:: MID TERM ASSIGNMENT ::

ID: 11533

Name: Ashir Ali Khan

Subject: Computer Architecture

Teacher: Sir Muhammad Amin

Iqra National University

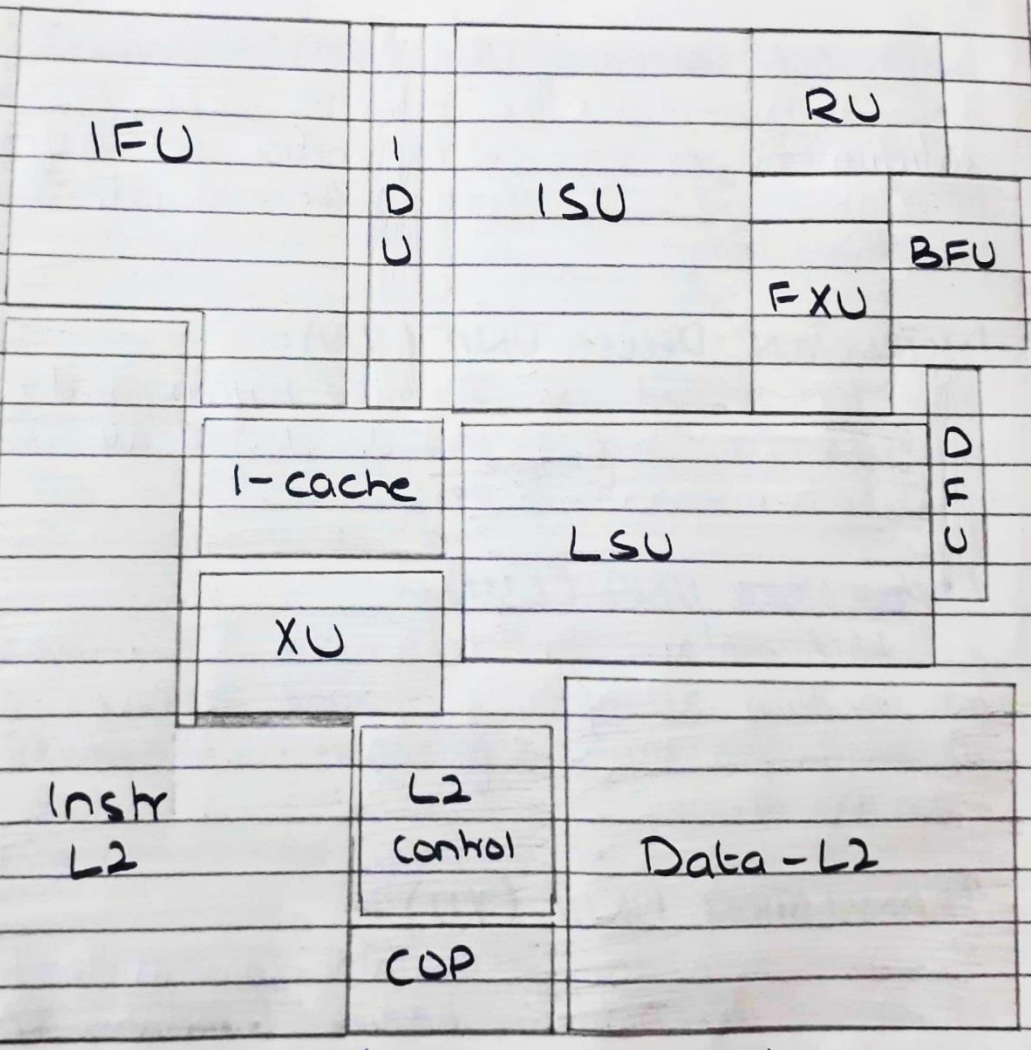
ROLL-NO :- 11533

QUESTION: 1 :-

Give Answers to each of the following:

PART : A :-

ANSWER :-



The following functional areas are on each core, as shown in the above figure:

INSTRUCTION SEQUENCE UNIT (ISU):-

This unit enables the out-of-order (OOO) pipeline. It

tracks register names, OOO instruction dependency, and handling of instruction resource dispatch. This unit is also central to performance measurement through a function called instrumentation.

INSTRUCTION FETCHING UNIT (IFU) (PREDICTION):-
These units contains the instruction cache, branch prediction logic, instruction fetching control, and buffers. Its relative size is the result of the elaborate branch prediction design.

