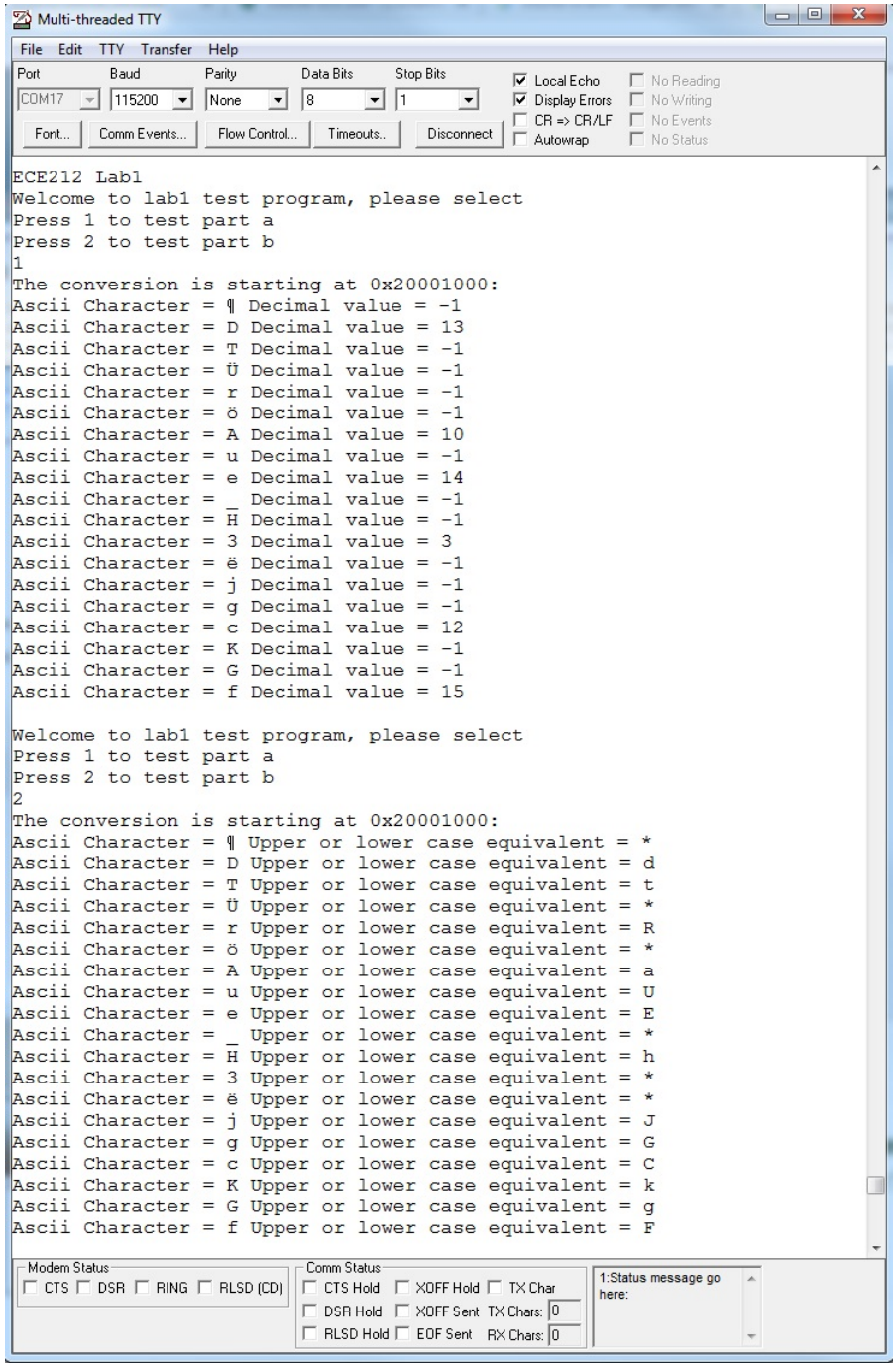
Lab 1: Introduction to ARM Assembly language



Lab Dates

Refer to the ECE212 Canvas lab website for the latest schedule

Introduction

In the first lab, students will be introduced to some of the basic instruction set used in ARM assembly language programming. There are two different instruction sets(ARM and THUMB) that are supported by the ARM microprocessor. The labs will only support the THUMB2 instruction set.

Objectives:

1. To gain experience developing code in THUMB2 assembly language
2. To gain experience working with the NUCLEO board(NUCLEO-L432KC) manufactured by STMicroelectronics
3. To gain experience working with STM32CubeIDE software.

