**Lab04**

**Date Submitted: 9/28/2019**

**Task 00: Execute provided code**

**Youtube Link: No submition required**

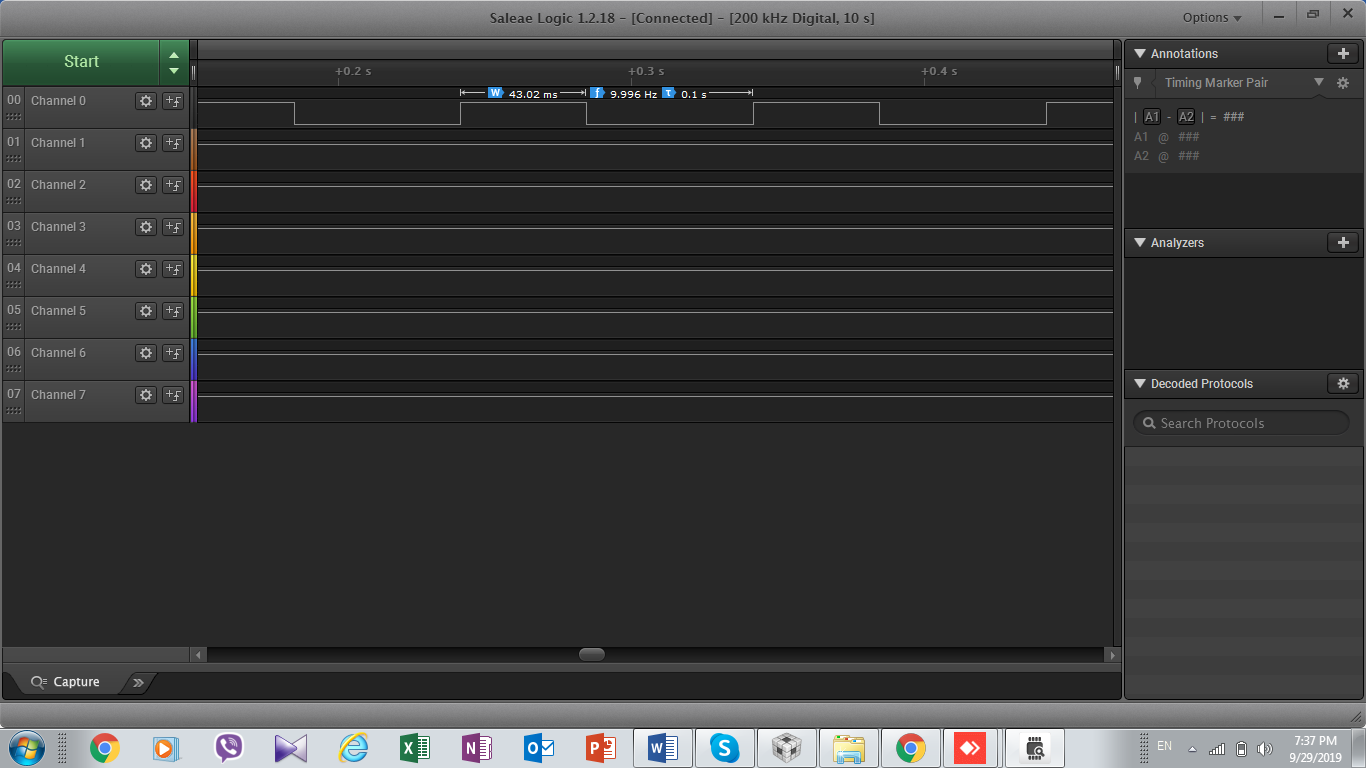
**------------------------------------------------------------------------------------**

