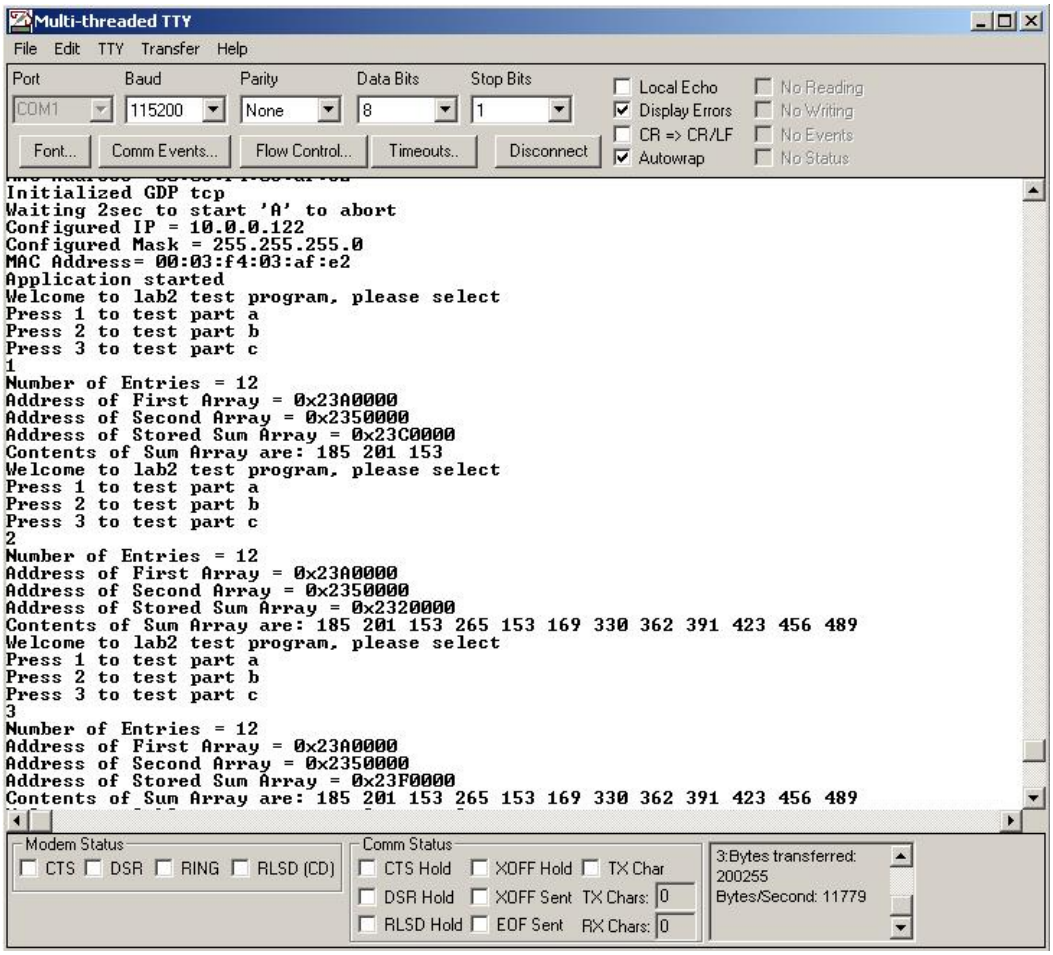
Lab 2: Introduction to Addressing Modes



Lab Dates

Refer to the schedule on the ECE212 Laboratories page for the latest schedule

Introduction

In this lab the students will be introduced to a few of the different types of addressing modes in assembly language programming.

Objectives:

1. To gain experience in developing code in assembly language
2. To gain experience in accessing memory with different addressing modes

Prelab and Preparation:

1. Read the lab prior to coming to your lab section.
2. Do the online prelab quiz
3. Provide flowcharts for the 3 sections in part A and one for Part B. (4 total)

