**10/09/2018**

**Task 00: Execute provided code**

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

Modifications highlighted in Yellow

**Task 01:**

Youtube Link: https: https://youtu.be/XQ\_CwMC-BXc

**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

**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 < 33) // 90 degree right turn

{

ui8Adjust = 33; //56

}

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

}

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

{

ui8Adjust++;

**if** (ui8Adjust > 130) //90 degree left turn

{

ui8Adjust = 130;

}

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

}

ROM\_SysCtlDelay(100000);

}

}

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

**Task 02:**

Youtube Link: https://youtu.be/FNctQdqKf5s

**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

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

{

**volatile** uint32\_t ui32Load;

**volatile** uint32\_t ui32PWMClock;

**volatile** uint32\_t ui8Adjust; // changed type from uint8\_t to uint32\_t due to truncation

ui8Adjust = 500;

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\_GPIOF);

ROM\_GPIOPinTypePWM(GPIO\_PORTF\_BASE, GPIO\_PIN\_1); // Setting pin PF1 to PWM

ROM\_GPIOPinConfigure(GPIO\_PF1\_M1PWM5); // PF1 pin maps with PWM module 1 -> Gen2 -> PWM5

// For switch 1 unlock

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);

// setting pwm load for duty cycle configuration

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

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

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

// setting period for generator 2 on PWM module 1

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

// Setting default duty cycle to 50%

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, ui8Adjust \* ui32Load / 1000);

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

// Enables Generator 2 which connects to pwm5

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

**while**(1)

{

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

{

ui8Adjust--;

**if** (ui8Adjust < 100)

{

ui8Adjust = 100; //10 percent high

}

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, ui8Adjust \* ui32Load / 1000);

}

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

{

ui8Adjust++;

**if** (ui8Adjust > 900)

{

ui8Adjust = 900; //90 percent high

}

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, ui8Adjust \* ui32Load / 1000);

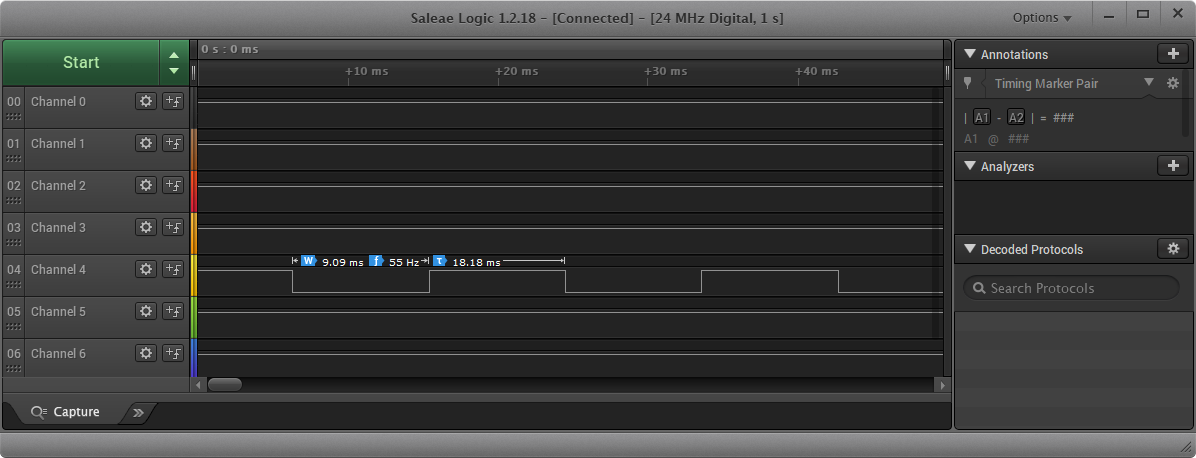
}

ROM\_SysCtlDelay(100000);