INSTRUCTION DECODE UNIT (IDU):
The IDU is fed from the IFU buffers, and is responsible for parsing and decoding of all z/Architecture operation codes.

LOAD-STORE UNIT (LSU):-
The LSU contains the data cache. It is responsible for handling all types of operand accesses of all lengths, modes, and formats as defined in the z/Architecture.

TRANSLATION UNIT (XU):-
The XU has a large translation look aside buffer (TLB) and the Dynamic Address Translation (DAT) function that handles the dynamic translation of logical to physical addresses.

11533 (3)

FIXED-POINT UNIT (FXU):-

The FXU handles fixed-point arithmatic.

BINARY FLOATING-POINT UNIT (BFU):-

The BFU handles all binary and hexadecimal floating-point and fixed-point multiplication operations.

DECIMAL FLOATING-POINT UNIT (DFU):-

The DU runs both floating-point and decimal fixed-point operations and fixed-point division operations.

RECOVERY UNIT (RU):-

The RU keeps a copy of the complete state of the system that includes all registers, collects hardware fault signals, and manages the hardware recovery actions.

DEDICATED CO-PROCESSOR (COP):-

The dedicated coprocessor is responsible for data compression and encryption functions for each core.

I-cache:-

This is a 65 KbL1 instruction cache, allowing the IFU to prefetch intructions before they are needed.

L2 control:-

This is the control logic that manages the traffic

through the two L2 caches.

DATA – L2 :-

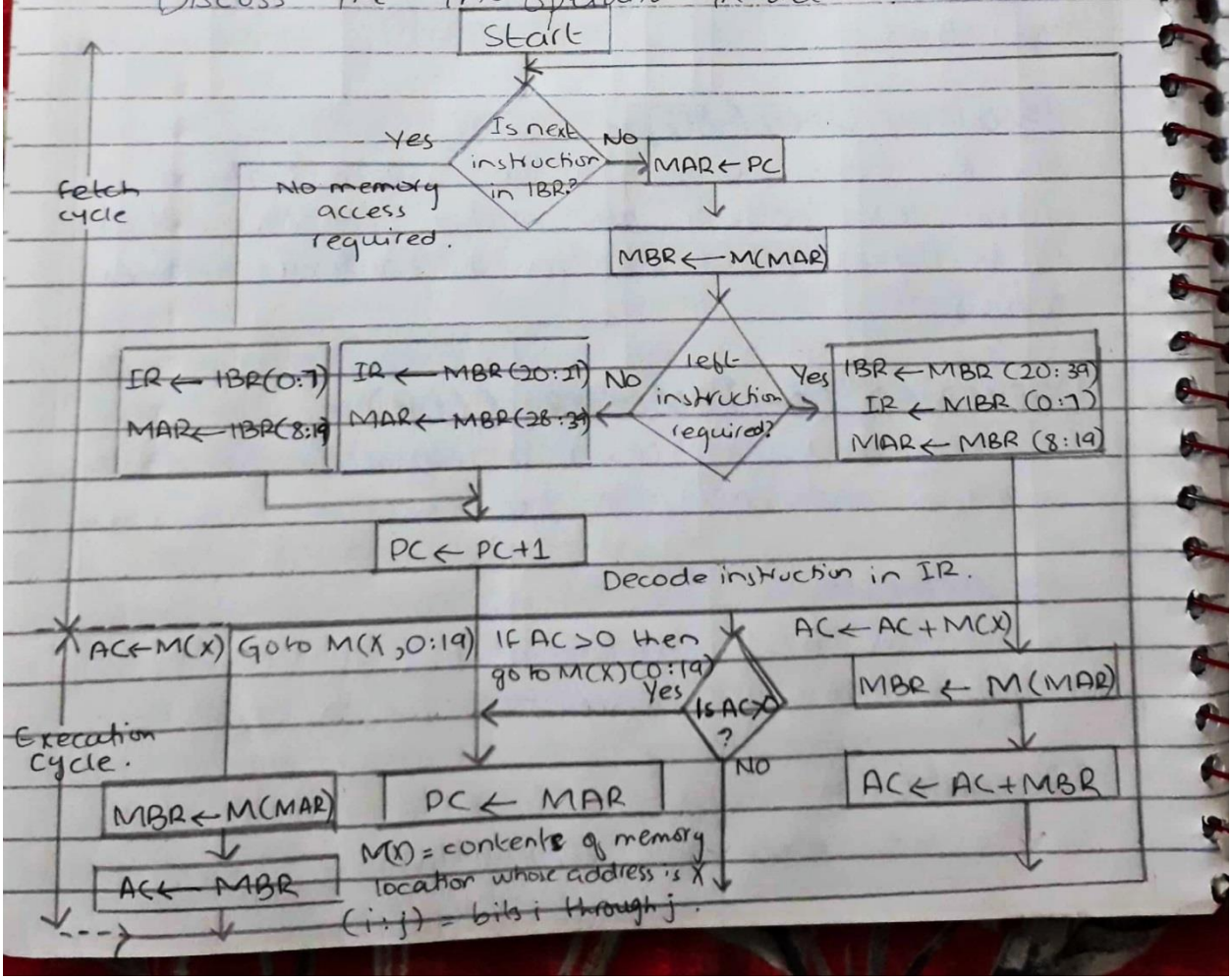
A 1mb L2 data cache for all memory traffic other than instructions.

