BECE204P-Microprocessors & Microcontrollers Lab

LAB-4

SMALLEST AND LARGEST, ASENDING, DECENDING OF AN ARRAY

LAB TASK-1

SMALLEST NUMBER IN AN ARRAY

- Write 8086 Assembly language program to find smallest number in an array.

Problem Analysis:

1. Let the array of numbers be "N" bytes
2. Assign AL register to store the smallest number
3. Assume first number of the array is the smallest number and it is saved in AL register
4. Then compare each byte of the array with the AL register
5. After each comparison, store the smallest data among the two in AL register
6. After N-1 comparison, the AL register hold the smallest number in an array

LAB TASK-1

SMALLEST NUMBER IN AN ARRAY

Algorithm:

1. Load the starting address of the array in SI register
2. Load the address of the result in DI register
3. Load the number of byte in the array on CL register
4. Increment the array pointer (SI register)
5. Get the first byte of the array in AL register
6. Decrement the byte count (CL register)
7. Increment the array pointer (SI register)
8. Get next byte of the array in BL register
9. Compare current smallest data (AL) and next byte (BL) of the array
10. Check carry flag. If carry flag is set then go to step 12, otherwise go to next step
11. Move BL to AL
12. Decrement the byte count (CL register)
13. Check zero flag, if Z=0 then go to step 7 otherwise go to next step
14. Save the smallest data in AL register into memory pointed by DI register
15. Stop

LAB TASK-1

SMALLEST NUMBER IN AN ARRAY

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 1100H	Load the starting address of the array in SI register
	MOV DI, 1200H	Load the address of the result in DI register
	MOV CL,[SI]	Load the number of byte in the array on CL register
	INC SI	Increment the array pointer (SI register)
	MOV AL,[SI]	Get the first byte of the array in AL register
	DEC CL	Decrement the byte count (CL register)
AGAIN:	INC SI	Increment the array pointer (SI register)
	MOV BL,[SI]	Get next byte of the array in BL register
	CMP AL,BL	Compare current smallest data (AL) and next byte (BL) of the array
	JC AHEAD	Check carry flag. If C=1 then go to step 12, otherwise go to next step
	MOV AL,BL	Move BL to AL
AHEAD:	DEC CL	Decrement the byte count (CL register)
	JNZ AGAIN	Check zero flag, if Z=0 then go to step 7 otherwise go to next step
	MOV [DI],AL	Save the smallest data in AL register into memory pointed by DI register
	HLT	Stop

Input:

ADDRESS	VALUE
1100H	05H
1101H	44H
1102H	22H
1103H	33H
1104H	11H
1105H	55H

Output:

ADDRESS	VALUE
1200H	11H

LAB TASK-2

LARGEST NUMBER IN AN ARRAY

- Write 8086 Assembly language program to find largest number in an array.

Problem Analysis:

1. Let the array of numbers be "N" bytes
2. Assign AL register to store the largest number
3. Assume first number of the array is the largest number and it is saved in AL register
4. Then compare each byte of the array with the AL register
5. After each comparison, store the largest data among the two in AL register
6. After N-1 comparison, the AL register hold the largest number in an array

LAB TASK-1

LARGEST NUMBER IN AN ARRAY

Algorithm:

1. Load the starting address of the array in SI register
2. Load the address of the result in DI register
3. Load the number of byte in the array on CL register
4. Increment the array pointer (SI register)
5. Get the first byte of the array in AL register
6. Decrement the byte count (CL register)
7. Increment the array pointer (SI register)
8. Get next byte of the array in BL register
9. Compare current largest data (AL) and next byte (BL) of the array
10. Check carry flag. If carry flag is reset then go to step 12, otherwise go to next step
11. Move BL to AL
12. Decrement the byte count (CL register)
13. Check zero flag, if Z=0 then go to step 7 otherwise go to next step
14. Save the largest data in AL register into memory pointed by DI register
15. Stop

LAB TASK-1

LARGEST NUMBER IN AN ARRAY

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 1100H	Load the starting address of the array in SI register
	MOV DI, 1200H	Load the address of the result in DI register
	MOV CL,[SI]	Load the number of byte in the array on CL register
	INC SI	Increment the array pointer (SI register)
	MOV AL,[SI]	Get the first byte of the array in AL register
	DEC CL	Decrement the byte count (CL register)
AGAIN:	INC SI	Increment the array pointer (SI register)
	MOV BL,[SI]	Get next byte of the array in BL register
	CMP AL,BL	Compare current largest data (AL) and next byte (BL) of the array
	JNC AHEAD	Check carry flag. If C=0 then go to step 12, otherwise go to next step
	MOV AL,BL	Move BL to AL
AHEAD:	DEC CL	Decrement the byte count (CL register)
	JNZ AGAIN	Check zero flag, if Z=0 then go to step 7 otherwise go to next step
	MOV [DI],AL	Save the largest data in AL register into memory pointed by DI register
	HLT	Stop

Input:

ADDRESS	VALUE
1100H	05H
1101H	44H
1102H	22H
1103H	33H
1104H	11H
1105H	55H

Output:

ADDRESS	VALUE
1200H	55H

LAB TASK-3

SORTING AN ARRAY IN ASCENDING ORDER

➢ Write 8086 Assembly language program to sort an array of data in ascending order.