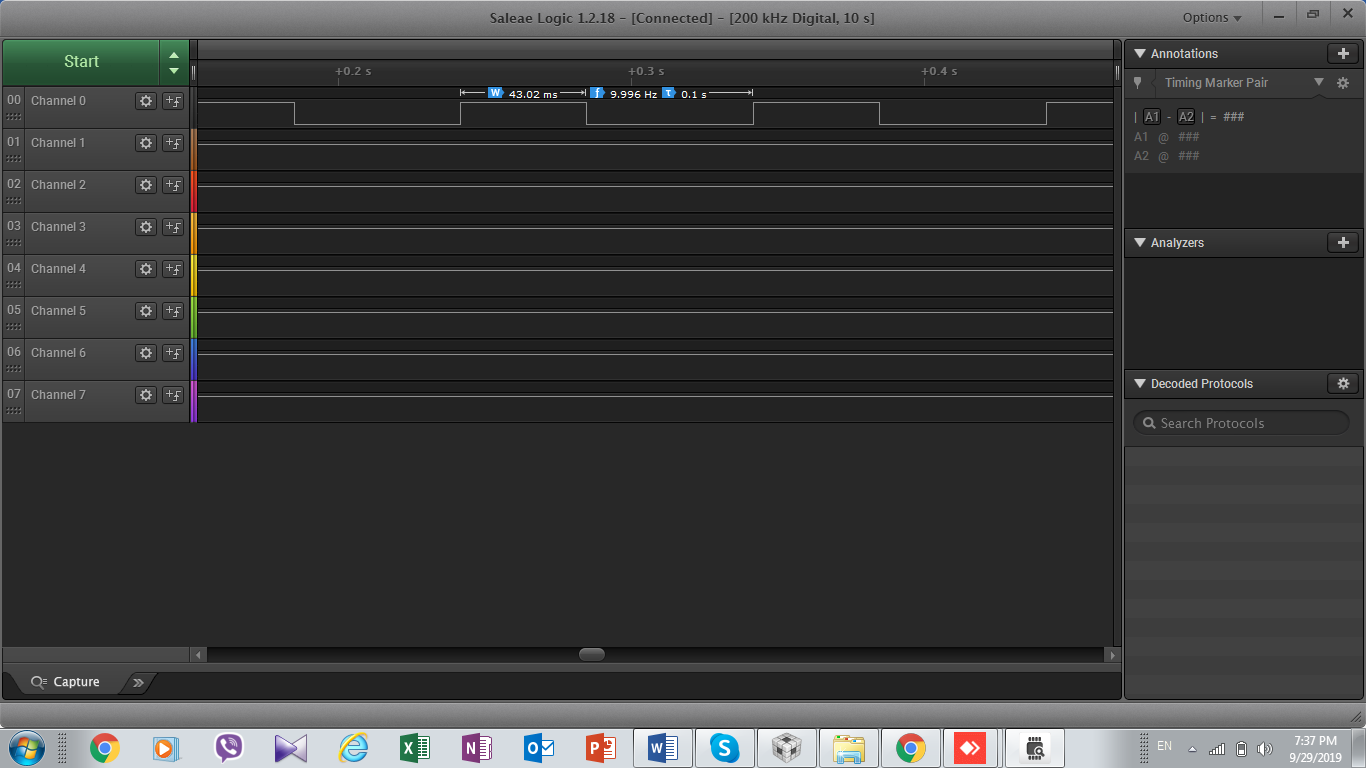
**Task 01:**

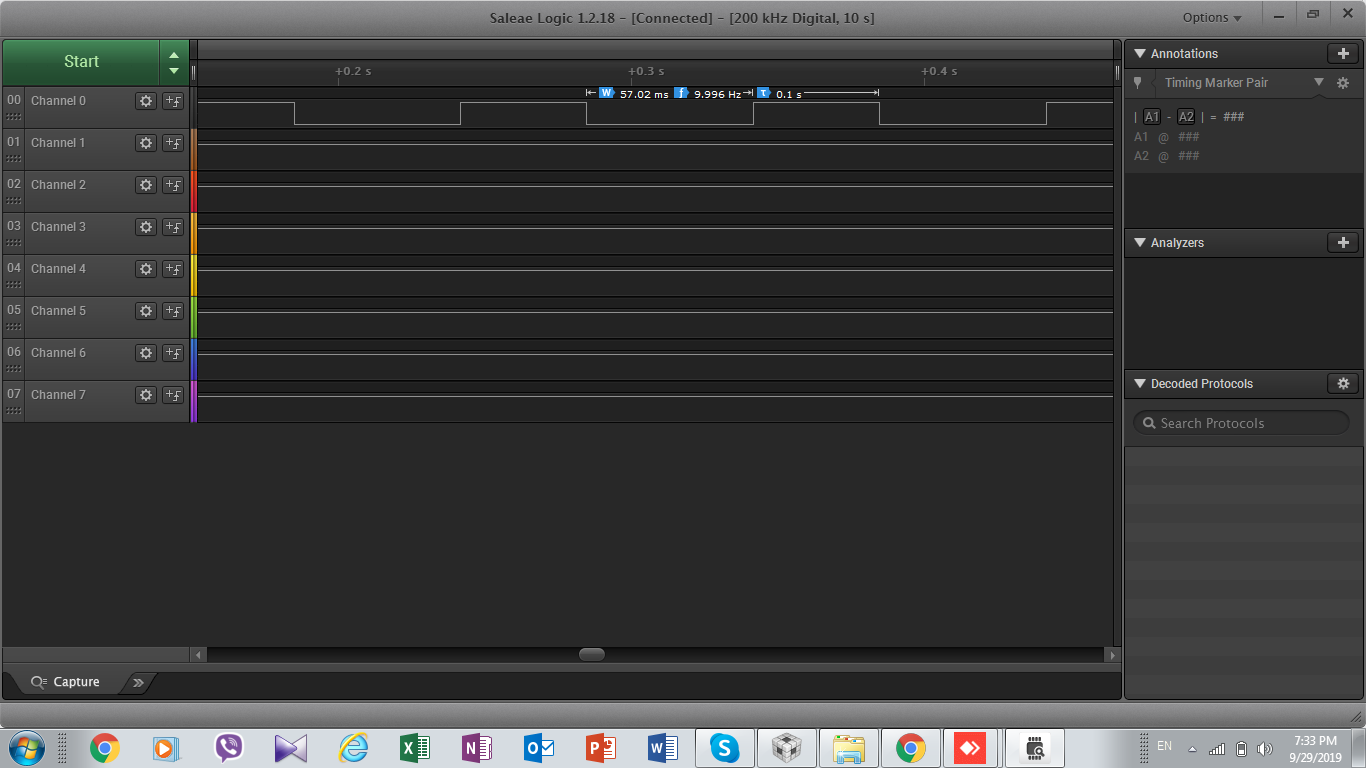