Lab Work and Specifications

1. Download the template files
    - main.cpp
    - DataStorage.s
    - SetZeros.s
    - Lab2a.s
    - Lab2b.s

Specifications

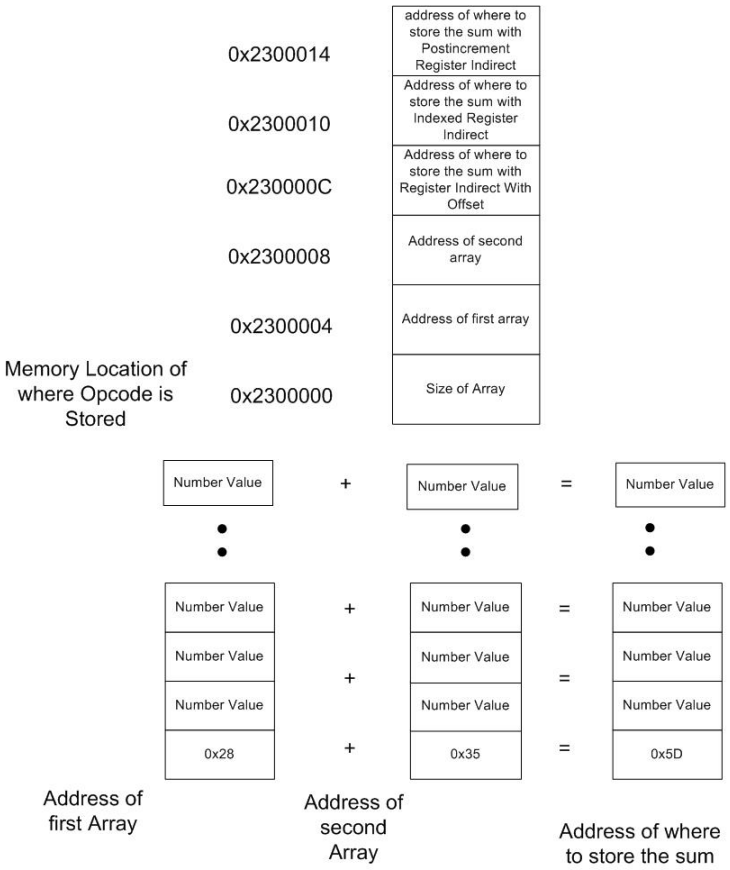
If you recall from the Lab1, the 'main.cpp' is the standard project template that is used to initialize and call standard functions/subroutines including our 'AssemblyProgram'. Do not modify any of the parameters in this file. 'SetZeros.s' and 'DataStorage' are also provided to initialize the memory contents. Do not modify any of the parameters in either file. Lab2a.s is provided for you to write your assembly language program for part A and Lab2b.s is provided for part B. All 3 parts for Part A should reside is Lab2a.s and part B should reside in Lab2b.s. A more detail description is provided below.

Part A - Different Addressing Modes

You are asked to write a program that will add the contents of two arrays stored at different memory locations and place the result at another specified memory location. The operational code will be provided at memory location 0x2300000. The operational code contains the information for where to access your arrays in memory and where to store your sum. Each piece of information is stored as a longword(32bits). For instance the first longword stored at memory location 0x2300000 will contain the size of the two arrays. The next longword will contain the address of where the first array is stored in memory. The next longword will contain the address of where the second array is stored in memory. A breakdown of the operational code is illustrated as follows

1. 0x2300000 - Size of array
2. 0x2300004 - address of first array
3. 0x2300008 - address of second array
4. 0x230000C - address of where to store the sum with Register Indirect With Offset
5. 0x2300010 - address of where to store the sum with Indexed Register Indirect
6. 0x2300014 - address of where to store the sum with Postincrement Register Indirect

Note: Each piece of information is stored as a longword(32bits)



Part 1

In part 1, you will perform the addition of the two arrays using the Register Indirect With Offset. You need only perform the addition of the first 3 values in the array to prove that you understand how to use this mode of operation.

Part 2

Repeat part 1 using the Indexed Register Indirect method. You need to perform the addition of the entire array.

Part 3

Repeat part 1 using the Postincrement Register Indirect. You need to perform the addition of the entire array.

