CSCI 592

LAB ASSIGNMENT – 4

Written by

DINESH SEVETI

Date: 02-22-2025

**OBJECTIVE:**

The objective of this lab assignment is to display a vector using assembly language programming for the 68K microprocessor. The program manipulates memory addresses and registers to move data efficiently.

**TECHNOLOGY USED**

* Easy68K Assembler software to run the code.
* Hypothetical or real CPU with registers and memory
* Hexadecimal Memory Addressing

**PROCEDURE**

* Define the memory addresses and load data into registers.
* Use the LEA instruction to load effective addresses into registers.
* Perform byte-level data movement using MOVE.B instructions.
* Store the manipulated values in the designated memory locations.
* Execute the program in a simulator to observe the results.

**OPERATIONS**

* Loading Addresses: LEA.L is used to load effective addresses into address registers.
* Byte Movement: MOVE.B transfers byte values from one memory location to another.
* Post-Increment and Pre-Decrement: Registers are modified accordingly to adjust data storage locations.
* Halt Execution: The program halts after execution to display results.

**ALGORITHM**

* Load memory locations into address registers A1, A2, A3, and A4.
* Retrieve a byte from the source vector (A1) and store it in D0.
* Move bytes from one address (A2) to another (A3) while reversing order.
* Store the retrieved value into another address (A4).
* Repeat the process for all vector elements.
* End execution.

**CODE LISTING**

ORG $1500

START: ; first instruction of program

LEA.L $004AC4, A1

LEA.L $004ACE, A2

LEA.L $004AE0, A3

LEA.L $004AEA, A4

MOVE.B (A1)+, D0

MOVE.B -(A2), (A3)+

MOVE.B D0, -(A4)

MOVE.B (A1)+, D0

MOVE.B -(A2), (A3)+

MOVE.B D0, -(A4)

MOVE.B (A1)+, D0

MOVE.B -(A2), (A3)+

MOVE.B D0, -(A4)

MOVE.B (A1)+, D0

MOVE.B -(A2), (A3)+

MOVE.B D0, -(A4)

MOVE.B (A1)+, D0

MOVE.B -(A2), (A3)+

MOVE.B D0, -(A4)

; halt simulator

\* Variables and Strings

ORG $004AC4

DC.L 'ABCDEFGHIJ'

END START ; last line of source

**DESCRIPTION**

This program initializes memory addresses and processes a vector by reading and writing data using assembly language instructions. It moves bytes between registers, ensuring correct data manipulation.

**OBSERVATIONS**

* The program successfully loads and modifies vector elements.
* The order of values is reversed when moved between memory locations.
* The MOVE.B instruction efficiently transfers bytes.

**RESULTS**

**LOG FILES**

Before Execution:

|  |  |
| --- | --- |
| **Memory Address** | **Value** |
| $004AC4 | A |
| $004AC5 | B |
| $004AC6 | C |
| $004AC7 | D |
| $004AC8 | E |
| $004AC9 | F |
| $004ACA | G |
| $004ACB | H |
| $004ACC | I |
| $004ACD | J |

After Execution:

|  |  |
| --- | --- |
| **Memory Address** | **Value** |
| **$004AE0** | **J** |
| **$004AE1** | **I** |
| **$004AE2** | **H** |
| **$004AE3** | **G** |
| **$004AE4** | **F** |
| **$004AE5** | **E** |
| **$004AE6** | **D** |
| **$004AE7** | **C** |
| **$004AE8** | **B** |
| **$004AE9** | **A** |

**CONCLUSIONS**

This lab assignment demonstrates memory manipulation techniques in 68K assembly language. The use of effective addressing modes, register operations, and byte transfers enables efficient vector handling. The program successfully achieves its objective of displaying and modifying a vector using assembly instructions.