WPU);

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

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

**PWMGenConfigure**(PWM1\_BASE, PWM\_GEN\_0, PWM\_GEN\_MODE\_DOWN);

**PWMGenPeriodSet**(PWM1\_BASE, PWM\_GEN\_0, ui32Load);

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_0, ui8Adjust \* ui32Load / 1000);

ROM\_PWMOutputState(PWM1\_BASE, PWM\_OUT\_0\_BIT, true);

ROM\_PWMGenEnable(PWM1\_BASE, PWM\_GEN\_0);

**while**(1)

{

**if**(ROM\_GPIOPinRead(GPIO\_PORTF\_BASE,GPIO\_PIN\_4)==0x00)

{

ui8Adjust--;

**if** (ui8Adjust < 30)

{

ui8Adjust = 30;

}

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_0, ui8Adjust \* ui32Load / 1000);

}

**if**(ROM\_GPIOPinRead(GPIO\_PORTF\_BASE,GPIO\_PIN\_0)==0x00)

{

ui8Adjust++;

**if** (ui8Adjust > 125)

{

ui8Adjust = 125;

}

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_0, ui8Adjust \* ui32Load / 1000);

}

ROM\_SysCtlDelay(100000);

}

}

**------------------------------------------------------------------------------------**

**Task 02:**

Youtube Link: https://www.youtube.com/watch?v=Wt\_7OTfV0Hc

**Modified Code:**

**#include** <stdint.h>

**#include** <stdbool.h>

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

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

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

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

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

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

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

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

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

**#define** PWM\_FREQUENCY 55

// Constants for min & max LED brightness

**#define** MIN\_BRIGHT 100

**#define** MAX\_BRIGHT 900

// Constant for LED delay

**#define** LED\_DELAY ROM\_SysCtlDelay(20000)

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

{

**volatile** uint32\_t ui32Load; // PWM count to be placed in Load register

**volatile** uint32\_t ui32PWMClock; // PWM clock (CPU\_Clock / 64)

**volatile** uint16\_t red; // Red LED brightness

red = MIN\_BRIGHT; // set brightness to minimum brightness

/\* Clock SETUP: CPU (40 MHz) & PWM (40 MHz / 64 = 625 KHz) \*/

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

ROM\_SysCtlPWMClockSet(SYSCTL\_PWMDIV\_64);

/\* PWM SETUP \*/

// enable PWM1 & Port F peripherals

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_PWM1);

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

// set PF1 as a PWM output

ROM\_GPIOPinTypePWM(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3);

ROM\_GPIOPinConfigure(GPIO\_PF1\_M1PWM5);

// set PWMs as down-counters & load count value based on desired PWM frequency

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

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

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

**PWMGenConfigure**(PWM1\_BASE, PWM\_GEN\_3, PWM\_GEN\_MODE\_DOWN);

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

**PWMGenPeriodSet**(PWM1\_BASE, PWM\_GEN\_3, ui32Load);

// set pulse width based on adjust & load value AND enable PWMs as output and to run

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, red \* ui32Load / 1000);

ROM\_PWMOutputState(PWM1\_BASE, PWM\_OUT\_5\_BIT|PWM\_OUT\_6\_BIT|PWM\_OUT\_7\_BIT, true);

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

ROM\_PWMGenEnable(PWM1\_BASE, PWM\_GEN\_3);

// cycle through red led with varying brightness

**while**(1){

**for**(; red <= MAX\_BRIGHT; red++){

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, red \* ui32Load / 1000);

LED\_DELAY;

}

**for**(; red >= MIN\_BRIGHT; red--){

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, red \* ui32Load / 1000);

LED\_DELAY;

}

}

}

**------------------------------------------------------------------------------------**

**Task 03:**

Youtube Link: https://www.youtube.com/watch?v=lyu6gWvb7MQ

**Modified Code:**

**#include** <stdint.h>

**#include** <stdbool.h>

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

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

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

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

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

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

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

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

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

**#define** PWM\_FREQUENCY 55

// Constants for min & max LED brightness

**#define** MIN\_BRIGHT 100

**#define** MAX\_BRIGHT 900

// Constant for LED delay

**#define** LED\_DELAY ROM\_SysCtlDelay(10000)

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

{

**volatile** uint32\_t ui32Load; // PWM count to be placed in Load register

**volatile** uint32\_t ui32PWMClock; // PWM clock (CPU\_Clock / 64)

**volatile** uint16\_t red, blue, green; // RGB LED brightness

red = blue = green = MIN\_BRIGHT; // set brightness to minimum brightness

/\* Clock SETUP: CPU (40 MHz) & PWM (40 MHz / 64 = 625 KHz) \*/

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

ROM\_SysCtlPWMClockSet(SYSCTL\_PWMDIV\_64);

/\* PWM SETUP \*/

// enable PWM1 & Port F peripherals

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_PWM1);

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

// set PF1, PF2, PF3 as a PWM output

ROM\_GPIOPinTypePWM(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3);

ROM\_GPIOPinConfigure(GPIO\_PF1\_M1PWM5);

ROM\_GPIOPinConfigure(GPIO\_PF2\_M1PWM6);

ROM\_GPIOPinConfigure(GPIO\_PF3\_M1PWM7);

// set PWMs as down-counters & load count value based on desired PWM frequency

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

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

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

PWMGenConfigure(PWM1\_BASE, PWM\_GEN\_3, PWM\_GEN\_MODE\_DOWN);

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

PWMGenPeriodSet(PWM1\_BASE, PWM\_GEN\_3, ui32Load);

// set pulse width based on adjust & load value AND enable PWMs as output and to run

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, red \* ui32Load / 1000);

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_6, blue \* ui32Load / 1000);

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_7, green \* ui32Load / 1000);

ROM\_PWMOutputState(PWM1\_BASE, PWM\_OUT\_5\_BIT|PWM\_OUT\_6\_BIT|PWM\_OUT\_7\_BIT, true);

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

ROM\_PWMGenEnable(PWM1\_BASE, PWM\_GEN\_3);

// cycle through colors with varying brightness

**while**(1){

**for**(; red <= MAX\_BRIGHT; red++){

**for**(; blue <= MAX\_BRIGHT; blue++){

**for**(; green <= MAX\_BRIGHT; green++){

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_7, green \* ui32Load / 1000);

LED\_DELAY;

}

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_6, blue \* ui32Load / 1000);

LED\_DELAY;

}

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, red \* ui32Load / 1000);

LED\_DELAY;

}

**for**(; red >= MIN\_BRIGHT; red--){

**for**(; blue >= MIN\_BRIGHT && red == MAX\_BRIGHT; blue--){

**for**(; green >= MIN\_BRIGHT && blue == MAX\_BRIGHT && red == MAX\_BRIGHT; green--){

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_7, green \* ui32Load / 1000);

LED\_DELAY;

}

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_6, blue \* ui32Load / 1000);

LED\_DELAY;

}

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, red \* ui32Load / 1000);

LED\_DELAY;

}

}

}