NAME FAYYAZ MUHAMMAD

ID 14294

Department BS(CS)

Semester 5th

Teacher Sir Amin

Subject Microprocessor AND Assembly language.

QUESTION NO :- (01)

Solve each of the following

1) $(64)_{10} = (?)_2$

2	64
2	32 — 0
2	16 — 0
2	8 — 0
2	4 — 0
2	2 — 0
	1 — 0

=> $(1000000)_2 = (64)_{10}$ Ans.

2) $(01111111)_2 = (?)_{10}$

$(01111111)_2 = (?)_{10}$

$\Rightarrow (0\times2)^7 \times (1\times2)^6 + (1\times2)^5 + (1\times2)^4 + (1\times2)^3 + (1\times2)^2 + (1\times2)^1 + (1\times2)^0$

$\Rightarrow 0 + 64 + 32 + 16 + 8 + 4 + 2 + 1$

$\Rightarrow (127)_{10}$ Ans.

3) $(407F)_{16} = (?)_{10}$

$\Rightarrow 4\times16^3 + 0\times16^2 + 7\times16^1 + F\times16^0$

$\Rightarrow 4\times4096 + 13\times256 + 7\times16 + 15\times1$

$\Rightarrow 16384 + 3328 + 112 + 15$

$\Rightarrow (19839)_{10}$ Ans.

4) $(128)_{10} = (?)_{16}$

$(128)_{10} = (?)_{16}$

16	128
16	8 - 0

$(80)_{16}$ Ans

5) $(3A6F)_{16} = (?)_2$

$(3A6F)_{16} = (?)_2$

first convert all to binary (4 bits)

3 A 6 F

0011 1010 0110 1111

$\Rightarrow (0011\ 1010\ 0110\ 1111)_2 = (3A6F)_{16}$

6) $(11\ 00\ 00\ 11\ 11\ 00\ 101)_2 = (?)_{16}$

$(1100\ 0011\ 1110\ 0101)_2 = (?)_{16}$

(12 3 14 5)

(C 3 E 5)

$\Rightarrow (C3E5)_{16}$ Ans.

7) $(-16)_{10} = (?)_2$

$(-16)_{10} = (?)_2$

As the decimal integer is negative So the MSB in Binary will be 1.

Now Converting 16 into Binary.

2	16
2	8 — 0
2	4 — 0
2	2 — 0
	1 — 0

=> 00010000

its the +ve 16 representation taking 2's complement.

=> 00010000

11101111

+ 1

11110000

$(-16)_{10} = (11110000)_2$ Ans.

8) $(01111111)_2 - (00000111)_2$ (2's complement)

1st Complement of 00000111 = 11111000

Now add 01111111

11111000

(1)01110111

Carry

Now add this carry to answer.

01110111

+ 1

$(01111000)_2$

$(01111000)_2$ Ans.

9) $(6D)_{16} - (3F)_{16}$

$(6D)_{16} = 01101101$

$(3F)_{16} = 0011\ 1111$

Taking 1st Complement of 2nd Number

0111111

$1000000 \Rightarrow$ 1st complement

Now add both numbers.

$$\begin{array}{r} 1101101 \\ +\ 1000000 \\ \hline (1)\,0101101 \end{array}$$

(1) ← Carry

add carry in result.

$$\begin{array}{r} 0101101 \\ +\ 1 \\ \hline (0101110)_2 \text{ Ans.} \end{array}$$

$\Rightarrow (2E)_{16}$ Ans.

10) $(11111111)_2 = \pm(?)_{10}$

$(11111111)_2 = \pm(?)_{10}$

Signed integer as MSB is "1" which show number is negative

=> 11 11 11 11

=> $1\times2^7+1\times2^6+1\times2^5+1\times2^4+1\times2^3+1\times2^2+1\times2^1+1\times2^0$

=> $1\times128+1\times64+1\times32+1\times16+1\times8+1\times4+1\times2+1\times1$

=> 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1

=> - 1 A

QUESTION NO 2

Write short note on each of the following :-

1) Embedded Systems :-

Embedded Systems is a combination of Software and Hardware. which is designed to perform a particular task that has to be completed in a given time.

For Example :-

=> Mobile phone
=> Washing machine
=> Micro oven.

2) DEVICE DRIVE :-

Device drive are programs that translate general operating System command into specific reference to hardware detail that only the

manufacture Knows Drives are hardware dependent and operating System specific A device drive is a file that lets the computer Know the configuration and specification of a contains hardware device.

3) Virtual Machine Concept :-

A virtual machine (VM) is a software program or Operating System that not only exhibits the behavior of a seperate computer, but is also capable of performing task such as running application and program like a seperate computer.

4) Instruction execution Cycle.

