IQRA NATIONAL UNIVERSITY
PESHAWAR

NAME	SHARIQ
ID	13698
SESSION	SUMMER
SUBJECT	ASSEMBLY LANGUAGE
SUBMITTED to	SIR. AMIN
DATE	18/08/2020

Question # 1

(a) $64_{10} = (1\square\square\square\square\square\square)_2$

(b) $(\square1111111)_2 = (127)_{10}$

(c) $4D7F = (19839)_{10}$

(d) $(128)_{10} = (80)_{16}$

(E) $3G6F1G = (0\square1110\square\square11\square1111)_2$

8	4	2	1	
1	1	1	1	F
0	1	1	0	6
1	0	1	0	A
0	0	1	1	3

(F) $(110\square\square\square1111\square\square101)_2 = (C3E5)_{16}$

8	4	2	1	
□	1	□	1	5
1	1	1	0	E
□	□	1	1	3
1	1	□	□	C.

$= (C3E5)_{16}$

(G) $(11111111)_2 = \pm(255)_{10}$

(H) $-1G_{10} = (-10\square\square\square)_2$

(i) $(\square11111111)_2 - (\square\square\square\square\square111)_2$

$= (11110\square\square)_2$

(j) $6076 - 3F16 = (2E)_{10}$

Question #2.

Embedded System

An embedded system is a combination of computer hardware and software designed for a specific function or a function within a large system.

The system can be programmable or with fixed functionality.

Device drivers

More commonly known as driver, a device driver or hardware driver is a group of files that enable one or more hardware devices to communicate with the computer's operating system. Without drivers, the computer would not be able to send and receive data correctly to hardware devices, such as printer.

Virtual machine concept

Virtual machine is an image file managed by the hypervisor that exhibits the behavior of a seperate computer, capable of performing task such as running application and programs like a seperate computer.

Instruction execution Cycle

A single machine instruction does not just magically execute all at once. The CPU has to go through a predefine sequence of steps to execute a machine instruction called the instruction execute cycle. Lets assume the instruction pointer register holds the address of instruction we want to execute. Here are the steps to execute it.

One Cycle:

- First the CPU has to fetch the instruction from an area of memory called the instruction queue. Right after doing this, it increments the instruction pointer.
- Next, the CPU decode the instruction by looking at its binary bits pattern. This bit pattern might revealed that the instruction has operande
- if operande are involved, the CPU fetch the operand from registers and memory. Sometimes this involve address calculation.
- The CPU executes the instruction using any oprands values it fetches during earlier step. it also updates a few status flags such as Zero, carry and overflow

- Finally if an output operand was part of the instruction, the CPU stores the result of its execution in the operand.

Motherboard Chipest.

A motherboard chipest is a collection of processor chips designed to work together on a specific type of motherboard.

Various Chipest have features that increase processing power, multimedia capabilities or reduce power consumtion.

Access level for input-output operation

Application programe routinely read input from keyboard and disk files and write output to the screen and to files I/O need, you not be accomplished by directly accessing hardware instead, you can call function provided by the operating system.
I/O is available at different access level similar to the virtual machine concept shown is. There are three primary levels.

- High-level language functions: (A high level programe language such as C++ contain function to perform input-output.
- Operating System: (Programmer can called operating system function from library known as Application programing interface. (API)

BIOS:

The basic input output system is the collection of low-level subroutines that communicate directly with hardware devices. The bois is install by computer's hardware. Operating systems typically communicate with the bios.

Basic Parts of an assembly language instruction

Instruction are statment that execute when we assemble the programme. The instruction (written in assembly language) are translated into machine language bytes.

An assembly instruction has 4 basic parts.

- Label (optional)
- Mnemonic (required)
- Operand (depends on the instruction)
- Comment (optional)

Question #3

Assembly Language and high level language

High-leved language are machine independent. They are essay to learn, easy to use and convenent for managing complex tasks.

Assembly language programe are machine specific it is the language that directly processor understand.

High-level language.

- It is programmer friendly language
- It is less memory efficient.
- Easy to understand.
- Simple to debug.
- Run on any plat form
- Simple to maintain
- Needs compiler for translation.

Assembly language

- It is machine friendly language.
- high memory efficient
- tough to understand.
- Complex to debugs
- Complex to maintain.
- Mechine dependent.
- Needs assembler for translation

Protected mode and real address

A real mode programm uses BIOS subroutines along with OS subroutine whereas a protected mode programme uses only OS subroutines.

Asembler and linker

Assembler translates the assembly program into codes.

