

**Khalifa University of Science, Technology and Research**

**Electronic Engineering Department**

**ELCE333 Microprocessor Systems laboratory**

**Laboratory Experiment 1**

**Microcontroller Assembly Program Development**

**Lab Partners**

Mohammad Jassem 100037002

Khalid Al-Zarouni 100035499

Khalid Al-Hosani 100037003

**Date Experiment Performed**: 28-1-2015

**Date Lab Report Submitted**: 4-2-2015

**Lab Instructor:**

**Dr. Mahmoud Khonji, and Dr. Mohammad Al Azaabi**

**Spring 2013**

# Table of Contents

Table of Contents 2

List of Figures and Tables 3

Summary 4

1. Introduction 5

1.1 Aim 5

1.2 Objectives 5

2. Design and results 6

**TASK - 1: Adding Numbers** 6

**TASK - 2: Subtracting Numbers** 7

**TASK-3: Simple Program:** 8

3. Assignment Questions 10

4. Conclusions and Recommendations 12

# List of Figures and Tables

Figure 1: The Code for Adding Numbers 6

Figure 2: The Code for Adding Numbers 7

Figure 3: object file content 9

Table 1: Registers contents while single stepping in the program 6

Table 2: Registers contents while single stepping in the program 7

Table 3: The memory contents for the developed program 9

# Summary

This lab is an introduction for both HCS12 micro controller and the assembly programs inside it. Two tasks were performed in this lab. It writing the assembly language, we will run, verify and practically test the results on the Dragon12-Plus-USB Trainer.

First of all, we create an assembly language program, which adds two numbers. After that, we edit the code and change it to subtract two numbers. At the end, we know how to obtain the instruction set information from the list file (LST).

# 1. Introduction

Microcontroller is an electronic integrated circuit that consists of four components, which are the Central processing unit (CPU) or the Microprocessor, The Read Only Memory (ROM), The Random Access Memory (RAM) and the Input/output (I/O) Interface. It also contains two buses, data bus and address bus.

Code Warrior IDE allow us to perform three operations which are editing, debugging and developing to any assembly language code for Microcontroller HCS12. In order to achieve this point it is recommended to understand all the operational code and the operands that are used in each op-code.

1.1 Aim

The aim of the lab is to deliver the knowledge and to perform several processes, which are developing, altering, compiling, debugging, simulating, downloading and executing a microcontroller assembly program.

1.2 Objectives

* Edit and develop a microcontroller HCS12 program
* Trace the program execution using single step debugging and check the system registers and memory contents.
* Use the HCS12 instruction manual as appropriate
* Understand the content of list file produced
* Download binary file into Dragon Plus board
* Produce a list file and binary file from the assembly code

# 2. Design and results

### **TASK - 1: Adding Numbers**

Using the instruction set written in the manual, we were able to write the following code that adds two numbers.

