CS501-Advance Computer Architecture
Midterm Special 2006

WWW.VUTUBE.EDU.PK

Q1

_____all memory systems are dumb, in that they respond to only two commands: read or write

Virtually
Logically
Physically
None of These

Q 2

Consider two programs having three types of instructions given as follows: **[10]**

Number of	Program 1	Program 2
Data transfer instructions	7	12
Control instructions	3	5
ALSU instructions	6	3

Instructions Type	CPI
Control	5
ALSU	3
Data Transfer	4

Compare both the programs for the following parameters:

1. **Instruction count (IC)**
2. **Speed of execution (ET)**

Q 3

To access an operand in memory, the CPU must first generate an address, which it then sends to the _____

MEMORY
REGISTER

DATA BUS
ALL OF ABOVE

Q 4

_____ or Branch instructions affect the order in which instructions are performed, the flow of the program

Control
DATA MOVMENT
Arithmetic
LOGICAL

Q5

Reverse assemble the following SRC machine language instructions: [10]

68C2003A h

Q6

An instruction that specifies one operand in memory and one operand in a register is known as a _____ address instruction.

2-1/2
1-1/2
0
2

Q7

The data movement instructions _____ data within the machine and to or from I/O devices.

Input/output
Store
LOAD
MOVE
NONE OF ABOVE.

Q8

Write short answers to the following questions: [3 x 5]

a. What is the advantage of a linker in the development of assembly language programs?

Solution:-

The linker:

When developing large programs, different people working at the same time can develop separate modules of functionality. These modules can then be 'linked' to form a single module that can be loaded and executed. The modularity of programs, that the linking step in assembly language makes possible, provides the same convenience as it does in higher-level languages; namely abstraction and separation of concerns. Once the functionality of a module has been verified for correctness, it can be re-used in any number of other modules. The programmer can focus on other parts of the program. This is the so-called "modular" approach or "top-down" approach.

b. Define term "Single stepping".

Solution:-

Single stepping:

Single stepping and breakpoints that allow the examination of the status of the program and registers at desired points during execution.

c. Define term "Type checking".

Solution:-

Type Checking:-

High-level languages provide various primitive data types, such as integer, Boolean and a string, that a programmer can use. Type checking provides for the verification of proper usage of these data types. It allows the compiler to determine memory requirements for variables and helping in the detection of bad programming practices.

On the other hand, there is generally no provision for type checking at the machine level, and hence, no provision for type checking in assembly language. The machine sees only strings of bits. Instructions interpret the strings as a type, and it is usually limited to signed or unsigned integers and floating point numbers. A given 32-bit word might be an instruction, an integer, a floating-point number, or 4 ASCII characters. It is the task of the compiler writer to determine how high-level language data types will be implemented using the data types available at the machine level, and how type checking will be implemented.

d. Define term "Instruction set".

Solution:-

Instruction Set

A collection of all possible machine language commands that a computer can understand and execute is called its instruction set. Every processor has its own unique instruction set. Therefore, programs written for one processor will generally not run on another processor. This is quite unlike programs written in higher-level languages, which may be portable. Assembly/machine languages are generally unique to the processors on which they are run, because of the differences in computer architecture.

There are three ways to list instructions in an instruction set of a computer:

- by function categories
- by an alphabetic ordering of mnemonics
- by an ascending order of op-codes

e. Why computer logic design is different from classical logic design?

Solution:-

Classical logic design versus computer logic design :

The traditional sequential circuit design techniques for a finite state machine are not very practical when it comes to the design of a computer, in spite of the fact that a computer is a finite state machine. The reason is that employing these techniques is much too complex as the computer can assume hundreds of states.