**10/04/2018**

**Task 00: Execute provided code**

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

**Task 01:**

Youtube Link: https://youtu.be/gTotu4k8hko

**Modified Code:**

uint32\_t ui32PeriodHigh;

uint32\_t ui32PeriodLow;

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

{

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

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

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

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_TIMER0);

**TimerConfigure**(TIMER0\_BASE, TIMER\_CFG\_PERIODIC);

ui32PeriodHigh = 3\* (**SysCtlClockGet**() /2) /4; // Period of 0.375s 2Hz

ui32PeriodLow = (**SysCtlClockGet**()/2) / 4; // Period of 0.125s 2Hz

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32PeriodHigh -1);

**IntEnable**(INT\_TIMER0A);

**TimerIntEnable**(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);

**IntMasterEnable**();

**TimerEnable**(TIMER0\_BASE, TIMER\_A);

**while**(1)

{

}

}

**void** **Timer0IntHandler**(**void**)

{

// Clear the timer interrupt

**TimerIntClear**(TIMER0\_BASE, TIMER\_TIMA\_TIMEOUT);

// Read the current state of the GPIO pin and

// write back the opposite state

**if**(**GPIOPinRead**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2))

{

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32PeriodLow -1); // loading Low

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, 0);

}

**else**

{

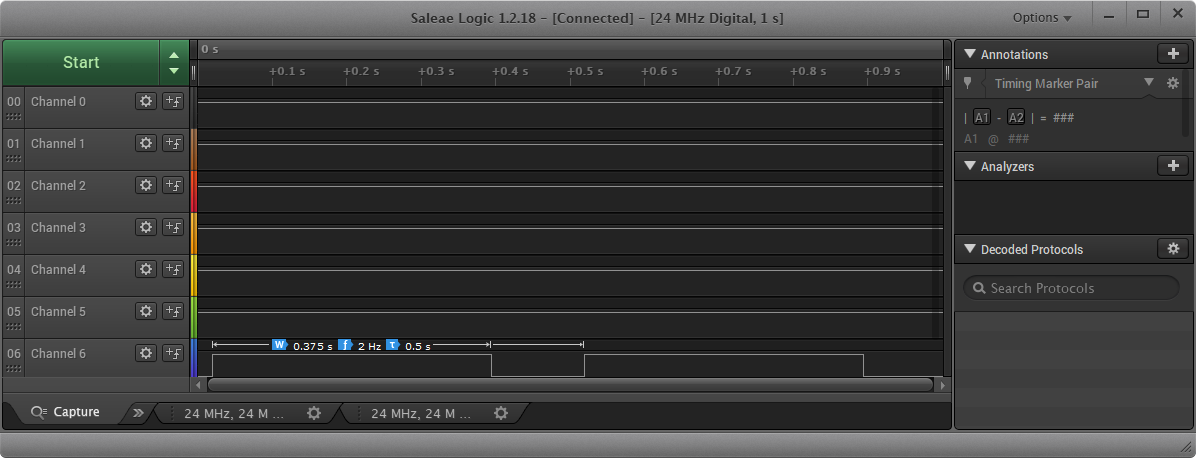
**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32PeriodHigh -1); //loading High

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 4);

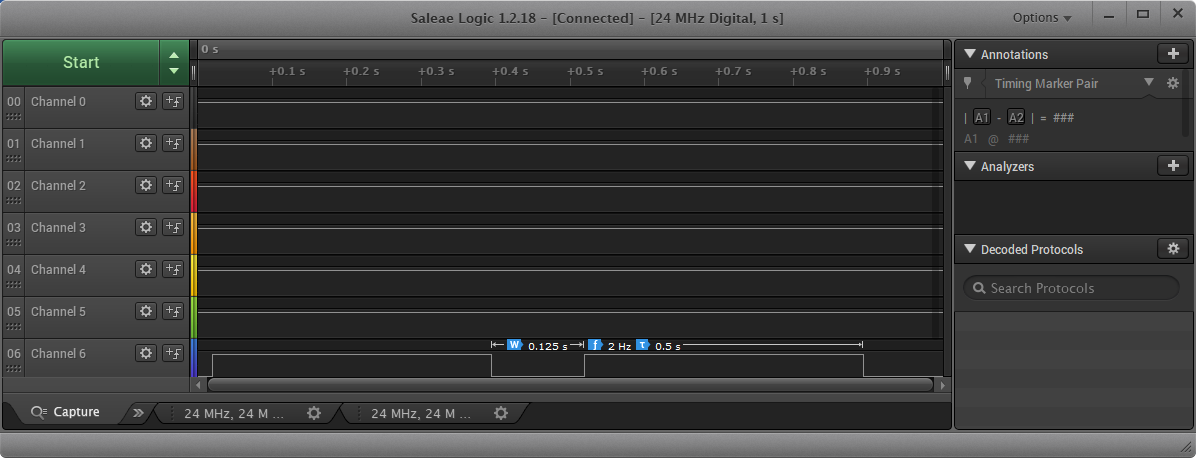
}

}

**Wave Form showing high signal of 0.375 seconds 2Hz 75% duty cycle**



**Wave Form showing low signal of 0.125 seconds 2Hz 75% duty cycle**



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

**Task 02:**

Youtube Link: <https://youtu.be/k8VaDwzBQ0o>

**Modified Code:**

**void** **timer1A\_delaymSec**( **int** ttime);

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

{

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

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

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_TIMER1);

**GPIOPinTypeGPIOInput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_4);

**GPIOPadConfigSet**(GPIO\_PORTF\_BASE,GPIO\_PIN\_4,GPIO\_STRENGTH\_2MA,GPIO\_PIN\_TYPE\_STD\_WPU);

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

**GPIOIntEnable**(GPIO\_PORTF\_BASE, GPIO\_INT\_PIN\_4);

**GPIOIntTypeSet**(GPIO\_PORTF\_BASE, GPIO\_INT\_PIN\_4, GPIO\_RISING\_EDGE);

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_TIMER1);

**IntEnable**(INT\_GPIOF);

**while**(1)

{

}

}

**void** **PortFPin4IntHandler**(**void**)

{

// Clear the GPIO interrupt

**GPIOIntClear**(GPIO\_PORTF\_BASE, GPIO\_INT\_PIN\_4);

// Read the current state of the GPIO pin and

// write back the opposite state

**GPIOPinWrite**(GPIO\_PORTF\_BASE,GPIO\_PIN\_2, GPIO\_PIN\_2); // turn on blue led

timer1A\_delaymSec( 1500 ); // 1.5 second delay

**GPIOPinWrite**(GPIO\_PORTF\_BASE,GPIO\_PIN\_2, 0); // turn off blue led

}

**void** **timer1A\_delaymSec**( **int** ttime)

{

**int** i;

SYSCTL\_RCGCTIMER\_R |=2;

TIMER1\_CTL\_R = 0;

TIMER1\_CFG\_R = 0x04;

TIMER1\_TAMR\_R = 0x02;

TIMER1\_TAILR\_R = 640 - 1;

TIMER1\_TAPR\_R = 275 - 1;

TIMER1\_ICR\_R = 0x1;

TIMER1\_CTL\_R |= 0x01;

**for**(i=0; i < ttime; i++)

{

**while** ((TIMER1\_RIS\_R & 0x1) == 0)

;

TIMER1\_ICR\_R = 0x1;

}

}

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