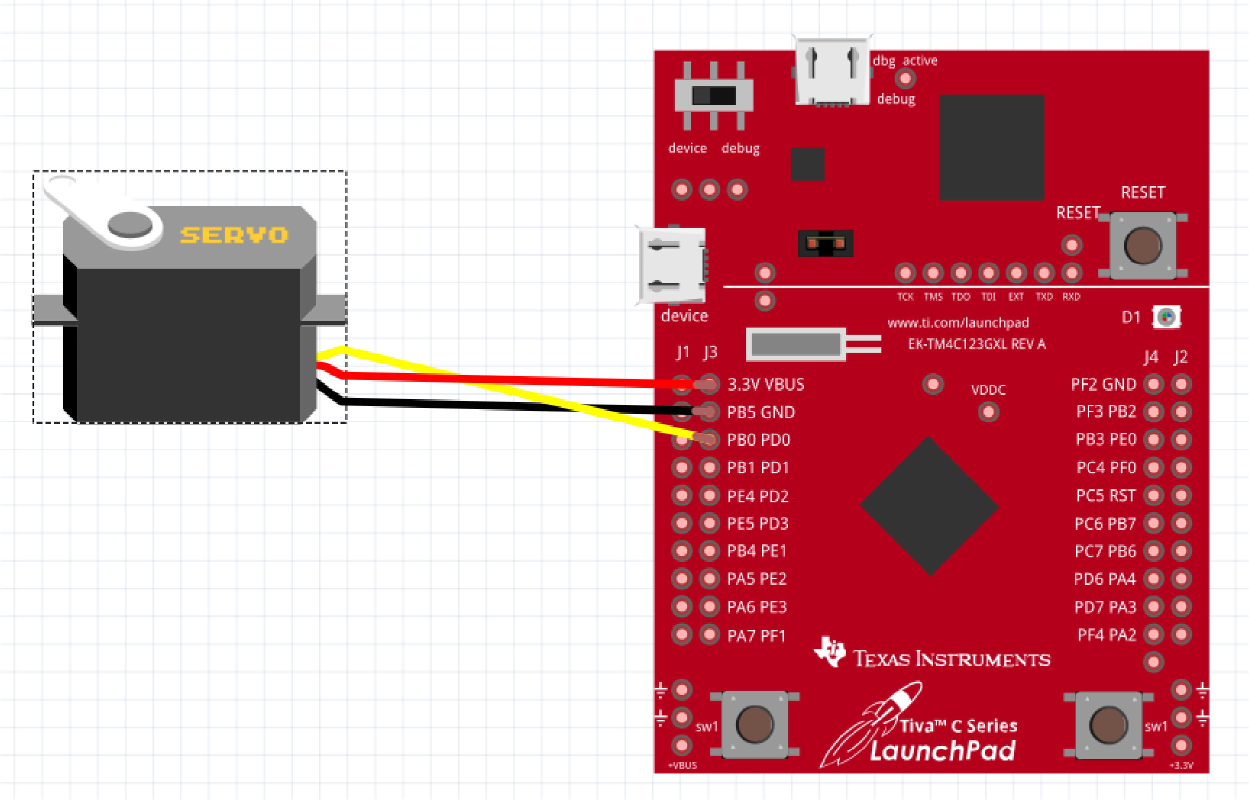
Nathan Hanuscin
Github root directory: https://github.com/nhanuscin/HappyFunStuff

Date Submitted: **10/6/19**

Task 01:

Youtube Link: https://www.youtube.com/watch?v=g5QV3AMOpTw

Modified Schematic (if applicable):



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
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

```
int main(void)
{
    volatile uint32_t ui32Load;
    volatile uint32_t ui32PWMClock;
    volatile uint32_t ui8Adjust;
    ui8Adjust = 83;

    //Set clock to 40MHz

ROM_SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);
    //Set PWM Clock to 625KHz
    ROM_SysCtlPWMClockSet(SYSCTL_PWMDIV_64);

    //Enable PWM1, PortD and PortF
    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_PWM1);
    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOD);
    ROM_SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);

    //Enable PD0 as output for PWM
    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);

    //Set PWM Counter and load value
    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 the initial pulse width 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)
    {
       //turn clockwise
       while((ui8Adjust < 120) & (ui8Adjust > 48))
       {
           ui8Adjust++;
           ROM_PWMPulseWidthSet(PWM1_BASE, PWM_OUT_0, ui8Adjust * ui32Load / 1000);
           ROM_SysCtlDelay(200000);
       }
       ui8Adjust = 119;
       //turn counterclockwise
       while((ui8Adjust < 120) & (ui8Adjust > 48))
       {
           ui8Adjust--;
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

```
           ROM_PWMPulseWidthSet(PWM1_BASE, PWM_OUT_0, ui8Adjust * ui32Load / 1000);
           ROM_SysCtlDelay(200000);
        }
       ui8Adjust = 49;
    }

}
-----------------------------------------------------------------------------------
```

Task 02:

Youtube Link: https://www.youtube.com/watch?v=DHB9-wKv-28

Modified Schematic (if applicable): N/A

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"
#include "driverlib/timer.h"


 //about 2ms at 40Mhz
#define time 28333

//PWM frequency in hz
uint32_t freq = 100000;

int main()
{

  //Set clock to 40Mhz
  SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);

  uint32_t i;
  uint32_t Period;
  uint32_t dutyCycle;
  Period = SysCtlClockGet()/freq ;  //set period to 400
  dutyCycle = Period-2;

  //Configure PF1 as T0CCP1
  SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
  SysCtlDelay(3);
  GPIOPinConfigure(GPIO_PF1_T0CCP1);
  GPIOPinTypeTimer(GPIO_PORTF_BASE, GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

```
  //Configure timer 0 to split pair and timer B in PWM mode
  //Set period and starting duty cycle.
  SysCtlPeripheralEnable(SYSCTL_PERIPH_TIMER0);
  SysCtlDelay(3);
  TimerConfigure(TIMER0_BASE, TIMER_CFG_SPLIT_PAIR|TIMER_CFG_B_PWM);
  TimerLoadSet(TIMER0_BASE, TIMER_B, Period -1);
  TimerMatchSet(TIMER0_BASE, TIMER_B, dutyCycle); // PWM

  //Turn on timer0B
  TimerEnable(TIMER0_BASE, TIMER_B);

  //Start by rising Red LED
  for(i=Period-2; i > 40 ;i--){
    TimerMatchSet(TIMER0_BASE, TIMER_B, i);
    SysCtlDelay(time);
  }
  while(1)
  {
    //Red brightness goes down
    for(i=1; i < 360; i++){
      TimerMatchSet(TIMER0_BASE, TIMER_B, i);
      SysCtlDelay(time);
    }
    //Red brightness goes up
    for(i=Period-2; i > 40; i--){
      TimerMatchSet(TIMER0_BASE, TIMER_B, i);
      SysCtlDelay(time);
    }

  }

}
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.

Nathan Hanuscin
Github root directory: https://github.com/nhanuscin/HappyFunStuff

Waiting for DC motor for tasks 3 and 4

Task 03:

Youtube Link:

Modified Schematic (if applicable):

Modified Code:

```
// Insert code here
------------------------------------------------------------------------------------
```

Task 04:

Youtube Link:

Modified Schematic (if applicable):

Modified Code:

```
// Insert code here
------------------------------------------------------------------------------------
```

Grading scheme: 30% Coding, 30% Documentation, 40% Execution/Video.