Adrian Ruiz

CPE 403

Lab 6

Task 00:

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**;** //adjust the position of the servo

ui8Adjust **=** 83**;** //center position of servo

//Run CPU at 40MHz

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

//Make the divder range 2 to 64

ROM\_SysCtlPWMClockSet**(**SYSCTL\_PWMDIV\_64**);**

//Enable PWM1 and GPIOD modules for PWM and GPIOF modules

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_PWM1);

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOD);

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF);

//Configure PD0 as a PWM output(M1PW0)

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

ROM\_GPIOPinConfigure(GPIO\_PD0\_M1PWM0);

//Set S1 and S2 on lauchpad to pull up buttons.

//We need to GPIO Commit Control register to make the change since S1 is a NMI input

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);

//Set up the PWM clock and load Values into the PWM registers

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);

//Set Pulse Width, change the resolution for servo, and enable PWM

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)

{

//When S1 is pressed servo positon decreases

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

{

ui8Adjust--;

if (ui8Adjust < 56)

{

ui8Adjust = 56;

}

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

}

//When S2 is pressed, servo position increases

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

{

ui8Adjust++;

if (ui8Adjust > 111)

{

ui8Adjust = 111;

}

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

}

ROM\_SysCtlDelay(100000);

}

}

Task 01

Altered Code:

int i**;**

**while(**1**)**

**{**

//Two for loops control the direction of the Servo

**for(**i **=** 30**;** i **<** 150**;** **++**i**)**

**{**

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

ROM\_SysCtlDelay**(**100000**);**

**}**

**for(**i **=** 150**;** i **>** 30**;** **--**i**)**

**{**

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

ROM\_SysCtlDelay**(**100000**);**

**}**

ROM\_SysCtlDelay**(**200000**);**

**}**

**}**

The while has been changed. The buttons are still enabled but they don’t do anything. Now, the two loops make the servo go from roughly 0 to roughly 180 by incrementing and decrementing i.

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**;** //adjust the position of the servo

ui8Adjust **=** 30**;** //center position of servo

//Run CPU at 40MHz

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

//Make the divder range 2 to 64

ROM\_SysCtlPWMClockSet**(**SYSCTL\_PWMDIV\_64**);**

//Enable PWM1 and GPIOD modules for PWM and GPIOF modules

ROM\_SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_PWM1**);**

ROM\_SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_GPIOD**);**

ROM\_SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_GPIOF**);**

//Configure PD0 as a PWM output(M1PW0)

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

ROM\_GPIOPinConfigure**(**GPIO\_PD0\_M1PWM0**);**

//Set S1 and S2 on lauchpad to pull up buttons.

//We need to GPIO Commit Control register to make the change since S1 is a NMI input

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**);**

//Set up the PWM clock and load Values into the PWM registers

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**);**

//Set Pulse Width, change the resolution for servo, and enable PWM

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**);**

int i**;**

**while(**1**)**

**{**

//Two for loops control the direction of the Servo

**for(**i **=** 30**;** i **<** 150**;** **++**i**)**

**{**

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

ROM\_SysCtlDelay**(**100000**);**

**}**

**for(**i **=** 150**;** i **>** 30**;** **--**i**)**

**{**

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

ROM\_SysCtlDelay**(**100000**);**

**}**

ROM\_SysCtlDelay**(**200000**);**

**}**

**}**

Task 02:

Altered Code:

//Configure PD1 as a PWM output(M1PW5)

ROM\_GPIOPinTypePWM**(**GPIO\_PORTF\_BASE**,** GPIO\_PIN\_1**);**

ROM\_GPIOPinConfigure**(**GPIO\_PF1\_M1PWM5);

I configured PD1 as PWM. Also, PORTF (Red LED) is used for the PWM.

//Set up the PWM clock and load Values into the PWM registers

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**);**

//Set Pulse Width and enable PWM

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

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

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

The Mux was changed from GEN\_0 to GEN\_2 to enable the PWM for the LED.

//When S1 is pressed, brightness is decreased

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

**{**

ui8Adjust**--;**

**if** **(**ui8Adjust **<** 100**)**

**{**

ui8Adjust **=** 100**;**

**}**

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

**}**

//When S2 is pressed, brightness increases

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

**{**

ui8Adjust**++;**

**if** **(**ui8Adjust **>** 900**)**

**{**

ui8Adjust **=** 900**;**

**}**

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

**}**

Inside the while loop, the button presses change the brightness of the LED. One switch lowers the brightness while the other increases it. The brightness also cannot go pass a certain min and max.

