ECE 441

Microprocessors

Instructor: Dr. Jafar Saniie

Teaching Assistant: Guojun Yang

Final Project Report:

**MONITOR PROJECT**

04/24/17

By: Javier Sorribes

Acknowledgment: I acknowledge all of the work including figures and codes are belongs to me and/or persons who are referenced.

Signature : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Table of Contents

Abstract 2

1-) Introduction 2

2-) Monitor Program 3

2.1-) Command Interpreter 4

2.1.1-) Algorithm and Flowchart 5

2.1.2-) 68000 Assembly Code 5

2.2-) Debugger Commands 6

2.2.1-) Debugger Command #1: HELP 6

2.2.2-) Debugger Command #2: MDSP 7

2.2.3-) Debugger Command #3: SORTW 7

2.2.4-) Debugger Command #4 8

2.2.5-) Debugger Command #5 8

2.2.6-) Debugger Command #6 9

2.2.7-) Debugger Command #7 10

2.2.8-) Debugger Command #8 10

2.2.9-) Debugger Command #9 11

2.2.10-) Debugger Command #10 11

2.2.11-) Debugger Command #11 12

2.2.12-) Debugger Command #12: EXIT 12

2.2.12-) Debugger Command #13: 12

2.2.12-) Debugger Command #14: CONV 12

2.3-)Exception Handlers 13

2.3.1-) Bus Error Exception 13

2.3.2-) Address Error Exception 14

2.3.3-) Illegal Instruction Exception 14

2.3.4-) Privilege Violation Exception 14

2.3.5-) Divide by Zero Exception 15

2.3.6-) Line A and Line F Emulators 15

2.4-)User Instructional Manual Exception Handlers 16

2.4.1-) Help Menu 16

3-) Discussion 17

4-) Feature Suggestions 18

6-) Conclusions 19

7-) References 20

Abstract

The summary of your design should go here. Someone who reads this abstract should have a clear understanding of your design and the overall flow of the report.

# *1-) Introduction*

This will be an introduction to your design. You can give design objectives, a clear description of the problem and design methodology and technology used. Any figures and tables should have clear descriptions.



*Figure 1.1. Monitor command interpreter block diagram*

***2-) Monitor Program***

A clear description of your design should be given here - what this program will do, requirements, etc. You may include a block diagram or table.



*Figure 2.1. Monitor program*

***2.1-) Command Interpreter***

A clear description of your design should be given here.

***2.1.1-) Algorithm and Flowchart***

An algorithm of the design and its flowchart will be explained here. You may need to put in comments for your algorithm.

*Clear //this where things starts*

*Do this m=0 // assign m*

*While m > n // while m > n*

*If m > n //*

*Do this //*

*Else //*

*Do these more //*

*End if //*

*m = m + 1 // increment m by 1*

*finish // finish*

*Figure 2.2. Command Interpreter Algorithm*

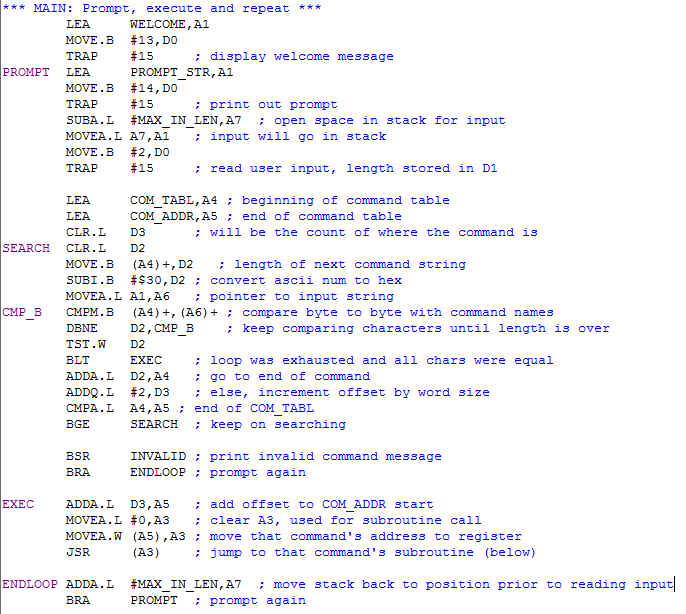
It may be necessary to explain more about your flowchart and your design ideas.



