CSCI 320-54 – Assignment 5

Created by: Thomas Hoerger

Objectives

This program inserts a new element into a linked list.

Equipment Used

EASY 68K simulator

Procedure

Construct a memory image of Figure 7.1. Load A5 with the address $0074B0 (do not do this in the code, do this before you run your program). Insert the following code which establishes pointers to e2 and e3:

```
LEA    $0074A8,A6   .A6 points to e1
MOVE.L       4(A6),A1       .A1 points to e2
MOVE.L       4(A1),A2       .A2 points to e3
```

```
LEA.L   $000074B0,A5
MOVE.L  A2,4(A5)
MOVE.L  A5,4(A1)
```

Insert the element that A5 points to between those two elements. You must turn in a copy of the linked list (memory display) prior to and after inserting the new element. You are also required to turn in a copy of your instructions for this operation.

New Operations Learned

MOVE.L
DC.L
LEA.L
LEA

Program Description

This program demonstrates the insertion of a new element into a linked list. It initializes memory for four elements (e1, e2, e3, e4) with respective data values and pointers to the next element. Then, it initializes memory for the new element with its data value and updates the pointer of e3 to point to the new element, effectively inserting it between e2 and e3 in the linked list.

SOURCE CODE

```
*-----------------------------------------------------------
* Title      : Linked List Insertion Program Lab 5
* Written by : Thomas Hoerger
* Date       : 3/30/2024
* Description: This program inserts a new element into a linked list.
*-----------------------------------------------------------
    ORG    $1000

START:
    ; Initialize memory for e1
    MOVE.L  #data1,D0     ; Load data for e1 into D0
    MOVE.L  D0,$0074A8    ; Store data for e1 at address $0074A8
    MOVE.L  #e2,D0        ; Load address of e2 into D0
    MOVE.L  D0,$0074AC    ; Store address of e2 (next pointer of e1) at address $0074AC

    ; Initialize memory for e2
    MOVE.L  #data2,D0     ; Load data for e2 into D0
    MOVE.L  D0,$0074D0    ; Store data for e2 at address $0074D0
    MOVE.L  #e3,D0        ; Load address of e3 into D0
    MOVE.L  D0,$0074D4    ; Store address of e3 (next pointer of e2) at address $0074D4

    ; Initialize memory for e3
    MOVE.L  #data3,D0     ; Load data for e3 into D0
    MOVE.L  D0,$0074B8    ; Store data for e3 at address $0074B8
    MOVE.L  #e4,D0        ; Load address of e4 into D0
    MOVE.L  D0,$0074BC    ; Store address of e4 (next pointer of e3) at address $0074BC

    ; Initialize memory for the new element
    MOVE.L  #data_new,D0  ; Load data for the new element into D0
    MOVE.L  D0,$0074B0    ; Store data for the new element at address $0074B0
    MOVE.L  #e3,D0        ; Load address of e3 into D0
    MOVE.L  D0,4+$0074B0 ; Store address of e3 (next pointer of the new element) at offset 4 from $0074B0

    ; Update the next pointer of e2 to point to the new element
    MOVE.L  #e2,D0        ; Load address of e2 into D0
    MOVE.L  $0074B0,D1    ; Load address of the new element into D1
    MOVE.L  D1,4(A0)      ; Store address of the new element (next pointer of e2) at offset 4 from e2

    SIMHALT               ; halt simulator

* Put variables and constants here
data1:    DC.L   41414141   ; data for e1
data2:    DC.L   42424242   ; data for e2
data3:    DC.L   43434343   ; data for e3
data4:    DC.L   44444444   ; data for e4
data_new: DC.L   45454545   ; data for the new element
e2:       DC.L   $0074D0    ; address of e2
e3:       DC.L   $0074B8    ; address of e3
e4:       DC.L   $0074C0    ; address of e4

    END    START              ; last line of source
```

Figure 1 shows the code properly entered in the simulator.

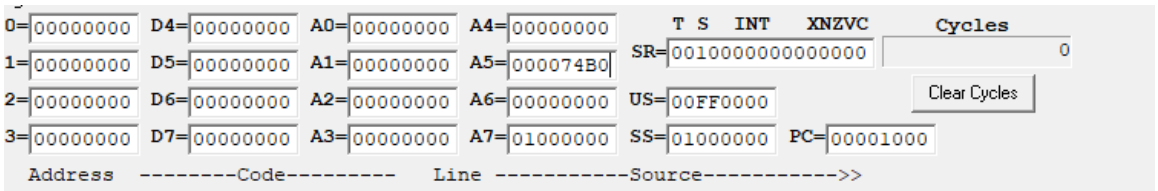

Figure 2 shows the A5 register loaded with the address $0074B0 before running.

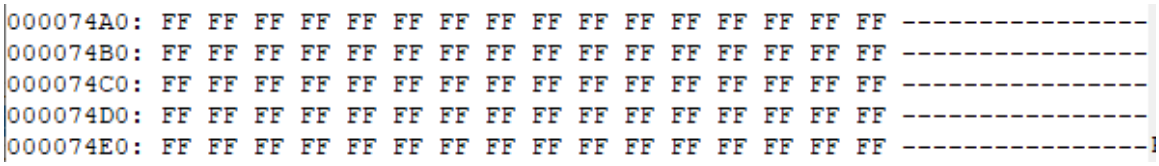

Figure 3 shows the memory window before execution.

Results

After Execution of the program:

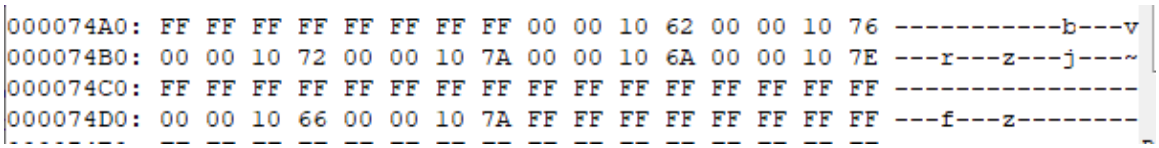

Figure 4 shows the memory window after running the program.

I am having a problem with the memory window. I'm unsure if I got it correct after execution. Could you give me some advice about how I can do this better or what I can do to fix it. I'm not sure what it's supposed to look like after execution. Could you give me an example maybe?