There three execution Cycle.

1) fetch

2) decode

1) execute.

5) Motherboard chipset

The chipset is the "glue" that connect the microprocessor to the rest of the motherboard and therefore to the rest of the computer, on a pc it consists of two basic part the north bridge and the southbridge. All of the various component of the computer communicate with the CPU through the chipset.

6) Access level for input-output operation.

In computer input/output or I/O (or, informally, io or IO) is the communication b/w an information processing system such as a computer and the outside world, possible human or another information processing system input are the signals or data received by the system and output are the signals or data sent from it. The team can also be used as part of an action; to "perform I/O" is to perform an input or output.

7) Basic part of assembly language instructions:

There are four basic part of instruction.

[label :] mnemonic [operands]

[;comment]

QUESTION NO: 3

* Differentiate between each of the following

1) Assembly language and high-level language.

An Assembly language directly control the physical hardware. A high level language is much more obstract; which must be compiled / translated is a step up from this. High level language cut the middle-man between human thinking and machine language.

2) Protected mode and real address mode.

Real mode:

Real mode is an operational mode that allows newer intel 286 processor to acquire the characteristic of the

reduced 8086 or 8088 processors. Real model provide a higher clock speed but only restricts that processor to a 16-bit and a minimum of 1mb RAM instruction.

Protected mode:-

The protected mode is a 32-bit operating mode that is identified on intel 80286 or later on. This mode allow the termination of a failed program without restarting the computer or running programs it offer access to addressing by virtual memory expanded memory, and multiple task while it protecting program against memory over-writing.

3) Assembler and linker:

The main output produce by assembler an input assembly language source file is the translation of the file into an object file in (ELF). ELF files produced by the assembler are relocatable files that hold code and or data they are input file for linker.

4) Instruction and directive?

INSTRUCTION:

An instruction is a task to be carried out by the processor at run time. Instruction are assembled into machine code and eventually linked into the final executabled.

DIRECTIVE:

A directive is an instruction to the assembler telling it how to treat the data it is asked to assemble. Directives only used at assembly time and although they may affect the way the code is generated, they don't result in any code generation themselves.

5) Code label and ~~to label~~ data label?

Code label:-

Code label is the label that we have on code as we see in case of condition jump (label 11) and its is normally used for loop control statement.

Data label:-

Data label is the label that we use to define data as we define memory location. num 1, num 2 ..... etc in our programs.

6) Link Comment and block comment?

LINK COMMENT:-

A Single line comment and as implied only applied to a Single line in the "Source code" (the program).

BLOCK COMMENT:-

A block comment and refers usually refers to a paragraph of text. A block comment has a start Symbol and an end Symbol and everything between is ignored by the computers.

7) Equal-Sign directive and EQU directive?

EQUAL-SIGN DIRECTIVE:-

The Equal-Sign directive associates a Symbol name with an integer expression.

This Syntax is

name = Expression

Expression is a 32-bit integer.

- May be redefined.
- Name is called a Symbolic Constant.

EQU DIRECTIVE:

Define a Symbol as either an integer or text expression.

- Cannot be redefined
- The EQU directive give a Symbolic name to a numeric constant a register-relative value or a PC-relative value.

QUESTION NO - 4

Give an answer to each of the following

1) Explain the concept of portability as it applies to programming languages.

A language whose Source program can be compiled and run on a wide variety of computer System is said to be portable.

2) Why would a high-level language not be an ideal tool for writing a program that directly accesses a particular brand of printer?

A high-level language may not be provide for direct hardware access. Even if it does awkward coding techniques possible maintance problem.

3) Why was Unicode invented?

Unicode is a character encoding Standard that has widespread acceptance. They store letter and other characters by assigning a number for each one. Before Unicode was invented there was hundreds of different encoding system for assigning these numbers. No single encoding could contain enough characters.

4) If W = 11101100, X = 00010011, and Y = 00111100, then find Z = W ∨ X ∧ ¬ Y.

W = 11101100
X = 00010011
Y = 00111100

Sols.

Z = W ∨ X ∧ ¬ Y

Y = 00111100
¬Y = 11000011

=> X ∧ ¬Y = 00010011
- 11000011
AND 00000011 Ans

=> W ∧ X ∧ ¬Y:

11101100
00000011 OR.
11101111

5) Create a truth table to show all possible inputs and outputs for the Boolean function described by ¬(AVB)

¬(AVB

Create table

A	B	¬A	¬AVB
F	T	T	T
F	T	T	T
T	F	F	F
T	F	F	T

6) Why does memory access take more machine cycles than register access?

Conventional memory is outside the CPU and it responds more slowly to access request. Register are hard-wired inside the CPU.

7) Discuss the basic program execution registers used in x86 32-Bit processors.

