COEN - 210

COMPUTER ARCHITECTURE

ASSEMBLY CODE

Team Members

1. Dhanasekar Ravi Jayanthi (SCU ID : 07700005982)
2. Manasa Madiraju (SCU ID : 07700009942)
3. Pujitha Kallu (SCU ID : W1653660)
4. Rohit Roy Chowdhury (SCU ID :7700000162)

Guided By : Prof. Yan Cui

Problem Statement

Use the instructions in the SCU ISA to write two versions of the assembly program for finding the maximum in *n* numbers described below.
(1) MAX = max {a1, a2, a3,....an}
a.The first version (software loop) does not use the MAX instruction
b.The second version (hardware loop) uses the MAX instruction.

Version 1 : (Software Loop)

Register Assumptions:

x0 - Constant 0 ‘Zero’ value
x5 - Address of the 1st element index in array
x7 - First element of an array or current index [i]
x8 - Length of an array [n]
x10 - Next element in the iteration
x11 - Maximum number in the array
x15 - Current maximum
x14 - Address of the current element in an array
x15, x16, x17, x18, x19 - Temporary registers to store values

CODE:

```
1.  ADD x7, x7, x0          // Add  x7 and  x0 ,store result in x7 (Assigning value i)
2.  INC x8, x8, -1          // Decrement the value in x8 by 1 and store in x13 (n-1)
3.  BRZ x0                  // branch to exit if n==0;
4.  ADD x14, x5, x0         // Add x5 and x0 and store the result in x14
5.  LD x15, x14             // Load the value from memory location pointed  by x14 into x15
6.  SVPC x16, 52            // Save the program counter (PC) value plus 52 into x16
7.  SVPC x17, 40            // Save the PC value plus 40 into x17
8.  SVPC x18, 4             // Save the PC value plus 4 into x18
9.  SUB x19, x7, x8         // Subtract  x8 from x7 and store the result in x19
10. BRZ x16                 // Branch to the address in x16 if value in x19 is zero
11. INC x14, x14, 4         // Increment the value in x14 by 4
12. INC x7, x7, 1           // Increment the value in x7 by 1
13. LD x10, x14             // Load the value from the memory location x14 into x10
14. SUB x11, x15, x10       // Subtract  x10 from x15 and store the result in x11
15. BRN x17                 // Branch to the address in x17 if value in x11 is negative
16. J x18                   // Jump to the address stored in x18
17. ADD x15, x10, x0        // Add x10 to x0 and store the result in x15
18. J x18                   // Unconditional jump to the address stored in x18
19. NOP                     // No-Operation, EXIT
```

Example:

M = [5,6,9,8]
Addresses = [1000,1004,1008,1012]
n = 4

Explanation :

In summary, the code iterates through an array element [n], maintaining a maximum value by comparing it with each element in the array. It updates the maximum value when it encounters a value greater than the current maximum. The code uses conditional branches to control the loop and update the maximum value as needed.

Version 2 : (Hardware Loop)

Register Description:

x1 - stores the current max value
x3 - Size of the input array, counter
x4 - Address of the size of the array

CODE:

```
1. LD     x3, x4            // load the size of array from x4 to x3
2. INC    x4, x4, 0x4       // move current position to the first element of an array
3. MAX    x1, x4, x3        // find maximum element of the array using MAX
4. NOP                      // Exit
```

Explanation :

In summary, the code is to load the size of an array, set the current position to the first element of the array, and then potentially find the maximum element within the array using custom "MAX" instruction.

Flow Diagram :