Prelab and Preparation:

- Read the lab prior to attending your lab session. Please only attend the section that you are registered in.
- Do the online prelab Quiz **individually** and submit it before the deadline
- Prepare rough drafts of the flow chart(s)/pseudocode for each part of the lab(Part A and Part B). You do not neet to use a CAD tool for this. The final CAD version will be submitted in your lab report.

Lab Work and Specifications

1. Download the template files
    - main_L432KC.c
    - retarget.c
    - retarget.h
    - DataStorage.s
    - SetZeros.s
    - Lab1a_L432KC.s (For part A)
    - Lab1b_L432KC.s (For part B)
2. Create an empty project and import the above files into it. For a quick referesher, consult the tutorial on how to do this. If you are working on Lab1a.s, exclude Lab1b.s from the project and vice versa. Compiling both files at once will cause an error.
3. Review the marking sheet for the breakdown of marks.

If you recall from the tutorial, the 'main.c' is the standard project template that is used to initialize and call standard functions/subroutines including our 'AssemblyProgram'. This will always be provided and you don't need not worry too much about it. 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. 'Lab1a.s' and 'Lab1b.s' are the two files provided for you to write your assembly language program. There are no passed parameters and your solution is expected to reside entirely in these two files. A more detail description is provided below.

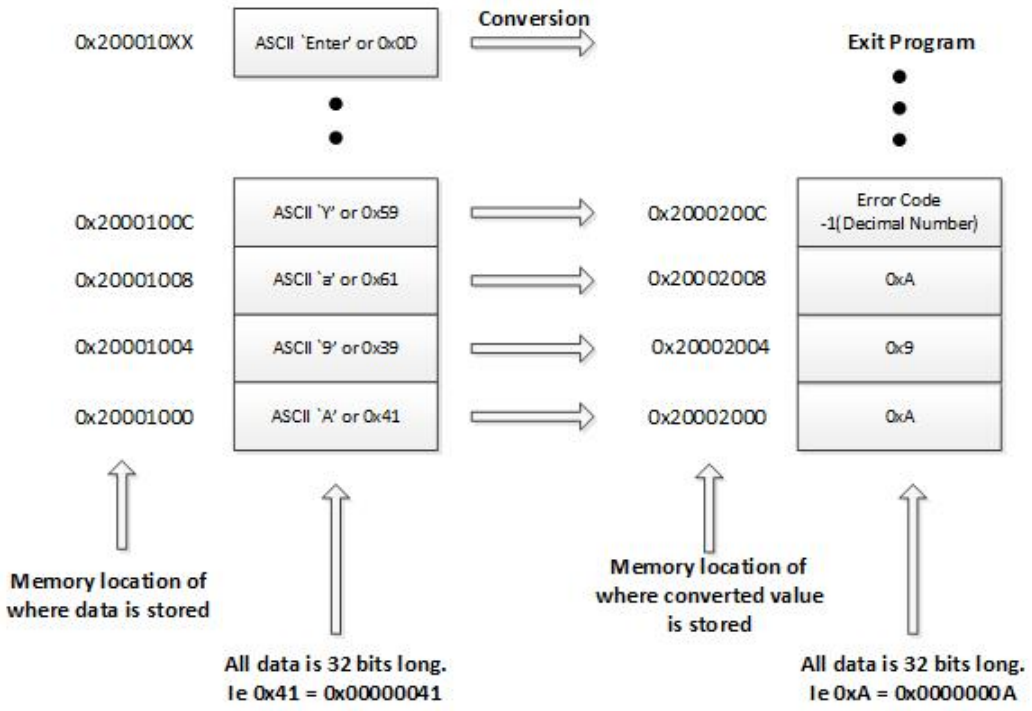
Specifications

Part A

Use the Lab1a.s template for your coding

You are asked to write a program that will convert an ASCII character(stored as an ASCII code in memory) to its hex/dec equivalent. Valid ASCII characters are from 0 to 9, A-F and a-f. For example, the ASCII character 'A' when stored in memory will be represented by the hex value 0x41 or one byte(8bits) of data. Each memory location on the development board contains one byte of information. In other words, one memory location can store one ASCII character. However, for our lab, we are going to use 4 memory locations to store one ASCII character by padding it with a bunch of zeros. A diagram below will give you a visual interpretation of how the data is stored. You are asked to convert this value to its hex/dec equivalent(hex value = 0xA)(decimal value = 10) and store it at some other memory location made up of 4 memory blocks. Invalid ASCII characters(ie,+,*) should be rejected with an error code stored in the memory location. The error code has been chosen to be the numberical value '-1'. This error code is 4 bytes(word)(32bit) long. If the memory location contains the ASCII 'Enter' code, your program should exit signaling the end and that no more conversions are required.