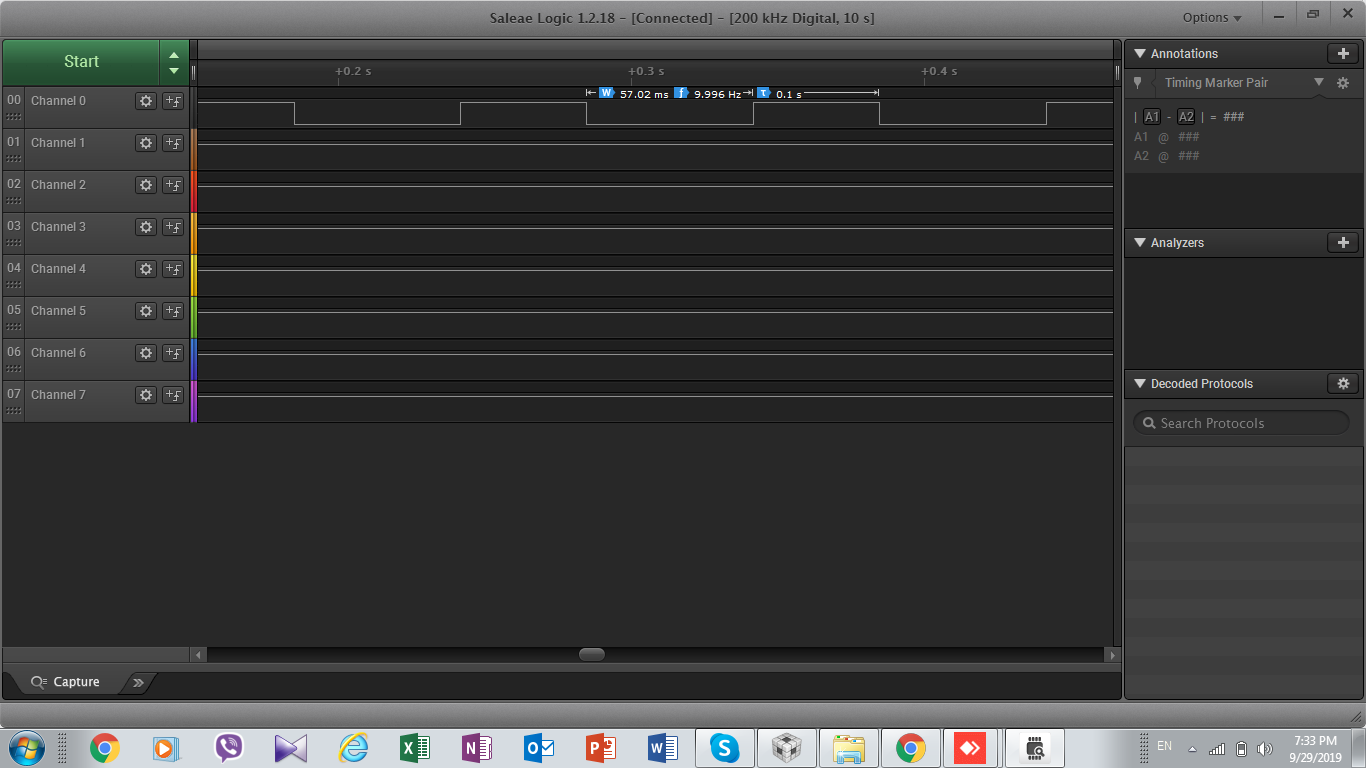
Youtube Link: N/A

