Microprocessor Systems Laboratory (ELCE333)

LaboratoryExperimentNo.1

MARKINGSHEET

Student Names and ID No's: **Amal AlQasimi 100036830**
Shaikha Albesher 100037064

Laboratory Section: **Wednesday** **Experiment Date:** **January 28, 2015**

No.	Criteria	Description	Weight %	Mark	Comments
1	Pre-lab	A mark will be allocated to each student that reflects his preparations for the lab.	20		
2	Performance In the lab	A mark will be allocated to each student individually that reflects his performance in the lab	10		
3	Results and Analysis	Documentation and analysis of the results for each task performed in the lab	30		
4	Summary/ Conclusions	Conclusions for each task performed in the lab	10		
5	Assignment Questions	Answers to assignment questions	20		
6	Report Presentation	Overall presentation of the report including proper layout and clarity of figures, tables, and graphs. Correct use of English language.	10		
	Total		**100**		

KHALIFA UNIVERSITY

ELCE 333: Microprocessor Systems Laboratory

Lab Report- ExperimentNo.1

Experiment Title: Microcontroller Assembly Program Development

Completed by:

Amal Al Qasimi (100036830)

Shaikha Al Besher (100037064)

Lab instructors:

Mahmoud Khonji

Mohammed Al Zaabi

Spring 2015

Table of Contents

Summary

The following lab report will include the details of the first Microprocessor System's Laboratory lab session. It will first start off by stating the aims and objectives of the tasks assigned to us throughout the lab session. Following the aims and objectives, the report will discuss the purpose of each task separately with the results obtained upon the completion of the task. Based on the observations of the tasks completed, the analysis and interpretation section will include the explanation of why we obtained the results we did. Finally the report will end with a brief conclusion. The assignment questions given to us as part of the lab are answered after the lab repot.

1. Aims and Objectives:

The following are the aims and objectives of the first Microprocessor Systems report:

Aim: The aim of this experiment is to familiarize students with the process of editing, assembling, simulating, downloading and subsequently executing an assembly program.

Upon the completion of this experiment we were expected to be able to:

Objectives:

1- Edit an HCS12 assembly program.
2- Assemble a program to generate a binary file and a list file.
3- Download the program binary file to the DragonPlusTrainer board.
4- Use CodeWarrier to single-step through the program.
5- Trace a program execution and check the system registers.
6- Use the HCS12 Instruction Set Manual to understand a given program and modify it as required.
7- Understand the content of the program list file.

2. Introduction

The aim of this lab was to introduce us to the layout and structure of the assembly language programs and the instructions and format. By performing operating such as creating projects, entering assembly language programs, assembling, simulating, and debugging, this lab overall helped us familiarize us with the use of CodeWarrior software. In general, the lab will focused on performing simple operations such as the basic data transfer instructions like move, transfer and exchange. In addition to the assembly arithmetic instructions such as addition and subtraction, we were also introduced to the status flags of the Condition Codes Register.

3. Lab tasks

1. TASK-1: Adding Numbers

The first task started off by creating a project by means of the CodeWarrier software and compiling the following code:

```
;Includederivative-specificdefinitions
        INCLUDE'derivativ
                e.inc'
;exportsymbols
        XDEFEntry
;Inserthereyourdatadefiniti
on. SUMAEQU$1002
SUMBEQU$1003
;codesection
    ORG$4000
Entry:CL
      RA
      LDAA#$55; A=$55
      LDAB#$25;B=$25
      ABA;Addreg.Btoreg
      A STAASUMA
      ADDA#$10
      ADDA#$06
      STAASUMB
HEREJMPHERE
```

We then assembled and simulated the program by single stepping through it and then recorded the content of the registersaftereachstepinTable1 below:

Table 1: Registers contents while single stepping the program

Instruction	A	B	D	Comments
CLRA	00	CB	00CB	Clear accumulators A
CLRB	00	00	0000	Clear accumulator B
LDAA #$55	55	00	5500	Load accumulator A with the value $55
LDAB #$25	55	25	5525	Load accumulator B with the value $25
ABA	7A	25	7A25	Add A to B, save the result in A
STAA SUMA	7A	25	7A25	Store the value of A at location $1002
ADDA #$10	8A	25	8A25	Add the value $10 to A
ADDA #$06	90	25	9025	Add the value $06 to A
STAA SUMB	90	25	9025	Store the value of A in location $1003

After the program finished we inspected the content s of memory locations $1002 and $1003 to be 7A and 90.

Table 2 : The contents of memory locations

	$1002	$1003	Comments
Before Simulating	--	--	Both memory location are empty
At the end of the program	7A	90	Address $1002 is loaded with the addition of A&B , address $10003 is loaded with the final contents of A.

Conclusion:

The program performs an addition operation, where the contents of A & B are added and the result is stored in register A. The addition result is stored in location $1002. Another two Hex values are added to the contents of A, and stored in location $1003.