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**;** //adjust the position of the servo

ui8Adjust **=** 83**;** //center position of servo

//Run CPU at 40MHz

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

//Make the divder range 2 to 64

ROM\_SysCtlPWMClockSet**(**SYSCTL\_PWMDIV\_64**);**

//Enable PWM1 and GPIOD modules for PWM and GPIOF modules

ROM\_SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_PWM1**);**

ROM\_SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_GPIOD**);**

//Enable PORTF modules

ROM\_SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_GPIOF**);**

//Configure PD1 as a PWM output(M1PW5)

ROM\_GPIOPinTypePWM**(**GPIO\_PORTF\_BASE**,** GPIO\_PIN\_1**);**

ROM\_GPIOPinConfigure**(**GPIO\_PF1\_M1PWM5**);**

//Set S1 and S2 on lauchpad to pull up buttons.

//We need to GPIO Commit Control register to make the change since S1 is a NMI input

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**);**

//Set up the PWM clock and load Values into the PWM registers

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**);**

//Set Pulse Width and enable PWM

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

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

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

**while(**1**)**

**{**

//When S1 is pressed, brightness is decreased

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

**{**

ui8Adjust**--;**

**if** **(**ui8Adjust **<** 100**)**

**{**

ui8Adjust **=** 100**;**

**}**

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

**}**

//When S2 is pressed, brightness increases

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

**{**

ui8Adjust**++;**

**if** **(**ui8Adjust **>** 900**)**

**{**

ui8Adjust **=** 900**;**

**}**

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

**}**

ROM\_SysCtlDelay**(**100000**);**

**}**

**}**

Task 03:

Altered Code:

//Configure PD1 as a PWM output(M1PW5)

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

//Configure the mux that selects each LED

ROM\_GPIOPinConfigure**(**GPIO\_PF1\_M1PWM5**);**

ROM\_GPIOPinConfigure**(**GPIO\_PF2\_M1PWM6**);**

ROM\_GPIOPinConfigure**(**GPIO\_PF3\_M1PWM7**);**

Each LED needs to be selected by the Mux, and needs to be configured as a PWM output.

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 and enable PWM

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

ROM\_PWMPulseWidthSet**(**PWM1\_BASE**,** PWM\_OUT\_6**,** ui8Adjust **\*** ui32Load **/** 1000**);**

ROM\_PWMPulseWidthSet**(**PWM1\_BASE**,** PWM\_OUT\_7**,** ui8Adjust **\*** 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**);**

Similar to task02, but each LED needs to have its Pulse set and put into an output state. Also another mux is used.

int r **=** 100**;**

int g **=** 100**;**

int b **=** 100**;**

**for(**r**;** r **<** 900**;** r**++)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**for(**b**;** b **<** 900**;** b**++)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**for(**g**;** g **<** 900**;** g**++)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**}**

**}**

**}**

**for(**g**;** g **>=** 100**;** g**--)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**}**

**for(**b**;** b **>=** 100**;** b**--)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**}**

**for(**r**;** r **>=** 100**;** r**--)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**}**

**}**

Inside the while loop, the LEDs brightness is changed. It increases the brightness of green then blue then red. Then it decreases the brightness of green then blue then red. This causes different colors to be shown.

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**;** //adjust the position of the servo

ui8Adjust **=** 83**;** //center position of servo

//Run CPU at 40MHz

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

//Make the divder range 2 to 64

ROM\_SysCtlPWMClockSet**(**SYSCTL\_PWMDIV\_64**);**

//Enable PWM1 and GPIOD modules for PWM and GPIOF modules

ROM\_SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_PWM1**);**

//Enable PORTF modules

ROM\_SysCtlPeripheralEnable**(**SYSCTL\_PERIPH\_GPIOF**);**

//Configure PD1 as a PWM output(M1PW5)

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

//Configure the mux that selects each LED

ROM\_GPIOPinConfigure**(**GPIO\_PF1\_M1PWM5**);**

ROM\_GPIOPinConfigure**(**GPIO\_PF2\_M1PWM6**);**

ROM\_GPIOPinConfigure**(**GPIO\_PF3\_M1PWM7**);**

//Set up the PWM clock and load Values into the PWM registers

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 and enable PWM

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

ROM\_PWMPulseWidthSet**(**PWM1\_BASE**,** PWM\_OUT\_6**,** ui8Adjust **\*** ui32Load **/** 1000**);**

ROM\_PWMPulseWidthSet**(**PWM1\_BASE**,** PWM\_OUT\_7**,** ui8Adjust **\*** 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**);**

**while(**1**)**

**{**

int r **=** 100**;**

int g **=** 100**;**

int b **=** 100**;**

**for(**r**;** r **<** 900**;** r**++)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**for(**b**;** b **<** 900**;** b**++)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**for(**g**;** g **<** 900**;** g**++)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**}**

**}**

**}**

**for(**g**;** g **>=** 100**;** g**--)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**}**

**for(**b**;** b **>=** 100**;** b**--)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**}**

**for(**r**;** r **>=** 100**;** r**--)**

**{**

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

ROM\_SysCtlDelay**(**10000**);**

**}**

**}**

**}**