**Computer Organization and Architecture (EET2211)**

**LAB IX: Determine the largest and smallest number in an array**

**Siksha ‘O’ Anusandhan Deemed to be University, Bhubaneswar**

|  |  |  |  |
| --- | --- | --- | --- |
| **Branch: Section:** | | | |
| **S. No.** | **Name** | **Registration No.** | **Signature** |
| 52 | Saswat Mohanty | 1941012407 | **D:\Pics and Sign\sign.jpg** |

**Marks: \_\_\_\_\_\_/10**

**Remarks:**

**Teacher’s Signature**

**I. OBJECTIVE:**

1. Write a program to determine the largest number in an array.
2. Write a program to determine the smallest number in an array.

**II. PRE-LAB**

**For Obj. 1:**

1. **Determine the largest number in an array.**

[1500h] = 03h

[1501h] = 13h

[1502h] = 22h

[1503h] = 11h

Output: 22h

1. **Write the assembly code.**

|  |
| --- |
| **org 100h**  **mov ax,0000h**  **mov ds,ax**  **mov si,1500h**  **mov di,1510h**  **mov cl,[si]**  **inc si**  **mov al,[si]**  **dec cl**  **l1: inc si**  **mov bl,[si]**  **cmp al,bl**  **jnc again**  **mov al,bl**  **again: dec cl**  **jnz l1**  **mov [di],al**  **hlt**  **ret** |
|  |

**For Obj. 2:**

1. **Determine the smallest number in an array.**

[1500h] = 03h

[1501h] = 13h

[1502h] = 22h

[1503h] = 11h

Output: 03h

1. **Write the assembly code.**

|  |
| --- |
| **org 100h**  **mov ax,0000h**  **mov ds,ax**  **mov si,1500h**  **mov di,1510h**  **mov cl,[si]**  **inc si**  **mov al,[si]**  **dec cl**  **l1: inc si**  **mov bl,[si]**  **cmp al,bl**  **jc again**  **mov al,bl**  **again: dec cl**  **jnz l1**  **mov [di],al**  **hlt**  **ret** |
|  |

**III. LAB:**

**Assembly Program:**

**For Obj. 1:**

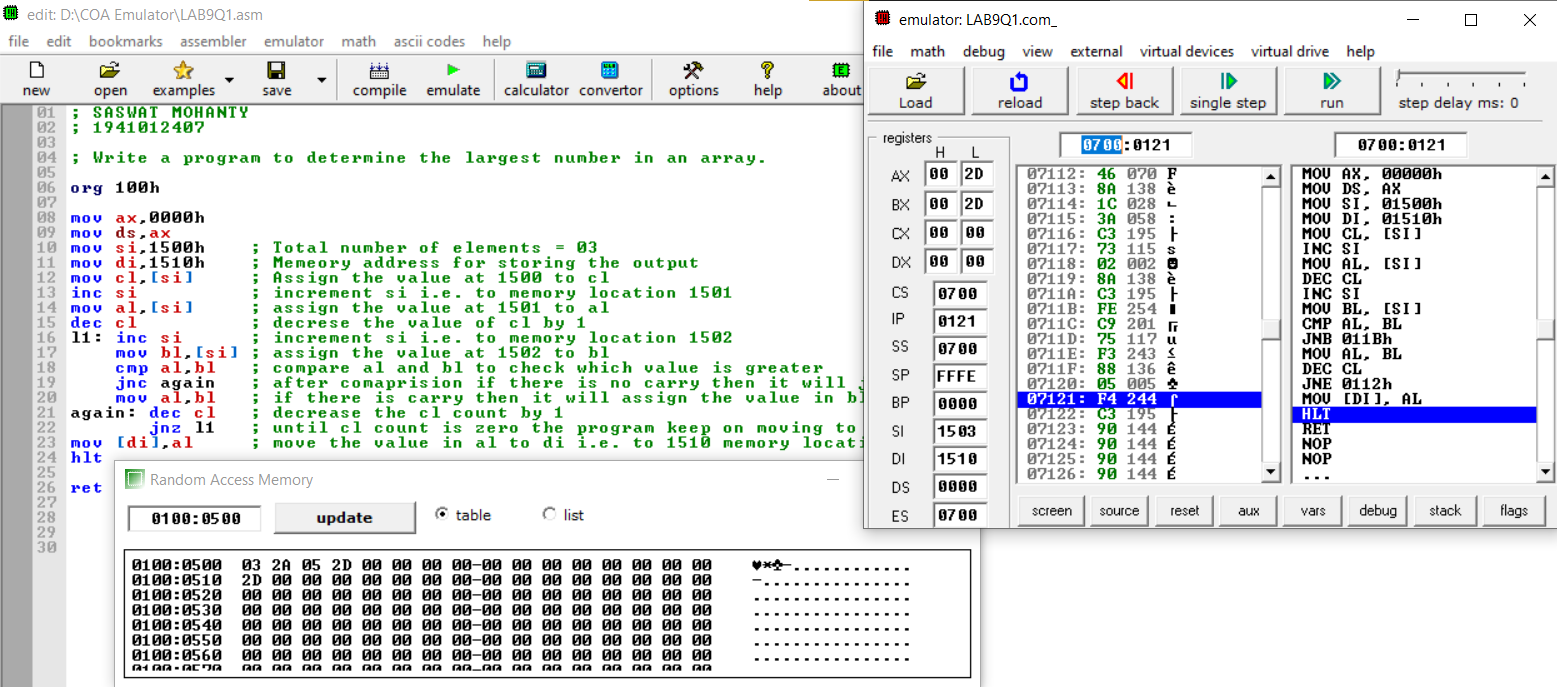
|  |
| --- |
| **; SASWAT MOHANTY**  **; 1941012407**  **; Write a program to determine the largest number in an array.**  **org 100h**  **mov ax,0000h**  **mov ds,ax**  **mov si,1500h ; Total number of elements = 03**  **mov di,1510h ; Memeory address for storing the output**  **mov cl,[si] ; Assign the value at 1500 to cl**  **inc si ; increment si i.e. to memory location 1501**  **mov al,[si] ; assign the value at 1501 to al**  **dec cl ; decrease the value of cl by 1**  **l1: inc si ; increment si i.e. to memory location 1502**  **mov bl,[si] ; assign the value at 1502 to bl**  **cmp al,bl ; compare al and bl to check which value is greater**  **jnc again ; after comaprision if there is no carry then it will jump to the "again" pointer**  **mov al,bl ; if there is carry then it will assign the value in bl to al**  **again: dec cl ; decrease the cl count by 1**  **jnz l1 ; until cl count is zero the program keep on moving to "l1" pointer**  **mov [di],al ; move the value in al to di i.e. to 1510 memory location**  **hlt**  **ret** |