1. 0x20001000 - Start Address of where the data is stored
2. 0x20002000 - Start Address of where the converted values are to be stored



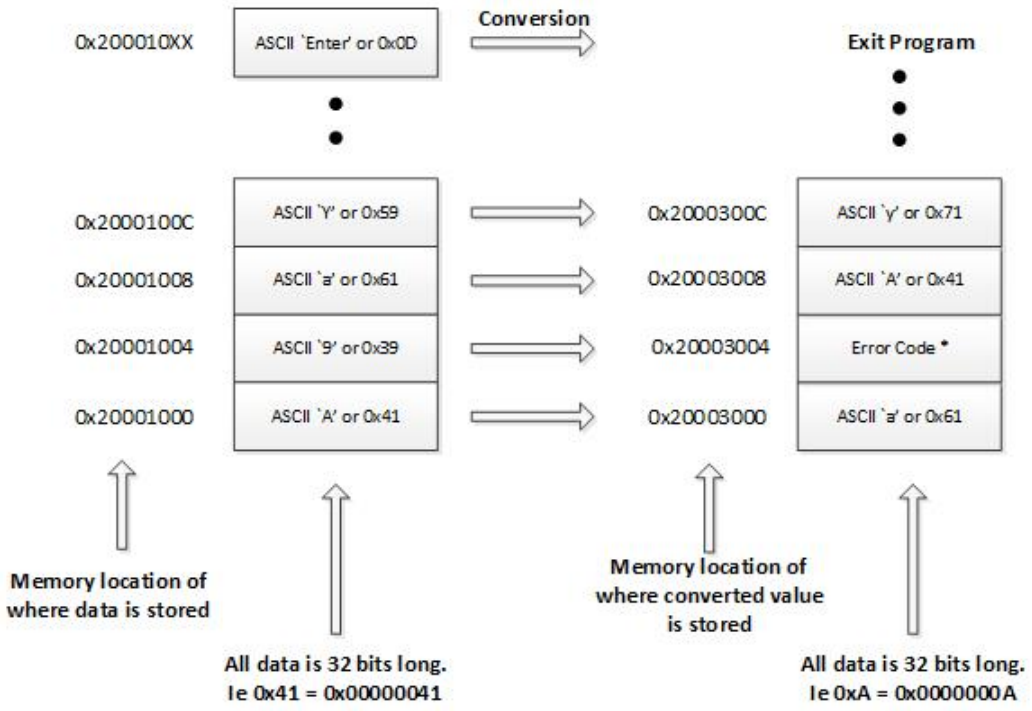
NOTE: Once you have completed part A, make sure you back up your lab1a.s by saving a copy of it somewhere. The best way is to back it up on your Google Drive or USB memory stick. If you proceed to part b without doing so, you may accidently delete your lab1a.s file and lose your solution.

Part B

Replace the Lab1a.s template with Lab1b.s for your coding. You can also exclude Lab1a.s from the project and work on Lab1b.s

You are asked to write a program that will convert an ASCII letter to its upper or lower case equivalent. Valid characters are all upper or lower case letters. For example, the ASCII character 'A' when stored in memory will be represented by the hex value 0x41. You are asked to convert this value to its lower case equivalent 'a' which will be represented by the hex value 0x61. This value is stored at another desginated memory location. Invalid ASCII characters should be rejected with an error code in the memory location. The error code has been chosen to be the ASCII character '*' If the memory location contains the ASCII 'Enter' code, your program should exit signaling the end and that no more conversions are required.

1. 0x20001000 - Start Address of where the data is stored
2. 0x20003000 - Start Address of where the converted values are to be stored



Questions

1. What happens when there is no exit code '0x0D' provided in the initialization process? Would it cause a problem? Why or Why not? How can we prevent this? Please give two options.

2. Your program when compiled is stored as binary bits. How does the mircroprosseor seperate where the instructions are stored and where the data is stored? Can you mix the two together in the same memory blocks?

Marking Scheme

Lab 1 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! When writing the report, please follow the ECE 212 Lab Report Guideline document uploaded to Canvas.

Demo and Report

The **report and demo due dates** are given on the ECE212 Laboratories page on Canvas.

During your demo, you solution will be graded on functionality(did you meet the requirements). In addition, we may you questions about your solution including specific instructions or ask you to make modificiatons to your program. **Only one demo is allowed** so make sure you thoroughly understand your code before requesting the demo. Late demo will not be accepted.

The report and source code must be submitted using the submission link on Canvas before **4:30 P.M. Do not be late** on your scheduled due date.

Late Submissions

There are late submission penalties for the Report(-10%) per day(2 days max) after the scheduled due date.