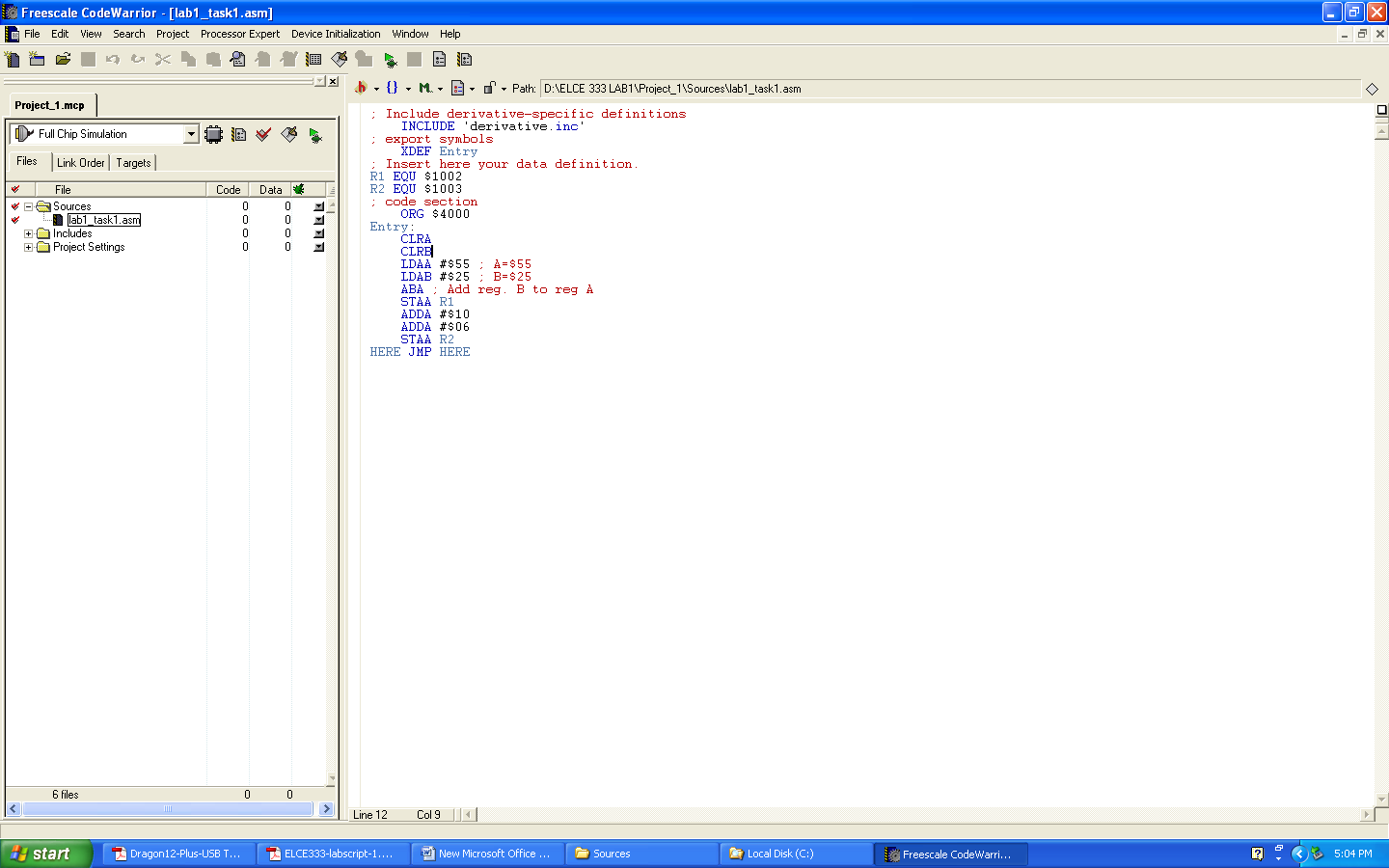


Figure : The Code for Adding Numbers

Using the step-by-step execution option, each command has been tested individually to see its corresponding execution action. However, in the following table, the result of each command is presented and the value of the registers A, B and D are recorded in accordance to the command.

Table 1: Registers contents while single stepping in the program

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Instruction* | *A* | *B* | *D* | *Comments* |
| *CLRA* | *00* | *00* | *0000* | *Register A is cleared* |
| *CLRB* | *00* | *00* | *0000* | *Register B si cleared* |
| *LDAA #$55* | *55* | *00* | *5500* | *$55 🡪 A* |
| *LDAB #$25* | *55* | *25* | *5525* | *$25 🡪 B* |
| *ABA* | *7A* | *25* | *7A25* | *Add Reg. B to A 🡪 A* |
| *STAA R1* | *7A* | *25* | *7A25* | *A 🡪 (R1)* |
| *ADDA #$10* | *8A* | *25* | *8A25* | *$10+A 🡪 A* |
| *ADDA #$06* | *90* | *25* | *9025* | *$06+A 🡪 A* |
| *STAA R2* | *90* | *25* | *9025* | *A 🡪 (R2)* |

Now the addresses $1002 & $1003 hold the values of A at the specified step.