XMM Registers:-

The x86 architecture also contain eight 128-bit register called XM registers they are used by Streaming SIMD extension to the instruction.

MMx Registers:

MMx technology improves the performance of intel processor when implementing advanced multimedia and communication application. The 64 bit MMx register support special instruction called SIMD (Single-instruction Multiple-DATA)

Segment Register:

In real address mode, 16-bit Segment register indicate base addresses of preassigned memory areas named Segments.

Basic Programming Execution Register:

Registers are high-speed storage location directly inside the CPU designed to be accessed at much higher speed than conventional memory. When a processing loop is optimized for speed.

QUESTION NO: 5

Discuss the following MASM directives in detail:

INCLUDE. 386 MODEL STACK

PROTO DATA CODE PROC

ENDP END

386 :-

```
model flat.
Stack 4096
Exit Process PROTO
dw Exit code: DWORD
```

The .386 directive identifies it as a 32-bit program Lin 2 uses the flat memory model, and windows requires the Model conversation to be used

Line 3 set a side 4096 bytes of storage Line 4 declares a Proto type for the Exit process function.

MODEL:-

This tells the assembler which memory model to use In 32-bit program. we use the flat memory model. which is associated with the processor's protected mode.

STACK :

The stack direction tells how many bytes of memory to reserve for the runtime stack.
4096 happen to correspond to the size of a memory page in this processor system for managing memory.

PROTO :.

The PROTO directive by using the INVOKE directive.

DATA :-

. DATA directive creates a near data segment.

. This segment contain the frequency used to data for your program.

. DATA Segment can occupy

up to 64K in MS-DOS
or up to 512 mega byte
under flat model in windows NT.

CODE :

Code
main PROC

it is the beginning of the code area of a program (meaning what's afterward

is using usually the main procedure.

PROC:-

MASM start and end of a procedure block called label.

Syntax:

label PROC [distance] [language_type]

[PUBLIC | PRIVATE | EXPORT]

[< Prologuearg >]

[user regist] [parameter [: tag]]

[FRAME [: ehandler - address]]

Statement

label END.

ENDP:-

Marks the end of procedure name previously begun with PROC

Syntax:

name ENDP

Remarks:

See PROC

END :-

- Directive for END of files command. That's END of file. Here as you are using "main" you have to end it with.
- Simply know that the Assembler need END directive to end the file END can be written without Main just end the file with END.
- Now ENDP denoted END of procedure here procedure id "main" so before ending the file, END the "main" procedure you have to end the procedure.

QUESTION No: 6

1) Write a program that calculates the following expression: A=(A+B)-(C+D).

A = (A+B) - (C+D)

```
.data : data Segment, read and write
var A BYTE 10
var B BYTE 20
var C BYTE 30
var D BYTE 40

Final val BYTE ?

.code : code Segment, read_only
    main PROC

    MOV a1, var A
    MOV b1, var B
    MOV c1, var C
    MOV d1, var D

    ADD a1, b1 ; Compute (A+B)
    ADD c1, d1 ; Compute (A+B)
    SUB a1, c1 ; Compute (A+B)-(C+D)
    Mov final val, al

    Call Dump Regs
    exit
main ENDP
END main
```

2) Show the order of individual bytes in memory for the following doubleword variable using little endian order dval DWORD 12345678h

When placed in memory at offset 0000, 78h would be stored in the first byte 56h would be stored in the second byte and the remain bytes would be at offset 0002 and 0003 as show in figure 3.14

Little-endian representation of 1,2,3,4,5,6,7,8h

0000	78
0001	56
0002	34
0003	12

3) Write a statement that causes the assembler to calculate the number of bytes in the following string, and assign the value to a symbolic constant named StringSize.

String 1 byte "Assembly language is easy", 0

- data

String byte "Assembly language is easy", 0
String Size byte ?

- Code

mov eax SizeE of string 1
mov string size, eax

4) Write a program that performs arithmetic operations on different register operands and stores the result in memory. Give stepwise explanation of each statement.

Let the Arithematic operation is
$x = -a + (b - c)$

```
INCLUDE     32 : inc.

• data
    x DWORD ? ; Uninitialized variable.
    a DWORD 10 ; initialize variable 'a'
    b DWORD 26 ; initialize variable 'b'
    C DWORD 15 ; initize variable 'C'

• Code

    main PROC
    mov   eax, a    ; move, value of 'a' into 'eax'
    neg   eax       ; EAX = -10
    mov   ebx, b    ; move value of 'b' into 'ebx'
    Sub   ebx, c    ; EBX = 11
    add   eax, ebx  ; perform -10+11
    mov   x, eax    ; 1

    call Dump legs

    exit

    main ENDP

    END main
```