



MANHATTAN
COLLEGE

Lab 4: FPU

School of Engineering

Electrical and Computer
Engineering Department

ECEG 721-61 Embedded
Systems

Jamie Quinn & Lucia
Rosado-Fournier

November 6, 2020



Table of Contents

1.1 Objective	3
1.2 Introduction.....	3
1.3 Component Requirements	3
1.4 Software	3
Flowchart.....	4
C code.....	5
1.5 Procedure.....	5
1.6 Observation	6
1.7 Summary.....	6

1.1 Objective

The objective of this experiment was to enable the floating-point unit (FPU) to allow floating point code to be run on the microcontroller.

1.2 Introduction

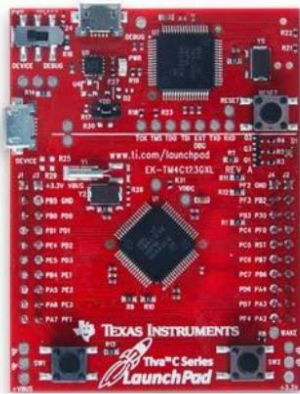
In this experiment we enabled the FPU on the microcontroller to calculate a full cycle of a sine wave, storing the values into an array that has the length of 100. The FPU allows the execution of calculations that require non-integer numbers. In the case of the sine wave, pi, a floating point, is needed, so the FPU is needed to achieve this. The points calculated were then inputted into a graphing software to show the sine wave.

1.3 Component Requirements

1.3.1 Software Requirements

1. Keil uVision4
2. TivaWare_C_Series

1.3.2 Hardware Requirements

S.No	Components	Specifications	Images
1.	Tiva LaunchPad	EK-TM4C123GXL LaunchPad	
2.	USB cable		

1.4 Software

The software for the experiment is written in C and developed using the Keil Integrated Development Environment (IDE). The software is programmed into the target device TM4C123GH6PM on the EK-TM4C123GXL using the USB interface.

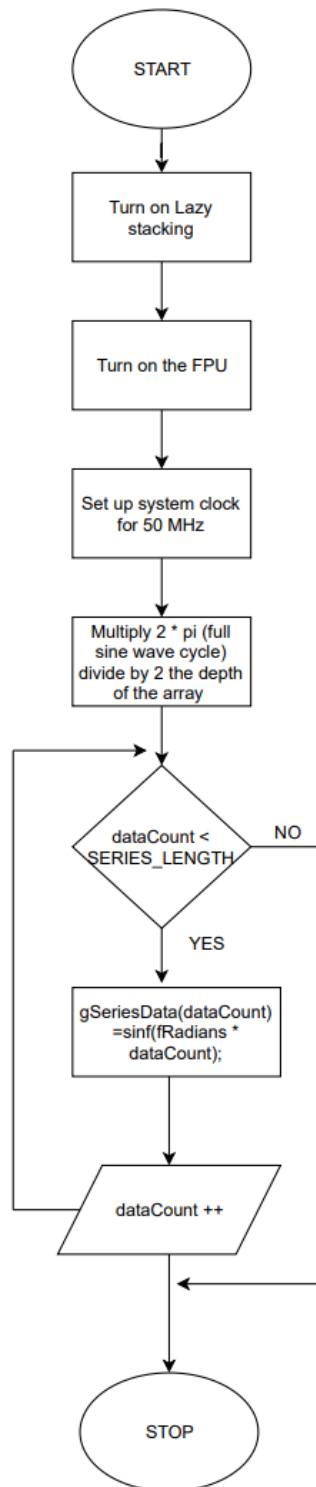
Flowchart

Figure 1 – Flowchart for enabling the FPU and creating the sine wave.

C code

```
#define TARGET_IS_TM4C123_RB1
#include <math.h>
#include <stdint.h>
#include <stdbool.h>
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "driverlib/fpu.h"
#include "driverlib/sysctl.h"
#include "driverlib/rom.h"
//Define value of Pi if it's not defined
#ifndef M_PI
    #define M_PI 3.14159265358979323846
#endif
//Length of the array
#define SERIES_LENGTH 100
//Define the array and initialize the data count for the loop later
float gSeriesData[SERIES_LENGTH];
int dataCount = 0;

int main(void) {
    float fRadians;
    //Enable the FPU and configure the clock
    ROM_FPULazyStackingEnable();
    ROM_FPUEnable();
    ROM_SysCtlClockSet(SYSCTL_SYSDIV_4|SYSCTL_USE_PLL|SYSCTL_XTAL_16MHZ|S
YSCTL_OSC_MAIN);
    //Calculate the radians
    fRadians = ((2 * M_PI) / SERIES_LENGTH);
    //Calculate the value for each point until it's done for 100 points
    while(dataCount < SERIES_LENGTH) {
        gSeriesData[dataCount] = sinf(fRadians * dataCount);
        dataCount++;
    }
    while(1) { }
}
```

1.5 Procedure

The procedure to complete this lab is as follows:

1. Open Keil and create the Lab project in main.c. Copy and paste the c code given into main.c.
2. Click the start/stop debug session button to open the debugger
3. Click the view button and open a watch window. Enter the expression “gSeriesData” into the watch window.
4. Observe the values and create a graph on excel.

1.6 Observation

After following the procedure detailed above, we were able to get the 100 data points needed to create the sine wave. To better visualize the sine wave, the points were taken and inputted into Excel to create the graph below. As expected, the graph is a sine wave with the maximum being 1 and the minimum being -1.

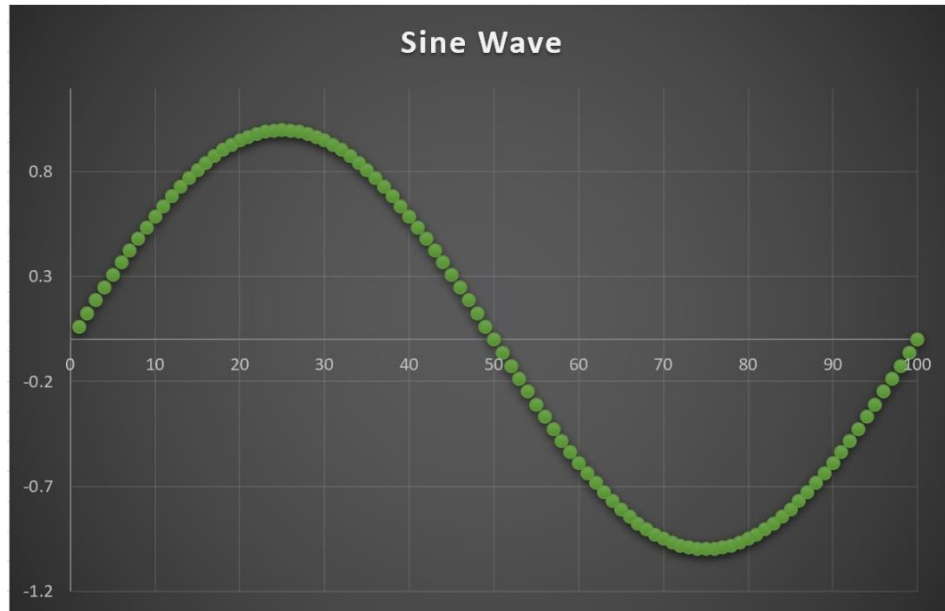


Figure 2 – The 100 points plotted to create the sine wave.

1.7 Summary

The experiment went as expected and there were no major issues to report. Overall, the experiment and report writing took about two hours in total.