***2.1-)***

*Figure 2.3. Command Interpreter Flowchart*

***2.1.2-) Command Interpreter Assembly Code***



*Figure 2.4. Main & Command Interpreter 68000 Assembly Code*

***2.2-) Debugger Commands***

A clear description of your design should be given here.

***2.2.1-) Debugger Command #1: HELP***

A clear description of this debugger command should be given here.

***2.2.1.1-) Debugger Command #1 Algorithm and Flowchart***

An algorithm of the design and its flowchart will be explained here. You may need to include comments for your algorithm.

*Clear //this where things starts*

*Do this m=0 // assign m*

*While m > n // while m > n*

*If m > n //*

*Do this //*

*Else //*

*Do these more //*

*End if //*

*m = m + 1 // increment m by 1*

*finish // finish*

*Figure 2.5. Debugger Command # 1 Algorithm*

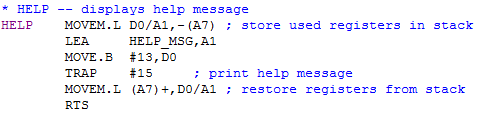
It may be necessary to explain more about your flowchart and your design ideas.



***2.1-)***

*Figure 2.6. Debugger Command # 1 Flowchart*

***2.2.1.2-) Debugger Command #1 Assembly Code***



*Figure 2.7. Debugger Command #1 Assembly Code*

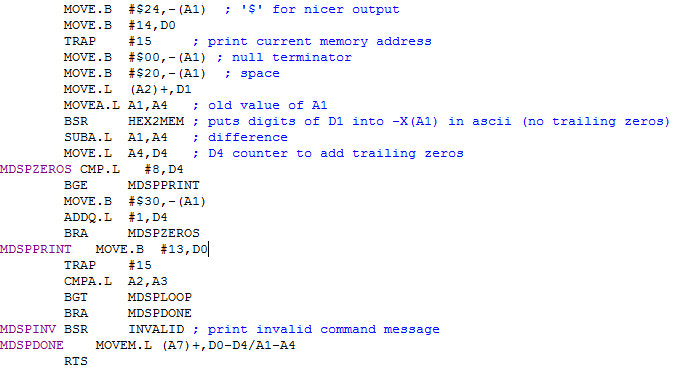
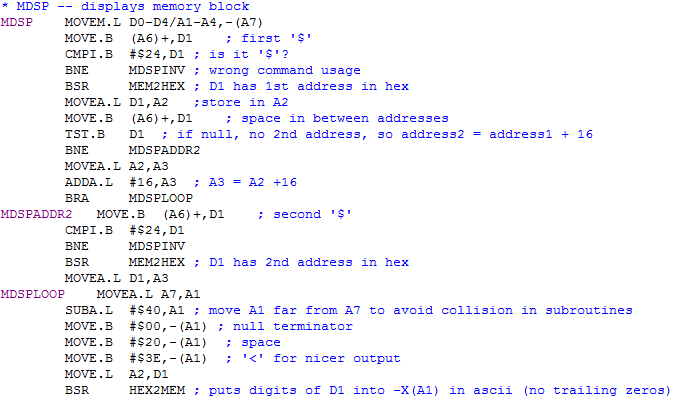
***2.2.2-) Debugger Command #2: MDSP***

It is similar to 2.2.1

***2.2.2.1-) Debugger Command #2 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.2.2-) Debugger Command #2 Assembly Code***



***2.2.3-) Debugger Command #3: SORTW***

Sorts a block of memory in between addresses 1 and 2 (inclusive) in either ascending or descending order. The command should be called in the form “SORTW <address1> <address2> A|D”, where A refers to ascending and D to descending (default).

The size of each number within the memory specified is expected to be word, and the type unsigned.

***2.2.3.1-) Debugger Command #3 Algorithm and Flowchart***

The algorithm for sorting is based on Bubble Sort, a method to “bubble up” items to their correct locations. By comparing numbers to the adjacent ones, we can decide whether to swap these or continue. Please refer to Lab Manual 2, Procedure 2.5 for more details.

In addition, a small check was implemented to be able to do either ascending or descending order as requested by the user.