### **TASK - 2: Subtracting Numbers**

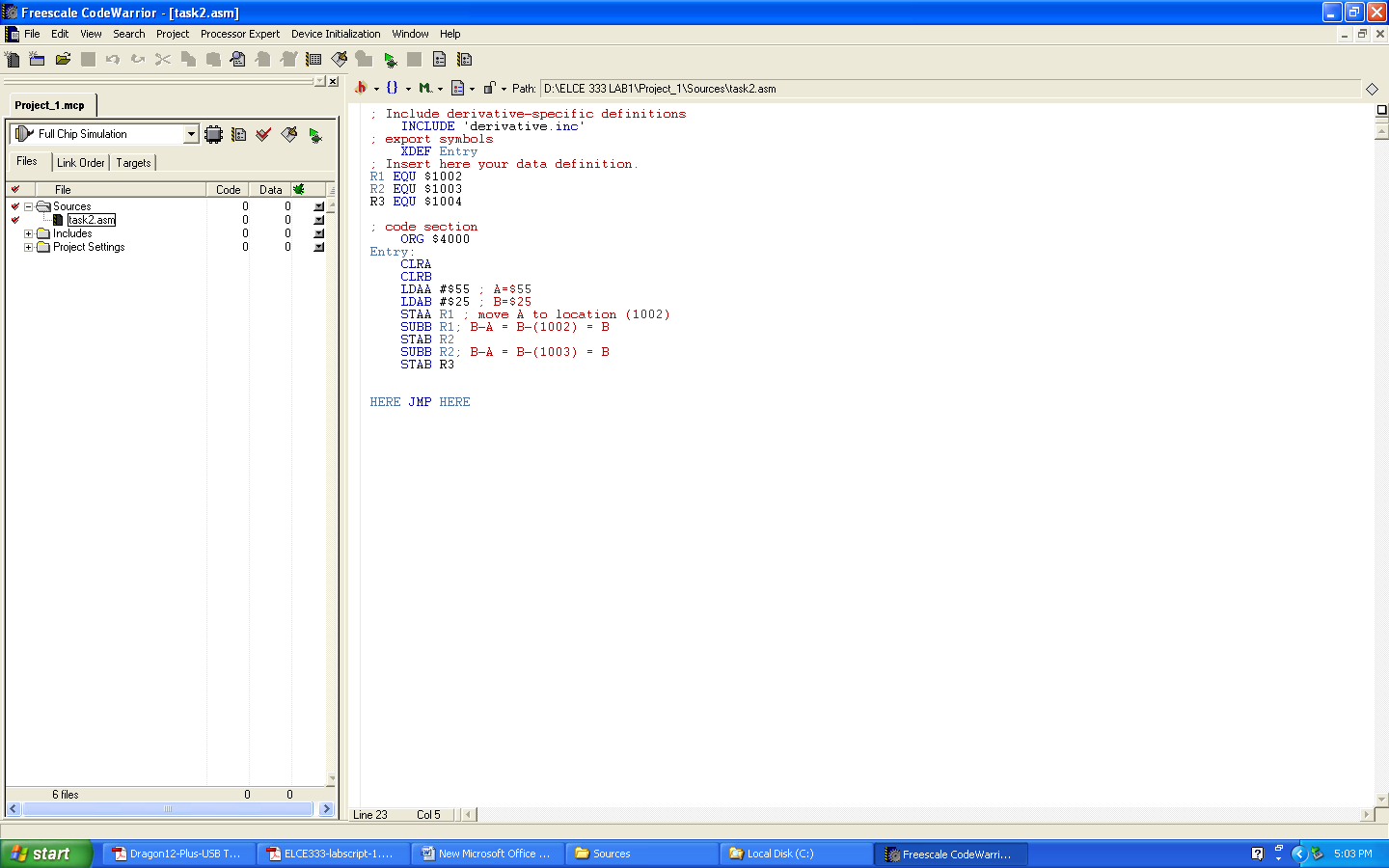
The previouslly written code has been modified to subtract two numbers instead of adding. Using the instruction set given, the code has been written as follows.

Figure : The Code for Adding Numbers

Using the step-by-step execution option, each command has been tested individually to see its corresponding execution action. However, in the following table, the result of each command is presented and the value of the registers A,B and D are recorded in accordance to the command.

By verifying the operation of this program on the dragon-12 plus, the results were proven correct.

### **TASK-3: Simple Program:**

Other important information could be obtained using the (LST) file, which can provide us with different data related to the code. It can provide us with the following data.

* A list of user-defined symbols used and their values
* The source-code program
* The assembled code in hexadecimal format (including memory locations)
* Any error messages

To generate the (LST) file, we take the first task as an example. Using CodeWarrior IDE project, the instructions mentioned in the manual have been followed. In the Project Manager Window, Full Chip Simulation has been used for debugging. The following figure shows the object file content.