Problem Analysis:

1. The array can be sorted in an ascending order by bubble sorting.
2. In bubble sorting of N-data, N-1 comparisons are performed by taking two consecutive data at a time
3. After each comparison, the two data can be rearranged in an ascending order in the same memory location
4. When N-1 comparisons are performed N-1 times, the array will be sorted in the ascending order

LAB TASK-3

SORTING AN ARRAY IN ASCENDING ORDER

Algorithm:

1. Set SI register as a pointer for array
2. Set CL register as count for N-1 comparisons (outer loop)
3. Initialize the SI register as array pointer
4. Set CH as count for N-1 comparisons (inner loop)
5. Increment the array pointer
6. Get the first element of array in AL register
7. Increment the array pointer
8. Compare the next element of the array with AL
9. Check carry flag, If C=1 goto step 11 otherwise goto next step
10. Exchange the content of memory pointed by SI and the content of previous memory location (AL to SI content and AL to SI-1 content)
11. Decrement the count for comparisons (CH register)
12. Check Zero flag, IF =0 goto step 6 otherwise go to next step
13. Decrement the count for repetitions (CL register)
14. Check the zero flag, if Z=0 go to step 3 otherwise goto next step.
15. Stop

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 1100H	Set SI register as a pointer for array
	MOV CL, [SI]	Set CL register as count for N-1 comparisons (outer loop)
	DEC CL	
REPEAT:	MOV SI, 1100H	Initialize the SI register as array pointer
	MOV CH, [SI]	
	DEC CH	Set CH as count for N-1 comparisons (inner loop)
	INC SI	Increment the array pointer
REPCOM:	MOV AL, [SI]	Get the first element of array in AL register
	INC SI	Increment the array pointer
	CMP AL, [SI]	Compare the next element of the array with AL
	JC AHEAD	Check carry flag, If C=1 goto AHEAD otherwise goto next step
	XCHG AL, [SI]	Exchange the content of memory pointed by SI and the content of previous memory location i.e AL
	XCHG AL, [SI-1]	Exchange the content of memory pointed by SI-1 and the content of previous memory location i.e AL
AHEAD:	DEC CH	Decrement the count for comparisons (CH register)
	JNZ REPCOM	Check Zero flag, IF =0 goto REPCOM otherwise go to next step
	DEC CL	Decrement the count for repetitions (CL register)
	JNZ REPEAT	Check the zero flag, if Z=0 go to REPEAT otherwise goto next step
	HLT	Stop

Before Execution:

ADDRESS	VALUE
1100H	05H
1101H	44H
1102H	22H
1103H	33H
1104H	11H
1105H	55H

After Execution:

ADDRESS	VALUE
1100H	05H
1101H	11H
1102H	22H
1103H	33H
1104H	44H
1105H	55H

LAB TASK-4

SORTING AN ARRAY IN DECENDING ORDER

➢ Write 8086 Assembly language program to sort an array of data in descending order.

Problem Analysis:

1. The array can be sorted in an descending order by bubble sorting.
2. In bubble sorting of N-data, N-1 comparisons are performed by taking two consecutive data at a time
3. After each comparison, the two data can be rearranged in an descending order in the same memory location
4. When N-1 comparisons are performed N-1 times, the array will be sorted in the descending order

LAB TASK-4

SORTING AN ARRAY IN DECENDING ORDER

Algorithm:

1. Set SI register as a pointer for array
2. Set CL register as count for N-1 comparisons (outer loop)
3. Initialize the SI register as array pointer
4. Set CH as count for N-1 comparisons (inner loop)
5. Increment the array pointer
6. Get the first element of array in AL register
7. Increment the array pointer
8. Compare the next element of the array with AL
9. Check carry flag, If C=0 goto step 11 otherwise goto next step
10. Exchange the content of memory pointed by SI and the content of previous memory location (AL to SI content and AL to SI-1 content)
11. Decrement the count for comparisons (CH register)
12. Check Zero flag, IF =0 goto step 6 otherwise go to next step
13. Decrement the count for repetitions (CL register)
14. Check the zero flag, if Z=0 go to step 3 otherwise goto next step.
15. Stop

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 1100H	Set SI register as a pointer for array
	MOV CL, [SI]	Set CL register as count for N-1 comparisons (outer loop)
	DEC CL	
REPEAT:	MOV SI, 1100H	Initialize the SI register as array pointer
	MOV CH, [SI]	
	DEC CH	Set CH as count for N-1 comparisons (inner loop)
	INC SI	Increment the array pointer
REPCOM:	MOV AL, [SI]	Get the first element of array in AL register
	INC SI	Increment the array pointer
	CMP AL, [SI]	Compare the next element of the array with AL
	JNC AHEAD	Check carry flag, If C=0 goto AHEAD otherwise goto next step
	XCHG AL, [SI]	Exchange the content of memory pointed by SI and the content of previous memory location i.e AL
	XCHG AL, [SI-1]	Exchange the content of memory pointed by SI-1 and the content of previous memory location i.e AL
AHEAD:	DEC CH	Decrement the count for comparisons (CH register)
	JNZ REPCOM	Check Zero flag, IF =0 goto REPCOM otherwise go to next step
	DEC CL	Decrement the count for repetitions (CL register)
	JNZ REPEAT	Check the zero flag, if Z=0 go to REPEAT otherwise goto next step
	HLT	Stop

Before Execution:

ADDRESS	VALUE
1100H	05H
1101H	44H
1102H	22H
1103H	33H
1104H	11H
1105H	55H

After Execution:

ADDRESS	VALUE
1100H	05H
1101H	55H
1102H	44H
1103H	33H
1104H	22H
1105H	11H