Abenezer Namaga

CPE 403 – 1001

TIVAC LAB9

|  |
| --- |
|  |

Github root directory: https://github.com/Ayertena/AdvEmbededSys

Date Submitted: 10/16/2018

**Task 01:** Submit a comprehensive commented file of the original code.

#include <stdint.h> // Variable definitions for the C99 standard  
#include <stdbool.h> // Boolean definitions for the C99 standard  
#include <math.h> // Provides access to sinf()  
#include "inc/hw\_memmap.h" // Macros defining the memory map of the Tiva C Series device.

// This includes defines such as peripheral base address locations

// such as GPIO\_PORTF\_BASE.  
#include "inc/hw\_types.h" // Defines common types and macros  
#include "driverlib/fpu.h" // Provides access floating point values #include "driverlib/sysctl.h" // Defines and macros for System Control API of DriverLib.

#include "driverlib/rom.h" // ROM functions

#ifndef M\_PI  
#define M\_PI 3.14159265358979323846 // Value of pi

#endif

#define SERIES\_LENGTH 100 // Amount to calculate float gSeriesData[SERIES\_LENGTH]; // Stores the evaluated value int32\_t i32DataCount = 0; // Keeps count # of calculations made

int main(void)

{  
 float fRadians; // Declare variables to hold 2pi / 100 ROM\_FPULazyStackingEnable(); // Enable lazy stacking

ROM\_FPUEnable(); // Enable FPU

// 50 MHz clock  
ROM\_SysCtlClockSet(SYSCTL\_SYSDIV\_4 | SYSCTL\_USE\_PLL |

SYSCTL\_XTAL\_16MHZ | SYSCTL\_OSC\_MAIN);

fRadians = ((2 \* M\_PI) / SERIES\_LENGTH); // 2pi / 100

while(i32DataCount < SERIES\_LENGTH) { // 100 calculations are made

gSeriesData[i32DataCount] = sinf(fRadians \* i32DataCount); // Get sine value

i32DataCount++; } // Increment to obtain next sine value

while(1){} }

-------------------------------------------------------------------------------------------------------------------------------

**Task 02:** Modify the code to implement the below equation to generate a frequency of 5 Hz. Display the equation for 1 sec. 1.0\*sin(2p50t) + 0.5\*cos(2p 200t)

**Youtube Link:**

#include <stdint.h> // Variable definitions for the C99 standard  
#include <stdbool.h> // Boolean definitions for the C99 standard  
#include <math.h> // Provides access to sinf()  
#include "inc/hw\_memmap.h" // Macros defining the memory map of the Tiva C Series device.

// This includes defines such as peripheral base address locations

// such as GPIO\_PORTF\_BASE.  
#include "inc/hw\_types.h" // Defines common types and macros  
#include "driverlib/fpu.h" // Provides access floating point values #include "driverlib/sysctl.h" // Defines and macros for System Control API of DriverLib.

#include "driverlib/rom.h" // ROM functions

#ifndef M\_PI  
#define M\_PI 3.14159265358979323846 // Value of pi

#endif

#define SERIES\_LENGTH 100 // Amount to calculate float gSeriesData[SERIES\_LENGTH]; // Stores the evaluated value int32\_t i32DataCount = 0; // Keeps count # of calculations made

int main(void)

{  
 float fRadians; // Declare variables to hold 2pi / 100 ROM\_FPULazyStackingEnable(); // Enable lazy stacking

ROM\_FPUEnable(); // Enable FPU

// 50 MHz clock  
ROM\_SysCtlClockSet(SYSCTL\_SYSDIV\_4 | SYSCTL\_USE\_PLL |

SYSCTL\_XTAL\_16MHZ | SYSCTL\_OSC\_MAIN);

fRadians = ((2 \* M\_PI) / SERIES\_LENGTH); // 2pi / 100

while(i32DataCount < SERIES\_LENGTH) { // 100 calculations are made

// 1.0\*sin(2p50t) + 0.5\*cos(2p 200t)

// Get sine value

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

i32DataCount++; // Increment to obtain next sine value }

while(1) // Continue for ever

{  
 }

}

---------------------------------------------------------------------------------------------------------------------

Waveform