**For Obj. 2:**

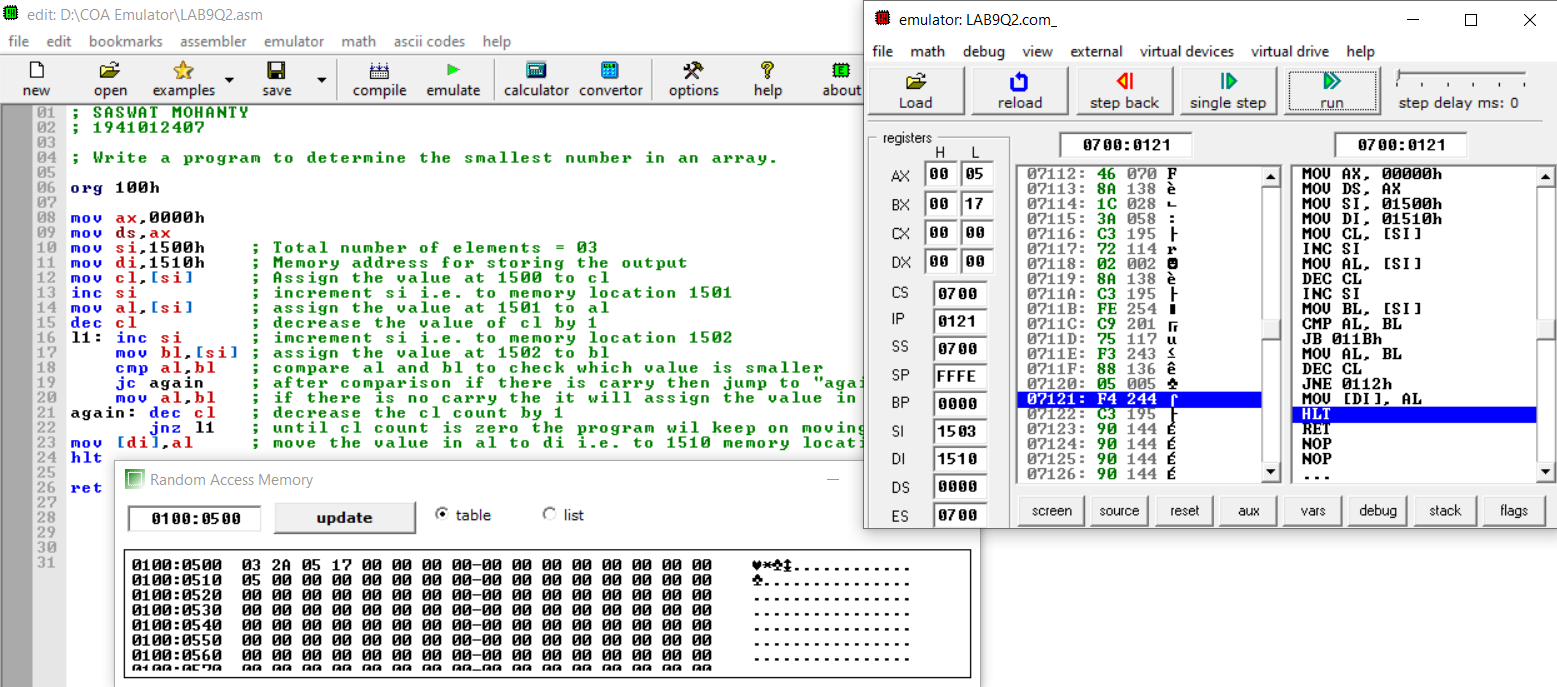
|  |
| --- |
| **; SASWAT MOHANTY**  **; 1941012407**  **; Write a program to determine the smallest number in an array.**  **org 100h**  **mov ax,0000h**  **mov ds,ax**  **mov si,1500h ; Total number of elements = 03**  **mov di,1510h ; Memory address for storing the output**  **mov cl,[si] ; Assign the value at 1500 to cl**  **inc si ; increment si i.e. to memory location 1501**  **mov al,[si] ; assign the value at 1501 to al**  **dec cl ; decrease the value of cl by 1**  **l1: inc si ; imcrement si i.e. to memory location 1502**  **mov bl,[si] ; assign the value at 1502 to bl**  **cmp al,bl ; compare al and bl to check which value is smaller**  **jc again ; after comparison if there is carry then jump to "again" pointer**  **mov al,bl ; if there is no carry the it will assign the value in bl to al**  **again: dec cl ; decrease the cl count by 1**  **jnz l1 ; until cl count is zero the program wil keep on moving to "l1" pointer**  **mov [di],al ; move the value in al to di i.e. to 1510 memory location**  **hlt**  **ret** |

**Observations (with screen shots):**

**For Obj. 1:**

****

**For Obj. 2:**

****

**Conclusion:**

**For Obj. 1:**

It can be concluded to determine the largest number in an array when dry run and executed in system found to be same. Thus, the program to determine the largest number in an array was executed.

**For Obj. 2:**

It can be concluded to determine the smallest number in an array when dry run and executed in system found to be same. Thus, the program to determine the smallest number in an array was executed.

**IV. POST LAB:**

**What is ARM processor?**

An ARM processor is one of a family of CPUs based on the RISC (reduced instruction set computer) architecture developed by Advanced RISC Machines (ARM).