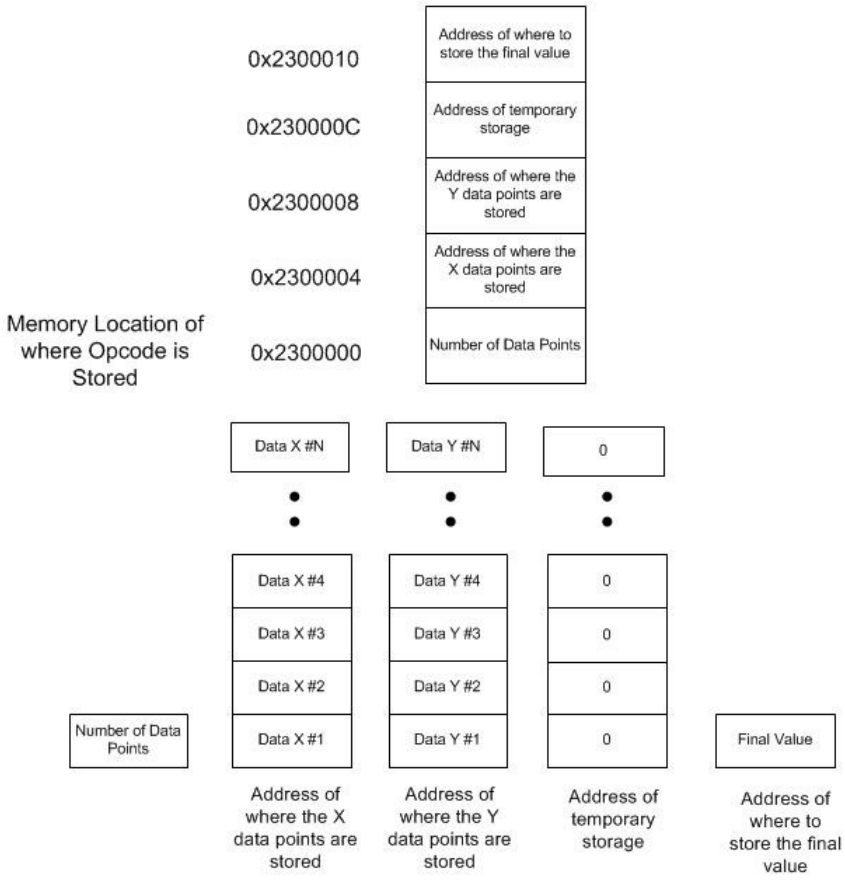
Note: Solutions for each part should reside in Lab2a.s

Part B - Trapezoidal rule

You are asked to write a program that will calculate the area underneath a curve(y = f(x)) using the Trapezoidal rule that you learned in your Calculus course. The data points for the curve(X and Y data points) are stored in arrays at specific spots in memory. For convenience, the distance between each X data point is either one or two unit. The operational code will be provided at memory location 0x2300000. The operational code contains the information such as number of data points, where to access the array of data points(X and Y) in memory, temporary storage space for use and where to store the final area value. Each bit of information is stored as a longword(32bits). For instance the first longword stored at memory location 0x2300000 will contain the number of data points. The next two longwords will contain the address of where the X and Y data points are stored in memory. The next long word contain an addresse for where you can store temporary values. The last longword will contain the address of where to store the final area memory. A breakdown of the operational code is illustrated as follows. For this part, please download and use only **Datastorage5.s and SetZeros5.s**

1. 0x2300000 - Number of Data Points
2. 0x2300004 - address of where the X data points are stored
3. 0x2300008 - address of where the Y data points are stored
4. 0x230000C - address of temporary storage space
5. 0x2300010 - address of where to store the final value

Note: Each piece of information is stored as a longword(32bits)



Questions

1. What are the advantages of using the different addressing modes covered in this lab?
2. If the differece between the X data points are not restricted to be either one or two units, how would you modify your program to calculate the area? You do not need to do this in your code.
3. From the data points, what is the function (y=f(x))? What is the percent error between the theoretical calculated area and the one obtain in your program?

Marking Scheme

Lab 2 is worth 25 % of the final lab mark. Please view the Marking Sheet for this lab to ensure that you have completed all of the requirements of the lab. The Marking Sheet also provides a limited test suite in the demo section for you to think about. Make use of it! The report is to follow the Report Writing Guidelines

Demo and Report

The **report and demo due dates** are given on the ECE212 Laboratories page on Moodle. Note that you have one week from the dating of your prelab to complete the demo.

The reports must be handed in by **4 P.M. Do not be late**.

Late Submissions

Late submission penalties for the demo and report portions of the labs are given on the ECE212 Moodle website