**Date Submitted: 12/13/2019**

**Task 00: Execute provided code**

**Task 01:**

**Modified Code:**

}

#include <stdint.h>

#include <stdbool.h>

#include <math.h>

#include "inc/hw\_memmap.h"

#include "inc/hw\_types.h"

#include "driverlib/fpu.h"

#include "driverlib/sysctl.h"

#include "driverlib/rom.h"

// If there is no previous definition of PI, Define it here.

#ifndef *M\_PI*

#define *M\_PI* 3.14159265358979323846

#endif

// Define a series length of 100

#define SERIES\_LENGTH 100

// A float array of length 100

float gSeriesData[SERIES\_LENGTH];

// Counter i = 0;

int32\_t i32DataCount = 0;

int main(void)

{

float fRadians;

// Faster speed for less accuracy

ROM\_FPULazyStackingEnable();

ROM\_FPUEnable();

// Set Clock speed

ROM\_SysCtlClockSet(SYSCTL\_SYSDIV\_4 | SYSCTL\_USE\_PLL | SYSCTL\_XTAL\_16MHZ | SYSCTL\_OSC\_MAIN);

// 2pi/100

fRadians = ((2 \* *M\_PI*) / SERIES\_LENGTH);

// for i < 100;

while(i32DataCount < SERIES\_LENGTH)

{

// Saving points in a sine wave in gSeries Data

gSeriesData[i32DataCount] = *sinf*(fRadians \* i32DataCount);

// i++

i32DataCount++;

}

while(1)

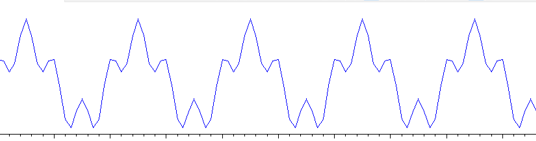
{

}

}

**Task 02:**

gSeriesData:



**Modified Code:**

#include <stdint.h>

#include <stdbool.h>

#include <math.h>

#include "inc/hw\_memmap.h"

#include "inc/hw\_types.h"

#include "driverlib/fpu.h"

#include "driverlib/sysctl.h"

#include "driverlib/rom.h"

// If there is no previous definition of PI, Define it here.

#ifndef *M\_PI*

#define *M\_PI* 3.14159265358979323846

#endif

// Define a series length of 100

#define SERIES\_LENGTH 100

// A float array of length 100

float gSeriesData[SERIES\_LENGTH];

// Counter i = 0;

int32\_t i32DataCount = 0;

int main(void)

{

float fRadians;

// Faster speed for less accuracy

ROM\_FPULazyStackingEnable();

ROM\_FPUEnable();

// Set Clock speed

ROM\_SysCtlClockSet(SYSCTL\_SYSDIV\_4 | SYSCTL\_USE\_PLL | SYSCTL\_XTAL\_16MHZ | SYSCTL\_OSC\_MAIN);

// 2pi/100

fRadians = ((2 \* *M\_PI*) / SERIES\_LENGTH);

// for i < 100;

while(i32DataCount < SERIES\_LENGTH)

{

// Saving points in a sine wave in gSeries Data

gSeriesData[i32DataCount] = (1.5 + *sinf*(fRadians \* (50\*i32DataCount))) + (0.5\**cosf*(fRadians\*(200\*i32DataCount)));

// i++

i32DataCount++;

}

while(1)

{

}

}