**Differentiate between ARM processor and RISC.**

|  |  |
| --- | --- |
| **ARM** | **RISC** |
| ARM is proprietary. | RISC is open-source. |
| ARM makes 32-bit and 64-bit RISC multi-core processors. | RISC processors are designed to perform a smaller number of types of computer instructions so that they can operate at a higher speed, performing more millions of instructions per second (MIPS). |
| ARM has added more complex instructions to increase processor performance (at the expense of higher power consumption). | RISC approach is more successful in reducing overall power consumption, sometimes at the expense of lower performance. |

**Differentiate between ARM processor and 8086.**

|  |  |
| --- | --- |
| **ARM** | **8086** |
| Integrated in designs which were manufactured on 28, 16, 14 or 10 nanometer FinFET nodes | Manufactured on a 3-micron process |
| RICS Design | CISC Design |
| Consists of a front end, back end (execution engine) and an un-core memory subsystem which includes the L2 cache. | Consists of two main blocks, the BIU and EU |

**Differentiate between ARM processor and microcontroller.**

ARM is core for both microprocessor and micro-controller. ARM is based on CPU architecture so we generally call it has microprocessor when placed on a chip if ARM is combined with memories (RAM and ROM) on a single chip we can call it has micro-controller it has limited memory but when coming to microprocessor RAM and ROM are connected externally speed will be more.

**List few applications of ARM processor-based system.**

* ARM processor features include:
* Load/store architecture.
* An orthogonal instruction set.
* Mostly single-cycle execution.
* Enhanced power-saving design.
* 64 and 32-bit execution states for scalable high performance.
* Hardware virtualization support.