*SORTW // first line*

*Parse input to get ‘start’, ‘end’ and ‘type’ (A or D)*

*While start < end // start will serve as an incrementing pointer*

*If start < start+1 and type = A // using start as an address pointer*

*Swap start with start+1 // so start+1 is the item after start*

*Reset start to original value (start over)*

*Else if start > start+1 and type = D*

*Swap start with start+1*

*Reset start to original value*

*Else // order is fine, move on to next*

*start = start + 1*

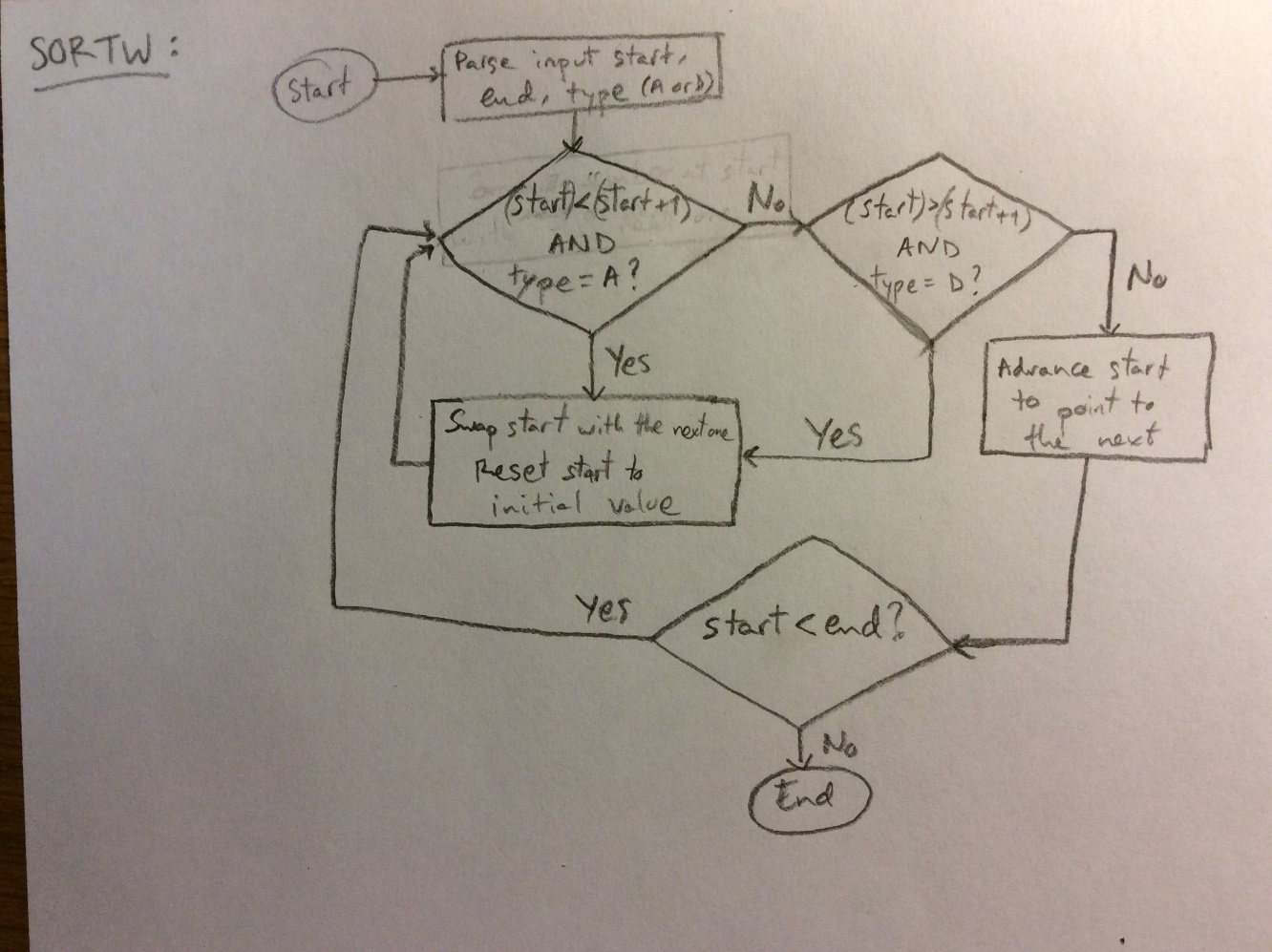
*End if*

*End while*

*Finish // finish*

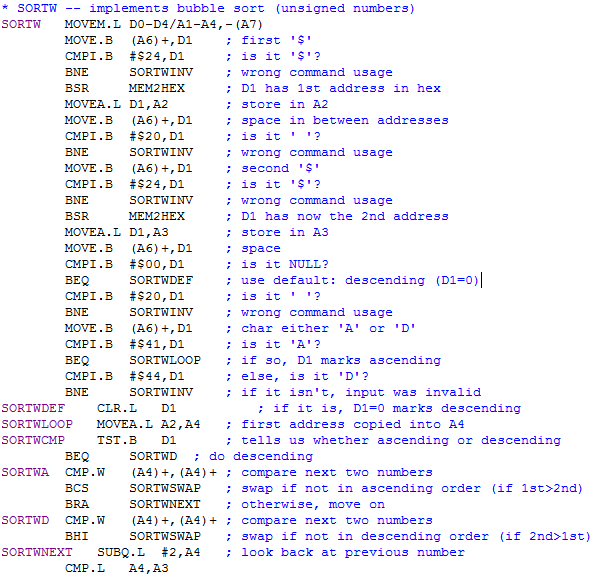
*Figure 2.X. Debugger Command #3 Algorithm*

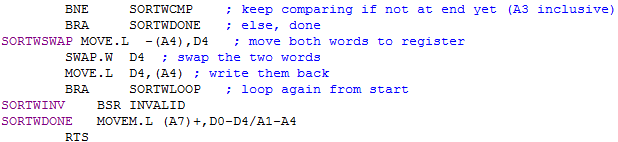
The following flowchart is an abstraction of the algorithm described above:



*Figure 2.X. Debugger Command #3 Flowchart*

***2.2.3.2-) Debugger Command #3 Assembly Code***





***2.2.4-) Debugger Command #4***

It is similar to 2.2.1

***2.2.4.1-) Debugger Command #4 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.4.2-) Debugger Command #4 Assembly Code***

It is similar to 2.2.1.2

***2.2.5-) Debugger Command #5***

It is similar to 2.2.1

***2.2.5.1-) Debugger Command #5 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.5.2-) Debugger Command #5 Assembly Code***

It is similar to 2.2.1.2

***2.2.6-) Debugger Command #6***

It is similar to 2.2.1

***2.2.6.1-) Debugger Command #6 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.6.2-) Debugger Command #6 Assembly Code***

It is similar to 2.2.1.2

***2.2.7-) Debugger Command #7***

It is similar to 2.2.1

***2.2.7.1-) Debugger Command #7 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.7.2-) Debugger Command #7 Assembly Code***

It is similar to 2.2.1.2

