**Date Submitted:**

**Task 00: Execute provided code**

**Youtube Link:**

No Submission is required for this task

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

**Task 01:**

Youtube Link: <https://youtu.be/aUtM8CyCGqU>

**Modified Schematic (if applicable): N/A**

**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 52 //frequency to control the servo 18.2ms period

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

{

**volatile** uint32\_t ui32Load;

**volatile** uint32\_t ui32PWMClock;

**volatile** uint8\_t ui8Adjust;//allows us to adjust the position of the servo

ui8Adjust = 79;//center position to create a 1.5 mS pulse from the PWM

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) //Checking if SW1 is pressed

{

ui8Adjust = ui8Adjust + 6; //increment the Adjust variable to turn the servo 0-180 degrees

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

}

ROM\_SysCtlDelay(100000);

}

}

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

**Task 02:**

Youtube Link:

**Modified Schematic (if applicable): N/A**

**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 52 //frequency to control the servo 18.2ms period

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

{

/\* volatile uint32\_t ui32Load;

volatile uint32\_t ui32PWMClock;

volatile uint8\_t ui8Adjust;//allows us to adjust the position of the servo

ui8Adjust = 79;//center position to create a 1.5 mS pulse from the PWM

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);

\*/

**int** brightLED;

SysCtlPWMClockset(SYSCTL\_PWMDIV\_1);

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

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_PWM1);

**GPIOPinTypePWM**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1);

GPIOinConfigure(GPIO\_PF1\_M1PWM5);

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;

//Set Pulse width to 300

//Set the Generator Periods to 500

**PWMPulseWidthSet**(PWM1\_BASE, PMW\_OUT\_5, 300);

**PWMGenPeriodSet**(PWM1\_BASE, PWM\_GEN\_2, 500);

**PWMGenPeriodSet**(PWM1\_BASE, PWM\_GEN\_3, 500);

/\*

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);

\*/

**PWMOutputState**(PWM1\_BASE, PWM\_OUT\_5\_BIT , true);

**while**(1)

{

**for**(brightLED = 50; brightLED < 450; brightLED++) //starting from 10% then go up to 90% of PWM period

{

**PWMPulseWidthSet**(PWM1\_BASE, PM\_OUT\_5, brightLED);

ROM\_SysCtlDelay(100000);

}

**for**(brightLED = 450; brightLED > 50; brightLED--) //starting from 90% then go down to 10% of PWM period

{

**PWMPulseWidthSet**(PWM1\_BASE, PM\_OUT\_5, brightLED);

ROM\_SysCtlDelay(100000);

}

}

}

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

**Task 03:**

Youtube Link:

**Modified Schematic (if applicable):**

**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 52 //frequency to control the servo 18.2ms period

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

{

/\* volatile uint32\_t ui32Load;

volatile uint32\_t ui32PWMClock;

volatile uint8\_t ui8Adjust;//allows us to adjust the position of the servo

ui8Adjust = 79;//center position to create a 1.5 mS pulse from the PWM

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);

\*/

**int** brightLED;

SysCtlPWMClockset(SYSCTL\_PWMDIV\_1);

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

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_PWM1);

**GPIOPinTypePWM**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1);

GPIOinConfigure(GPIO\_PF1\_M1PWM5);

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;

//Set Pulse width to 300

//Set the Generator Periods to 500

**PWMPulseWidthSet**(PWM1\_BASE, PMW\_OUT\_5, 300);

**PWMGenPeriodSet**(PWM1\_BASE, PWM\_GEN\_2, 500);

**PWMGenPeriodSet**(PWM1\_BASE, PWM\_GEN\_3, 500);

/\*

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);

\*/

**PWMOutputState**(PWM1\_BASE, PWM\_OUT\_5\_BIT , true);

**while**(1)

{

**for**(brightLED = 50; brightLED < 450; brightLED++) //starting from 10% then go up to 90% of PWM period

{

**PWMPulseWidthSet**(PWM1\_BASE, PM\_OUT\_5, brightLED);

ROM\_SysCtlDelay(100000);

}

**for**(brightLED = 450; brightLED > 50; brightLED--) //starting from 90% then go down to 10% of PWM period

{

**PWMPulseWidthSet**(PWM1\_BASE, PM\_OUT\_5, brightLED);

ROM\_SysCtlDelay(100000);

}

}

}

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