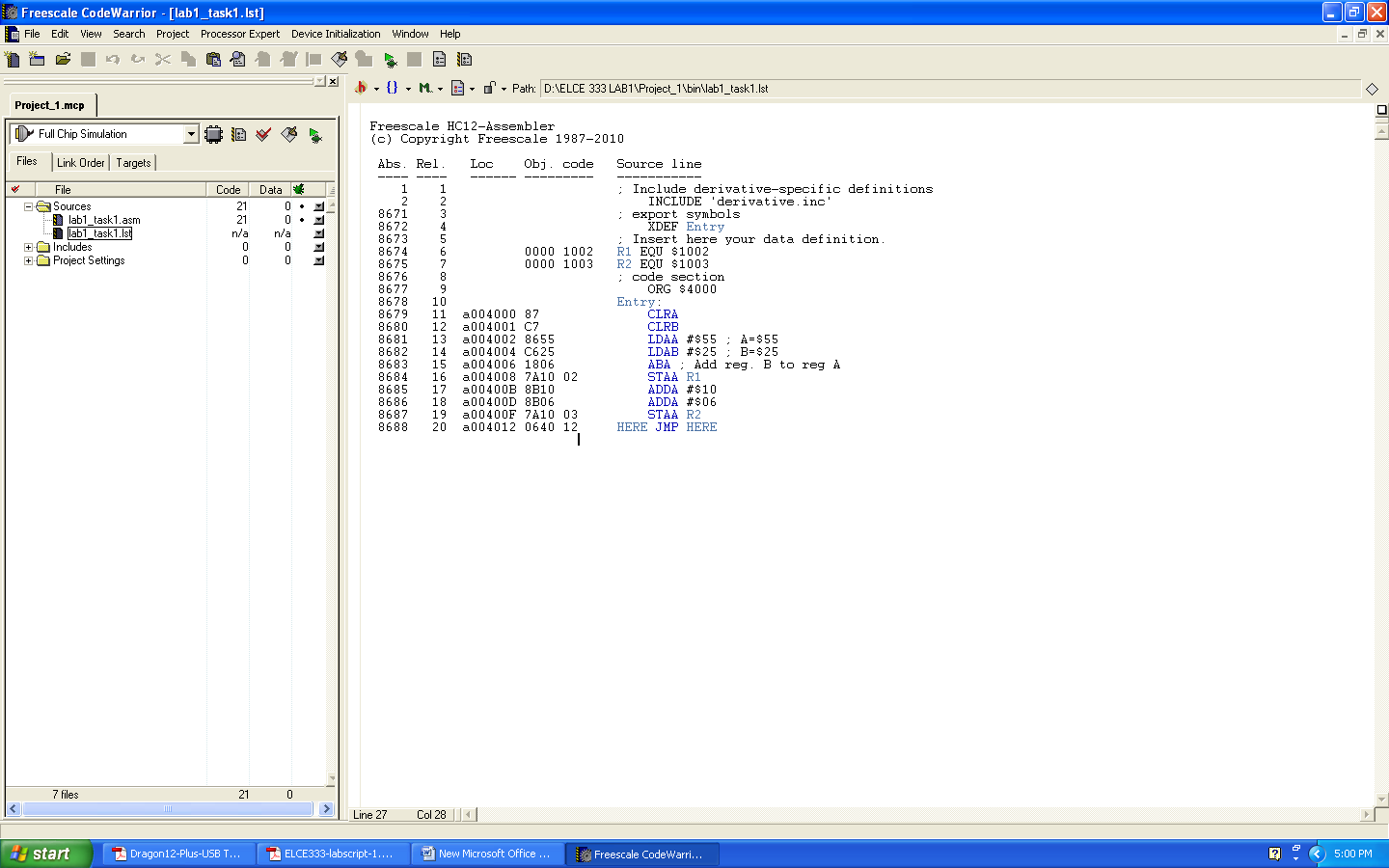
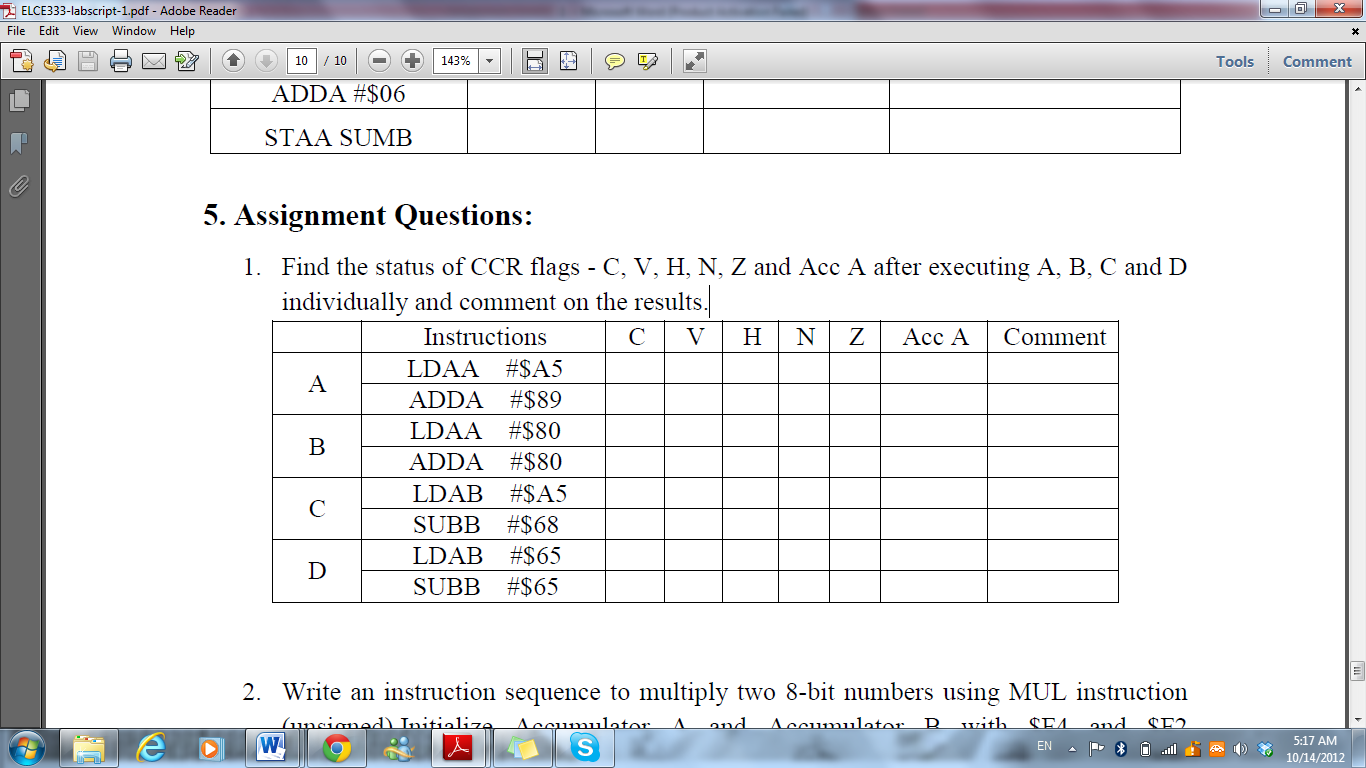