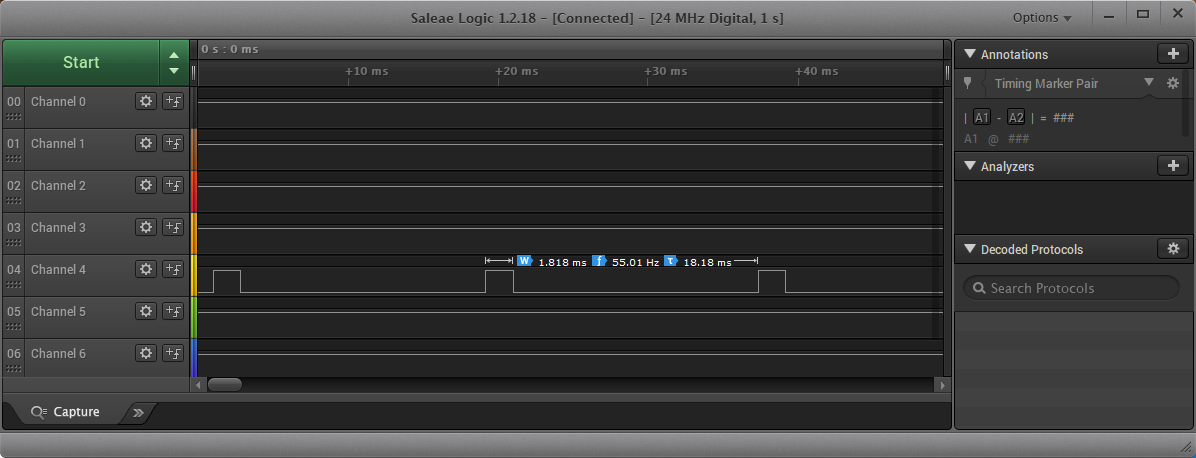
}

}

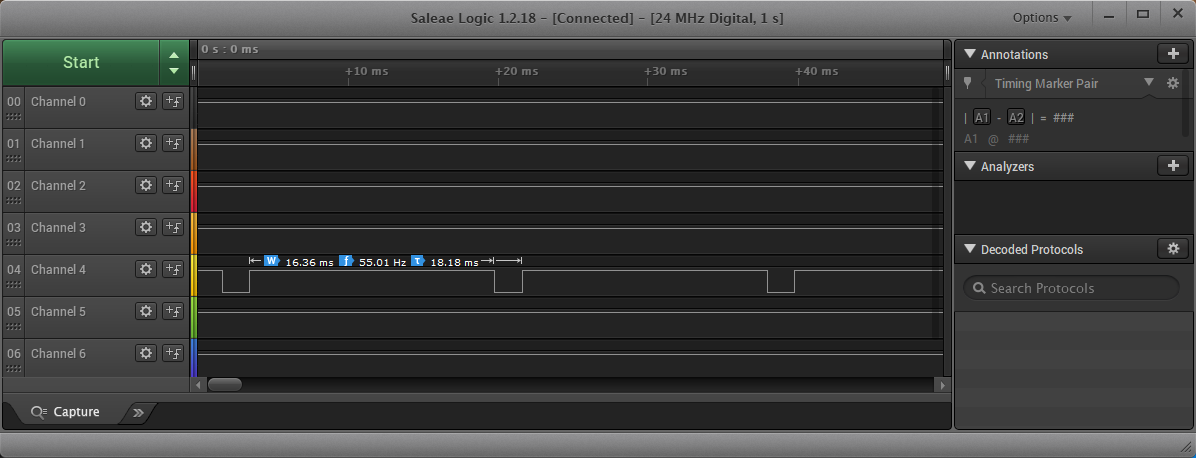
Default duty cycle of 50%, 9.09 ms high and low, period 18.18 ms, freq of 55Hz



Duty cycle of 10%, 1.89 ms high, period 18.18 ms, freq of 55Hz



Duty cycle of 90%, 16.36ms high, period 18.18 ms, freq of 55Hz



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

**Task 03:**

Youtube Link: https://youtu.be/z1chUYLy9sA

**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

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

{

**volatile** uint32\_t ui32Load;

**volatile** uint32\_t ui32PWMClock;

**volatile** uint32\_t ui8AdjustRed; // Adjust variable for Red LED

**volatile** uint32\_t ui8AdjustBlue; // Adjust variable for Blue LED

**volatile** uint32\_t ui8AdjustGreen; // Adjust variable for Blue LED

// Default 10% duty cycle

ui8AdjustRed = 100;

ui8AdjustBlue = 100;

ui8AdjustGreen = 100;

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\_GPIOF);

// Setting pins PF1, PF2 and PF3 to PWM

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

// Configuring pins for module 1 pwm outputs 5,6 and 7

ROM\_GPIOPinConfigure(GPIO\_PF1\_M1PWM5);

ROM\_GPIOPinConfigure(GPIO\_PF2\_M1PWM6);

ROM\_GPIOPinConfigure(GPIO\_PF3\_M1PWM7);

// For switch 1 unlock

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);

// setting pwm load for duty cycle configuration

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

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

// Configuring PWM generator 2 for pwm5 and generator 3 for pwm6 and pwm7

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

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

// setting period for generators 2 and 3 on PWM module 1

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

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

// Setting default duty cycles

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, ui8AdjustRed \* ui32Load / 1000);

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_6, ui8AdjustBlue \* ui32Load / 1000);

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_7, ui8AdjustGreen \* ui32Load / 1000);

// Enabling PWM outputs 5,6 and 7

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

// Enables Generator 2 which connects to pwm5

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

// Enables Generator 3 which connects to pwm6 and pwm7

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

**while**(1)

{

// Nested loops driving all 3 LEDs with 5% duty cycle increments

**for**(ui8AdjustRed=100; ui8AdjustRed<900; ui8AdjustRed++)

{

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, ui8AdjustRed \* ui32Load / 1000);

ROM\_SysCtlDelay(100000);

**for**(ui8AdjustBlue=100; ui8AdjustBlue<=10; ui8AdjustBlue++)

{

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_6, ui8AdjustBlue \* ui32Load / 1000);

ROM\_SysCtlDelay(100000);

**for**(ui8AdjustGreen=100; ui8AdjustGreen<=900; ui8AdjustGreen++)

{

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_7, ui8AdjustGreen \* ui32Load / 1000);

ROM\_SysCtlDelay(100000);

ui8AdjustGreen = (ui8AdjustGreen+50)-1;

}

ui8AdjustBlue= (ui8AdjustBlue+50)-1;

}

ui8AdjustRed= (ui8AdjustRed+50)-1;

}

**for**(ui8AdjustGreen=100; ui8AdjustGreen<900; ui8AdjustGreen++)

{

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_7, ui8AdjustGreen \* ui32Load / 1000);

ROM\_SysCtlDelay(10000);

**for**(ui8AdjustBlue=100; ui8AdjustBlue<900; ui8AdjustBlue++)

{

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_6, ui8AdjustBlue \* ui32Load / 1000);

ROM\_SysCtlDelay(10000);

**for**(ui8AdjustRed=100; ui8AdjustRed<900; ui8AdjustRed++)

{

ROM\_PWMPulseWidthSet(PWM1\_BASE, PWM\_OUT\_5, ui8AdjustRed \* ui32Load / 1000);

ROM\_SysCtlDelay(10000);

ui8AdjustRed= (ui8AdjustRed+50)-1;

}

ui8AdjustBlue= (ui8AdjustBlue+50)-1;

}

ui8AdjustGreen = (ui8AdjustGreen+50)-1;

}

}

}