2. TASK-2: Subtraction Numbers

In the second task, we modified the code used in task 1 in order to subtract the byte in A from the byte in B and store the result in memory locations $1003 and$1004. SUB command is used to perform the subtraction.

The modified code is:

```
; Include derivative-specific definitions
        INCLUDE 'derivative.inc'
; export symbols
        XDEF Entry
; Insert here your data definition.
SUMA EQU $1003
SUMB EQU $1004
; code section
    ORG $4000
Entry: CLRA
    CLRB
    LDAA #$55 ; A=$55
    LDAB #$25 ; B=$25
    EXG A,B;
    SBA ; Sub A from B
    STAA SUMA
STAA SUMB
HERE JMP HERE
```

The contents of the registers are changing after each line, the following table shows the contents of the registers:

Table 3 : Registers contents while single stepping

Instruction	A	B	D	Comments
CLRA	00	CB	00CB	Clear accumulators A
CLRB	00	00	0000	Clear accumulator B
LDAA #$2	02	00	0200	Load accumulator A with the value 2
LDAB #$5	02	05	0205	Load accumulator A with the value 5
STAA MEMO1	02	05	0205	Store the value of A in MEMO1
SUBB MEMO1	02	03	0203	Subtract the value of MEMO1 from B
STAB RESULT1	02	03	0203	Store the result of the subtraction in RESULT1

And the content of the memory locations $1003 and $1004 are tested and results are shown in table below:

Table 4: The contents of the memory locations

$1001	$1002	Comments
02	03	The content of register A will appear at address 1001 and the final result after the subtraction will appear in address 1002

Conclusion:

Subtracting the contents of A from B doesn't have a direct instruction, so the value of A is saved in the memory location $1001, and then it is subtracted from the contents of B. The operation is as follows:

A=(MEMO1)
B=B-(MEMO1)
B→(RESULT1)

3. TASK-3: Simple Program

In this task list file (.LST) is used in order to develop the program by checking and correcting errors. The following table illustrates the memory contents for the developed program.

Table 5: The memory contents for the developed program

Instruction	Start Address	No. of Bytes	Instruction Code	Comment
CLRA	4000	1	87	Clear Acc A
CLRB	4001	1	C7	Clear Acc B
LDAA #$55	4002	2	8655	$55 - > A
LDAB #$25	4004	2	C625	$25 - > B
ABA	4006	2	1806	A + B - > A
STAA SUMA	4008	3	7A10	A - > $1002
ADDA #$10	400B	2	8B10	A + $10 - > A
ADDA#$06	400D	2	8B06	A + $06 - > A
STAA SUMB	400F	3	7A10	A - > $1003

The program starts form address 4000. The number of bytes shows the difference between the start address of the wanted instruction and the following instruction, for instance 4001-4000=1.

4. ANALYSIS AND INTERPRETATION

Laboratory experiment 1 consists of 3 tasks:

1- Creating a project in CodeWarrier to learn the steps of the Addition operation using Assembly code, understanding the code and how it impacts each step, and find the information or data in memory.
2- Subtraction, to perform this operation the information has to be saved in the main memory, which increases the number of steps. Then it follows task 1 steps.
3- List File is used, which allow testing the Start Address, No. of Bytes, Instruction Code and other features of the code.

5. CONCLUSION AND RECOMMENDATIONS

In this laboratory experiment CodeWarrior software was used to execute some Assembly instructions.

Upon lab completion we achieved the objectives of the experiment and become familiar with editing, assembling, simulating, downloading and subsequently.

As well as, we gained the ability to identify the characteristics of List File (.LST).

6. ASSIGNMENT QUESTIONS

1)

	Instructions	C	V	H	N	Z	Acc A	Comment	
A	**LDAA #$A5**				√		A5	$A5 - > A	Negative
	ADDA #$89	√	√				2E	A +$89 - > A	Carry + Over flow
B	**LDAA #$80**	√			√		80	$80 - > A	Carry + Negative
	ADDA #$80	√	√			√	0	A + $80 - > A	Carry + O.V + Zero
C	**LDAB #$A5**	√			√		0	$A5 - > A	Carry + Negative
	SUBB #$68		√				0	B - $68 - > B	Oveflow
D	**LDAB #$65**						0	$65 - > B	
	SUBB #$65					√	0	B – $65 - > B	Zero

2)

Check with :	Instr	C	V	H	N	Z	Acc D	Comment
LDAA #$2 LDAB #$3	MUL						6	2 * 3 = 6 In decimal or Hex
LDAA #$F4 LDAB #$F2	MUL	√			√		E6A8	F4 * F2 = E6A8 In Hex

```
            ABSENTRY Entry
            INCLUDE 'derivative.inc'
SUMA        EQU $1002
SUMB        EQU $1003
;   code section
            ORG $4000
Entry:
            LDAA #$F4
            LDAB #$F2
            MUL
HERE        JMP     HERE
```

References

1. CourseTextBook
2. http://www.evbplus.com/
3. MC9S12DP256User'sManual