**Modified Schematic (if applicable): N/A**

The following screenshots are the verification of %43 duty cycle at 10 Hz using Timer0.







Code for Task 01:

**Modified Code:**

**// Insert code here**

**#include** <stdint.h>

**#include** <stdbool.h>

**#include** "inc/tm4c123gh6pm.h"

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

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

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

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

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

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

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

{

uint32\_t ui32Period;

**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);

ui32Period = ((**SysCtlClockGet**() / 10)\*43) /100;

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period -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))

{

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

uint32\_t ui32Period = ((**SysCtlClockGet**() / 10)\*57) /100;

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period -1);

}

**else**

{

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

uint32\_t ui32Period = ((**SysCtlClockGet**() / 10)\*43) /100;

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period -1);

}

}

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

**Task 02:**

Youtube Link: <https://www.youtube.com/watch?v=mclfv-lnJM8>

**Modified Schematic (if applicable): N/A**

**Code for task02**

**Modified Code:**

**// Insert code here**

**#include** <stdint.h>

**#include** <stdbool.h>

**#include** "inc/tm4c123gh6pm.h"

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

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

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

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

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

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

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

{

uint32\_t ui32Period;

**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);

GPIO\_PORTF\_LOCK\_R = 0x4C4F434B;

GPIO\_PORTF\_CR\_R = 1;

**GPIOPadConfigSet**(GPIO\_PORTF\_BASE, GPIO\_PIN\_0, GPIO\_STRENGTH\_4MA,GPIO\_PIN\_TYPE\_STD\_WPU);

GPIO\_PORTF\_LOCK\_R = 0x4C4F434B;

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

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

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

**TimerConfigure**(TIMER1\_BASE, TIMER\_CFG\_ONE\_SHOT);

ui32Period = ((**SysCtlClockGet**() / 10)\*43) /100;

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period -1);

ui32Period = **SysCtlClockGet**();

**TimerLoadSet**(TIMER1\_BASE, TIMER\_A, ui32Period -1);

**IntEnable**(INT\_TIMER0A);

**IntEnable**(INT\_TIMER1A);

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

**TimerIntEnable**(TIMER1\_BASE, TIMER\_TIMA\_TIMEOUT);

**IntMasterEnable**();

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

**while**(1)

{

uint32\_t value = **GPIOPinRead**(GPIO\_PORTF\_BASE, GPIO\_PIN\_0);

**if**(!value)

{

**while**(!**GPIOPinRead**(GPIO\_PORTF\_BASE, GPIO\_PIN\_0));

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

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, 0x04);

ui32Period = **SysCtlClockGet**();

**TimerLoadSet**(TIMER1\_BASE, TIMER\_A, ui32Period -1);

**TimerEnable**(TIMER1\_BASE, TIMER\_A);

}

}

}

**void** **Timer1IntHandler**(**void**)

{

**TimerIntClear**(TIMER1\_BASE, TIMER\_TIMA\_TIMEOUT);

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

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

}

**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))

{

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

uint32\_t ui32Period = ((**SysCtlClockGet**() / 10)\*57) /100;

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period -1);

}

**else**

{

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

uint32\_t ui32Period = ((**SysCtlClockGet**() / 10)\*43) /100;

**TimerLoadSet**(TIMER0\_BASE, TIMER\_A, ui32Period -1);

}

}

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