***2.2.8-) Debugger Command #8***

It is similar to 2.2.1

***2.2.8.1-) Debugger Command #8 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.8.2-) Debugger Command #8 Assembly Code***

It is similar to 2.2.1.2

***2.2.9-) Debugger Command #9***

It is similar to 2.2.1

***2.2.9.1-) Debugger Command #9 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.9.2-) Debugger Command #9 Assembly Code***

It is similar to 2.2.1.2

***2.2.10-) Debugger Command #10***

It is similar to 2.2.1

***2.2.10.1-) Debugger Command #10 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.10.2-) Debugger Command #10 Assembly Code***

It is similar to 2.2.1.2

***2.2.11-) Debugger Command #11***

It is similar to 2.2.1

***2.2.11.1-) Debugger Command #11 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.11.2-) Debugger Command #11 Assembly Code***

It is similar to 2.2.1.2

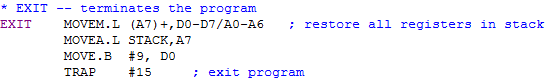
***2.2.12-) Debugger Command #12: EXIT***

It is similar to 2.2.1

***2.2.12.1-) Debugger Command #12 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.12.2-) Debugger Command #12 Assembly Code***



***2.2.13-) Debugger Command #13***

It is similar to 2.2.1

***2.2.13.1-) Debugger Command #13 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.13.2-) Debugger Command #13 Assembly Code***

It is similar to 2.2.1.2

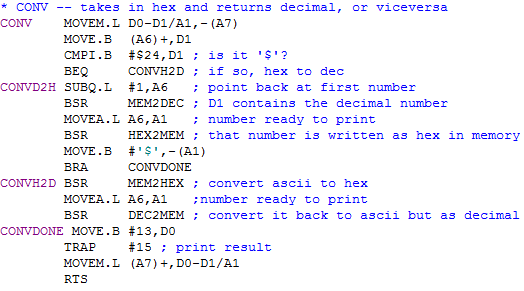
***2.2.14-) Debugger Command #14: CONV***

It is similar to 2.2.1

***2.2.14.1-) Debugger Command #14 Algorithm and Flowchart***

It is similar to 2.2.1.1

***2.2.14.2-) Debugger Command #14 Assembly Code***



***2.3-) Exception Handlers***

Brief information about Exception Handlers should be given here.

