**Date Submitted: 10/01/2018**

**Assignment Youtube Playlist:**

**https://www.youtube.com/playlist?list=PL4oTyvRrubXerSPkBeJqt99R3o75mPwYp**

**Task 00: Execute the provided code, no submission is required.**

**Youtube Link:** **https://youtu.be/HbGd-C6YSVY**

**Original Code (added comments):**

**#include** <stdint.h>

**#include** <stdbool.h>

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

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

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

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

// All of our included files are above, including our linked driver libraries.

uint8\_t ui8PinData=2; //Setting the first blink variable as 2, or 0010 (red)

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

{

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

//Main oscillator, external 16MHz source, 400MHz PLL enabled, dividing by 5, total frequency 400MHz/(5\*2)=40MHz.

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOF);//Enabling GPIO Port F.

**GPIOPinTypeGPIOOutput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3);//Port F Pins 1,2,3 (LED pins) are output.

**while**(1)

{

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, ui8PinData);//Write variable data

**SysCtlDelay**(2000000);//Delay 2,000,000 (3 clock cycles each)

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, 0x00);//Write 0s to pins (all off)

**SysCtlDelay**(2000000);//Delay 2,000,000 (3 clock cycles each)

**if**(ui8PinData==8) {ui8PinData=2;} **else** {ui8PinData=ui8PinData\*2;}

//In effect, steps through pattern 2,4,8,2,4,8 for 0010,0100,1000,0010,0100,1000, or R,B,G,R,B,G.

}

}

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

**Task 01: Determine the current period and on-time of the LED blinking. Change the delay of the LED blink (approx. 0.425 sec) by changing the delay and clock source and configuration – determine the CLK frequency – verify the delay to be approx. 0.425 sec.**

For Task 01 I decided to use continue using the external oscillator and change the clock divider from 5 to 64, for a total division of 128. Using the 400MHz PLL, this gives us a clock frequency of 3.125 MHz. I then calculated the new delay required to be .

Youtube Link: https://youtu.be/KzVuByQhNB8

**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"

//All of our included files are above, including our linked driver libraries.

uint8\_t ui8PinData=2; //Setting the first blink variable as 2, or 0010 (red)

**int** clock\_speed = 0;//Sensing clock speed variable.

**int** delay = 442708;

//The total delay with our clock divider at 64 (\*2=128) and 3 clock cycles per delay will be:

//((1/3.125MHz)\*3) \* 442,708 = 424.99ms ~ 425ms.

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

{

**SysCtlClockSet**(SYSCTL\_SYSDIV\_64|SYSCTL\_USE\_PLL|SYSCTL\_XTAL\_16MHZ|SYSCTL\_OSC\_MAIN);

//Main oscillator, external 16MHz source, 400MHz PLL enabled, dividing by 64, total frequency 400MHz/(64\*2)=3.125MHz.

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOF);//Enabling GPIO Port F.

**GPIOPinTypeGPIOOutput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3);//Port F Pins 1,2,3 (LED pins) are output.

clock\_speed = **SysCtlClockGet**();//Sensing clock speed.

**while**(1)

{

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, ui8PinData);//Write variable data

**SysCtlDelay**(delay);//Delay 442,708 (3 clock cycles each)

**if**(ui8PinData==8) {ui8PinData=2;} **else** {ui8PinData=ui8PinData\*2;}

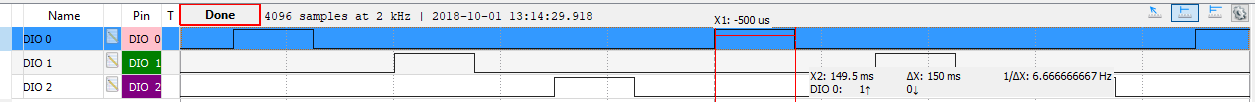
//In effect, steps through pattern 2,4,8,2,4,8 for 0010,0100,1000,0010,0100,1000, or R,B,G,R,B,G.

}

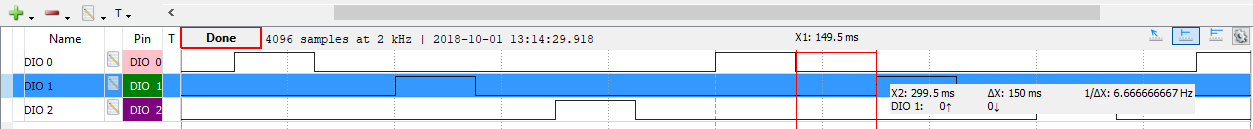
}

Using original code from Task 00:

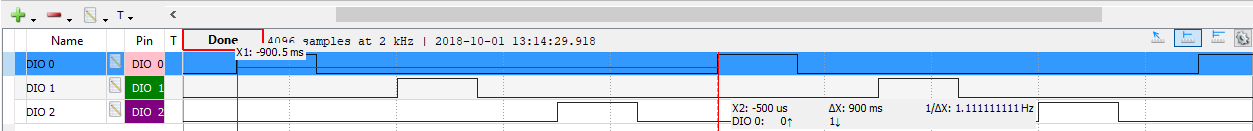
Original on-time: 150ms



Time between blinks: 150ms

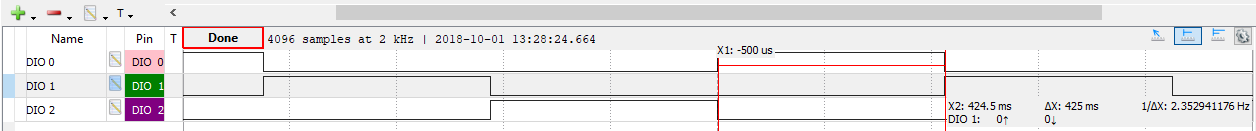


Current cycle period: 900ms



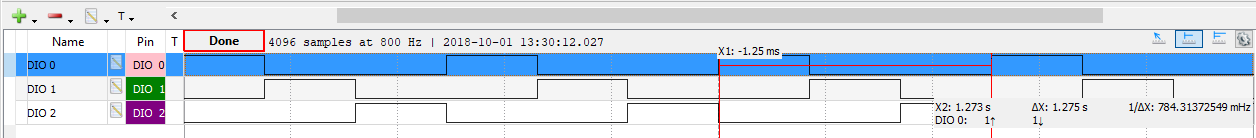
Using modified code for Task 01:

On-time: 425ms



No off-time per Dr. Venki’s instructions.

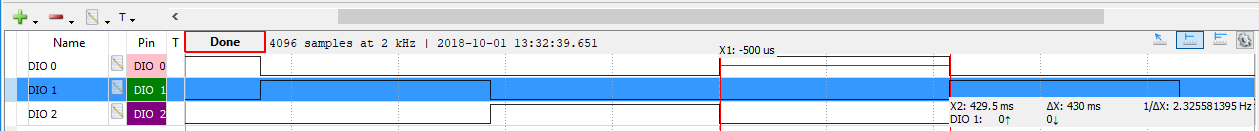
Current cycle period: 1.275s



Modified clock frequency determined to be: 3.125MHz.



Point of interest: If we were to use the internal oscillator of the same listed frequency rather than our internal crystal, we can see the reduction in accuracy as our delay moves from 425ms to 430ms:



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

**Task 02: Change the a) sequence of LED blinking (from RGB sequence to BGR), and b) blink one LED, two LED, and three LED at an instance and with a sequence (sequence of blinking with delay – R, G, B, RG, RB, GB, RGB, R, G, …).**

Per Dr Venki’s E-mail I have completed Task 02 in one part. This is accomplished by running two separate patterns two times. Per instructions the delay has been re-introduced into the pattern. The first pattern will be B\_G\_R\_B\_G\_R, the second pattern will be R\_G\_B\_RG\_RB\_GB\_RGB\_R\_G\_B\_RG\_RB\_GB\_RGB. These two patterns will be separated by an extended delay.

Youtube Link: https://youtu.be/RUl3XVaiTGI

**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"

// All of our included files are above, including our linked driver libraries.

uint8\_t ui8PinData=2; //Setting the first blink variable as 2, or 0010 (red)

**int** clock\_speed = 0;//Sensing clock speed.

**int** delay = 442708;

//The total delay with our clock divider at 64 (\*2=128) and 3 clock cycles per delay will be:

//((1/3.125MHz)\*3) \* 442,708 = 424.99ms ~ 425ms.

**const** uint8\_t R = 2;

**const** uint8\_t B = 4;

**const** uint8\_t G = 8;

//Setting values for the three LEDs for ease of use later.

uint8\_t i = 0; //Counting variable for LED color cycle.

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

{

uint8\_t LEDpatterns[24];

LEDpatterns[0] = B;

LEDpatterns[1] = G;

LEDpatterns[2] = R;

LEDpatterns[3] = B;

LEDpatterns[4] = G;

LEDpatterns[5] = R;

LEDpatterns[6] = 0;

LEDpatterns[7] = 0;

LEDpatterns[8] = R;

LEDpatterns[9] = G;

LEDpatterns[10] = B;

LEDpatterns[11] = R+G;

LEDpatterns[12] = R+B;

LEDpatterns[13] = G+B;

LEDpatterns[14] = R+G+B;

LEDpatterns[15] = R;

LEDpatterns[16] = G;

LEDpatterns[17] = B;

LEDpatterns[18] = R+G;

LEDpatterns[19] = R+B;

LEDpatterns[20] = G+B;

LEDpatterns[21] = R+G+B;

LEDpatterns[22] = 0;

LEDpatterns[23] = 0;

//The colors of LEDs can be obtained with simple addition.

//This method is less space-efficient but is easier for quick use.

**SysCtlClockSet**(SYSCTL\_SYSDIV\_64|SYSCTL\_USE\_PLL|SYSCTL\_XTAL\_16MHZ|SYSCTL\_OSC\_MAIN);

//Main oscillator, external 16MHz source, 400MHz PLL enabled, dividing by 64, total frequency 400MHz/(64\*2)=3.125MHz.

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOF);//Enabling GPIO Port F.

**GPIOPinTypeGPIOOutput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3);//Port F Pins 1,2,3 (LED pins) are output.

clock\_speed = **SysCtlClockGet**();

**while**(1)

{

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, LEDpatterns[i]);//Write variable data

**SysCtlDelay**(delay);//Delay 442,708 (3 clock cycles each)

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, 0x00);//Write 0s to pins (all off)

**SysCtlDelay**(delay);//Delay 442,708 (3 clock cycles each)

**if**(i == 23) {i=0;} **else** {i=i+1;}

//Step through each portion of the array to create the pattern above.

}

}

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