A linker tool is used to link all parts of the programe together for execution

Instruction and directive

A directive is mainly an order, usually issued by an aurthority. A directive may established policy, assign responsibilities, define objective and delegated authority to those working in and with authorittative figure

Code lable and data label

- Data lable is the label that we use to define data as we defined memory locations num1, num2 ... etc is our progreme
- Code lable that we have on code and is normally used for loop control. statment

Line Comment and blockcomment

The first is called single line comment and as implies only to single line in the (source area). The second is called block comment and refers to a graph of text. A block comment has a stevt symbol and an end symbol and everything between is ignore by Computer.

Question # 4

Concept of Portibility as it applies to programming language

Probability in relation to software, is a measure of How easily application can be transffered from one computer environment to another. A computer software application is consider portable to a new environment if the effor required to adapt it to the new environment is within reasonable limits. The meaning of the abtract term (reasonable) depend upon the nature of application and it open difficult to expvess in quantifiable units Probability is the form of revsability. Some kinds of software are known to be less protable than other. An example of software that is not portable would by assembly code. Since Assembly code is specific to processor type.

No software is perfectly portable because all software has limitations. Some programming language are fairly portable, for example the C-language C compiler are readily available for all the majority of operating system, which is turn make C program very portable. This portability of C language programe has result in some programming re-writing their program and recompiling them is C to make them much more portable.

Portability is also used to describe the flexibility of the use of data. Some file formats are less portable then others. For example the view files with file formats such as PDF or JPEG, the formats depend on the availibity of appropriate software application.

Why would a high-level language not be an ideal tool for writting programe that directly Access a particular brand of printer?

A high level language may not be provided for direct hardware access. Even if it does awkward coding technique must often be used resulting in possible maintenance problems.

Why was unicode invented?

An earlier time, being an electronic device, a computer stove data converting into corresponding device. There were several charactd encoding syetems for making charata to some ASCII value or wumbers. For different language, numerous encoding is required for the letters, punctuation and other system. Hence the same character cauld used the same number or even two number are used for a single charadet, before discoving the unicode scheme, there may be a comflict within these traditional encoding scheme, Unicode waso descoveed to resolve this issue, representation that can be used for numbers, character, mathematical symbols and all other charcter. A unique number (8/16/32-bit) between U+0000 to U+10FFFF is availdble for encoding text or number from all the language. For example U+0042 represent the English letter "B"

Discuss the Basis program execution register used in x86 32-bit processor

Register: High-speed memories located in the CPU.

Register for 8086 and are 16 bits wide

Registers for IA-32 family are 32 bits wide.

Irvine kip R. Assimbly Language for x86 processor 6/e, 2010

Why does memory access take more machine cycles than register access

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

Question #5

Discuss the following MASM directives.

Include:

Insert source code from the source file given by file name into the current source file during assembly. Syntax.

Include file name.

.386

Enable assembly of nonprivileged instruction for the 80386 processor.

Disable assembly of instruction introduced with later processors (32-bit MASM only.)

Model

Initionalizes the program memory model (32-bit MASM only)

Syntax

.Model memory-model [[, language-type]]

[[, stack-option]]

Stack

Stack Every program written only in MASM has one main module where programe execution begins, Main module can contain code, data or stack segment define will all of the simplified segment directive. Any additional modules should contain only code and data segments.

Proto:

Proto type a function or procedure. You can call the function prototyped.
by the proto typet invoke directive by using the .Model, starts a near data segment. Syntax
. Label PROTO [[distance]] [[Language-type]]
[[. [[parameter]] :tag ...]]

Data

(32-bit MASM) When used with .Model starts a near data segment for initialized data.

.Code

(32-bit MASM only.) When used with .Model indicates the start of a code segment. Syntax.

.Code name

[[SegmentItem]] ... [[codesegmentnameoldENDS;;]]

PROC

Mark start and end of procedure block called label. The statments in the block can be called with the call instruction or INVOKE directive.

ENDP

Marks the end of procedure name privously begun with PROC syntax

name ENDP.

END

Marks the end of a module and optionally, sets the program entry point.

to procld. Syntax

End [procld]

Question #6

Write a programme that cacluculat the following expres

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

```
.386
.model flat, stdcall
.stack 4096
ExitProcess PROTO, dwExitCode: DWORD.

.Code.
main PROC.

moveax, 3h
mov ebx, 8h
mov ecx, 1h
mov edx, 8h.

add eax, ebx
add eax, edx
sub eax, ecx

Invoke Exitprocess, 0
main ENDP
END main
```

Write a programe that performs arithmetic operation on different register operands and store the result memory. Give step wise explanation

```
N = 12 // Loop index initial value
ADDC (r31, N, r1) // r1 = loop index
ADDC (r31, 1, r0) // r0 = compiled product
Loop: MUL (r0, r1, r0) // r0 = r0 * r1

SUBC (r1, 1, r1) // r1 = r1 - 1

BNE (r1, loop, r31) // if r1 != 0, Next PC = loop.
```