***2.3.1-) Bus Error Exception***

A clear description of this debugger command should be given here.

***2.3.1.1-) Bus Error Exception Algorithm and Flowchart***

An algorithm of the design and its flowchart will be explained here. You may need to add comments for your algorithm.

*Clear //this where things starts*

*Do this m=0 // assign m*

*While m > n // while m > n*

*If m > n //*

*Do this //*

*Else //*

*Do these more //*

*End if //*

*m = m + 1 // increment m by 1*

*finish // finish*

*Figure 2.8. Debugger Command # 1 Algorithm*

It may be necessary to explain more about your flowchart and your design ideas.



*Figure 2.9. Debugger Command # 1 Flowchart*

***2.3.1.2-) Bus Error Exception Assembly Code***

The assembly code should be written using the algorithm above.

*ORG $1000*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*END ;*

*Figure 2.10. Debugger Command # 1 Assembly Code*

***2.3.2-) Address Error Exception***

It is similar to 2.3.1

***2.3.1.1-) Bus Error Exception Algorithm and Flowchart***

It is similar to 2.3.1.1

***2.3.1.2-) Bus Error Exception Assembly Code***

It is similar to 2.3.1.2

***2.3.3-) Illegal Instruction Exception***

It is similar to 2.3.1

***2.3.3.1-) Illegal Instruction Exception Algorithm and Flowchart***

It is similar to 2.3.1.1

***2.3.3.2-) Illegal Instruction Exception Assembly Code***

It is similar to 2.3.1.2

***2.3.4-) Privilege Violation Exception***

It is similar to 2.3.1

***2.3.4.1-) Privilege Violation Exception Algorithm and Flowchart***

It is similar to 2.3.1.1

***2.3.4.2-) Privilege Violation Exception Assembly Code***

It is similar to 2.3.1.2

***2.3.5-) Divide by Zero Exception***

It is similar to 2.3.1

***2.3.5.1-) Divide by Zero Exception Algorithm and Flowchart***

It is similar to 2.3.1.1

***2.3.5.2-) Divide by Zero Exception Assembly Code***

It is similar to 2.3.1.2

***2.3.6-) Line A and Line F Emulators***

It is similar to 2.3.1

***2.3.6.1-) Line A and Line F Emulators Algorithm and Flowchart***

It is similar to 2.3.1.1

***2.3.6.2-) Line A and Line F Emulators Assembly Code***

It is similar to 2.3.1.2

***2.4-) User Instructional Manual Exception Handlers***

Brief information about Instructional Manual Handlers should be given here.

***2.4.1-) Help Menu***

A clear description of this debugger command should be given here

***2.4.1.1-) Algorithm and Flowchart***

An algorithm of the design and its flowchart will be explained here. You may need to include comments for your algorithm.

*Clear //this where things starts*

*Do this m=0 // assign m*

*While m > n // while m > n*

*If m > n //*

*Do this //*

*Else //*

*Do these more //*

*End if //*

*m = m + 1 // increment m by 1*

*finish // finish*

*Figure 2.11. Debugger Command # 1 Algorithm*

It may be necessary to explain more about your flowchart and your design ideas.



*Figure 2.12. Debugger Command # 1 Flowchart*

***2.4.1.2-) Assembly Code***

The assembly code should be written using the algorithm above.

*ORG $1000*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*\*\*\*\*\*\*\*\*\* // Comment goes here*

*END ;*

*Figure 2.13. Debugger Command # 1 Assembly Code*

***3-) Discussion***

Design challenges and discussion about this project should be provided here. You may subdivide this section further and supply figures and table if necessary.

***4-) Feature Suggestions***

You may suggest ideas for expanding this project, such as exception handlers, etc. You may subdivide this section further and supply figures and tables if necessary.

***5-) Conclusion***

The conclusion goes here.[1]

***6-) References***

[1] T. Harman and D. Hein, “The Motorola MC 68000 Microprocessor Family”, Prentice-Hall Inc., Englewood Cliffs, NJ, 1996.

[2] A. Clements, “Microprocessor Systems Design”, PWS Publishing Company, Boston, MA, 1997.

[3] Experiment 2 Lab Manual

[4] Educational Computer Board Manual