Serak Gebremedhin
Github root directory: https://github.com/Ber-geb/solid-octo-tribble.git

Date Submitted: **10/23/19**

Task 01:

Comprehensive commented file of the of the original code below

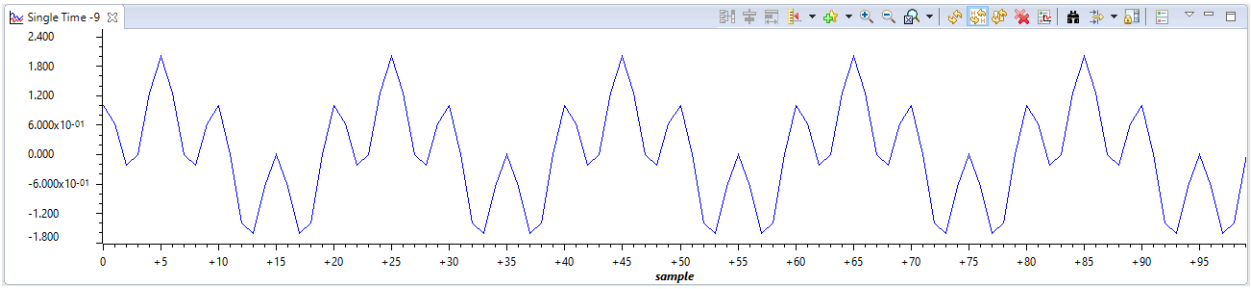
Modified Code:

```
// Insert code here
#include <stdint.h>
#include <stdbool.h>
#define TARGET_IS_BLIZZARD_RB1
#include <math.h> //the code uses the sinf() function prototyped by this header file
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "driverlib/fpu.h" //support for floating point unit
#include "driverlib/sysctl.h"
#include "driverlib/rom.h"

#ifndef M_PI //just in case M_PI is not already defined, this code will do so
#define M_PI                    3.14159265358979323846
#endif

#define SERIES_LENGTH 100 //this is the depth of our data buffer

float gSeriesData[SERIES_LENGTH]; //array of floats SERIES_LENGTH long

int32_t i32DataCount = 0; //a counter

int main(void)
{
    float fRadians; //float variable used to calculate sine

    ROM_FPULazyStackingEnable(); //turn on lazy stacking
    ROM_FPUEnable(); //turn on FPU

    ROM_SysCtlClockSet(SYSCTL_SYSDIV_4 | SYSCTL_USE_PLL | SYSCTL_XTAL_16MHZ |
SYSCTL_OSC_MAIN);
    //set up system clock 50 MHz

    fRadians = ((2 * M_PI) / SERIES_LENGTH);
    //full sine wave cycle is 2pi radians...divide 2pi by the depth of the array

    while(i32DataCount < SERIES_LENGTH) //loop while i32DataCount < SERIES_LENGTH
    {
        gSeriesData[i32DataCount] = sinf(fRadians * i32DataCount);
        //calculate sine value for each of the 100 values of the angle and place them
        //in the data array
        i32DataCount++; //increment
    }

    while(1) //infinite loop
    {
```

```
    }
}
```

Task 02:

Screen Capture of equation wave:



Modified Code:

```
// Insert code here
#include <stdint.h>
#include <stdbool.h>
#define TARGET_IS_BLIZZARD_RB1
#include <math.h> //the code uses the sinf() function prototyped by this header file
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "driverlib/fpu.h" //support for floating point unit
#include "driverlib/sysctl.h"
#include "driverlib/rom.h"

#ifndef M_PI //just in case M_PI is not already defined, this code will do so
#define M_PI                    3.14159265358979323846
#endif

#define SERIES_LENGTH 1000 //this is the depth of our data buffer

float gSeriesData[SERIES_LENGTH]; //array of floats SERIES_LENGTH long

int32_t i32DataCount = 0; //a counter

int main(void)
{
    float fRadians; //float variable used to calculate sine

    ROM_FPULazyStackingEnable(); //turn on lazy stacking
    ROM_FPUEnable(); //turn on FPU

    ROM_SysCtlClockSet(SYSCTL_SYSDIV_4 | SYSCTL_USE_PLL | SYSCTL_XTAL_16MHZ |
SYSCTL_OSC_MAIN);
    //set up system clock 50 MHz

    fRadians = ((2 * M_PI) / SERIES_LENGTH);
    //full sine wave cycle is 2pi radians...divide 2pi by the depth of the array
```

```
    while(i32DataCount < SERIES_LENGTH) //loop while i32DataCount < SERIES_LENGTH
    {
        gSeriesData[i32DataCount] = sinf(fRadians * (50 * i32DataCount)) +
cosf(fRadians * (200 * i32DataCount));
        //sin (2π*50t)+0.5*cos(2π*200t)
        //generates frequency 5Hz
        //calculate sine value for each of the 100 values of the angle and place them
        //in the data array
        i32DataCount++; //increment
    }

    while(1) //infinite loop
    {
    }
}
```

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