Figure : object file content

Table 3: The memory contents for the developed program

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Instruction | Start Address | No. of Bytes | Instruction Code | Comments |
| CLRA | $4000 | 1 | 87 | INH, no use of memory locations so only 1 byte |
| CLRB | $4001 | 1 | C7 | Also INH |
| LDAA #$55 | $4002 | 2 | 8655 | IMM, 2B, 1 byte for op-code and 1 for operand |
| LDAB #$25 | $4004 | 2 | C625 | IMM, 2B |
| ABA | $4006 | 2 | 1806 | IMM, 2B |
| STAA R1 | $4008 | 3 | 7A10 02 | IMM, 3B, 1B for op-code and 2B for operand |
| ADDA #$10 | $400B | 2 | 8B10 | IMM, 2B |
| ADDA #$06 | $400D | 2 | 8B06 | IMM, 2B |
| STAA R2 | $400F | 3 | 7A10 03 | IMM, 3B |

# 3. Assignment Questions

1. Find the status of CCR flags - C, V, H, N, Z and Acc A after executing A, B, C and D individually and comment on the results.

Carry + overflow + zero result

Carry + neg. no.

Carry + overflow

Neg. no.

1

1

1

1

1

1

1

80

2E

A5

1

0

0

0

0

0

0

0

0

0

0

0

0

0

2. Write an instruction sequence to multiply two 8-bit numbers using MUL instruction (unsigned). Initialize Accumulator A and Accumulator B with $F4 and $F2 respectively using the load instructions. Highlight the result in the D register and the CCR flags.

zero result

Nothing

Overflow

Carry + neg. no.

0

0

0

0

1

1

1

1

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

0

****

4. Conclusions and Recommendations

To conclude, the fundamental instructions that are used in the assembly language are understood and manipulated easily. Two basic programs have been programmed and tested in simulation and practice. The students are now able to edit, develop and debug any instructions set given the proper purpose of the program. All of this has been tested on a HCS12 microcontroller. Also, we understand the mechanism by which microcontroller deal with registers. To finish off, a set of useful information of the instruction set were generated using the .LST file.