INSTRA – L2 :-

A 1mb L2 instructions cache.

PART : B :-

Discuss the IAS operation in detail.



11533 (5)

DATA TRANSFER:-
Move data between memory and ALU registers or between two ALU registers.

UNCONDITIONAL BRANCH:-
Normally, the control unit executes intructions in sequences from memory. This sequence can be changed by a branch instruction, which facilitates repetitive operations.

CONDITIONAL BRANCH:-
The branch can be made dependent on a condition, thus allowing decision points.

ARITHMETIC:-
Operations performed by the ALU.

ADDRESS MODIFY:-
Permits addresses to be computed in the ALU and then inserted into instructions stored in memory. This allows a program considerable addressing flexibility.

X——X——X——X

PART: C :-

ANSWER:-
An Embedded system is a combination of computer hardware and software. As with any electronic system, this system requires a hardware platform and that is built with a microprocessor or microcontroller. The Embedded system hardware includes elements like user interface, Input/output interfaces, display and memory etc. Generally, an embedded system comprises power supply, processor, memory, timers,

serial communication ports and system application specific circuits.
Embedded system software is written in a high-level language and then compiled to achieve a specific function within a non-volatile memory in the hardware. Embedded system software is designed to keep in view of three limits. They are availability of system memory and processor speed. When the system runs endlessly, there is a need to limit the power dissipation for events like run, stop and wake up.
Examples of Embedded Systems :-

- Digital alarm clocks.
- Smart watches and digital wrist watches
- Washing machines and dishwashers.
- Air-conditioners and thermostats.
- Traffic lights.
- Printers, photocopy, fax machines and scanners.
- Digital and video cameras.
- Calculators.
- Digital thermometers.

PART : D) :-
ANSWER :-
Different desktop applications that requires the great power of contemporary micro processor-based system are :

⇒ Image processing applications.
⇒ Speech recognition.

⇒ Videoconferencing.
⇒ Multimedia authoring.
⇒ Voice and video annotation of files.
⇒ Simulation modeling.

PART: E :-

ANSWER :-

The techniques used for contemporary processors to increase speed are:

1) **Pipelining :-**
Pipelining is when computers receives multiple instructions and carry them out as they are recieved

2) **Branch Prediction :-**
Branch prediction is the process of being able to predict the next set of instruction so that they can caried out.

3) **Superscalar Execution :-**
Superscalar execution is when you are able to give more than one set of instruction at time.

4) **Data Flow Analysis :-**
Data flow analysis analyzes instructions that need each other.

5) **Speculative execution :-**
Specalative execution carry out instructions before they are actually executed.

PART : F :-

ANSWER :-

As the clock speeds and logic density increase, a number of obstacles become more significant including ....

- **POWER :-**

The power density increases with an increase in logic density and clock speed. One challenge of this is the difficulty of dissipating the heat generated on high-density, high-speed chips.

- **RC DELAY :-**

The speed at which electrons can flow on a chip between transistors is limited by the resistance and capacitance of the metal wires connecting them. delay increases as the RC product increases. As components on the chip decrease in size, the wires are closer together, increasing capacitance.

MEMORY LATENCY :-

Memory speeds lag processor speeds.

X——X——X—X

PART : G :-

ANSWER :-

Amdahl's law deals with the potential speedup of a program using multiple processors compared to a single processor. Consider a program running on a single processor such that a fraction (1-f) of the execution time involves code that is inherently sequential, and a fraction f that involves code that is infinitely parallelizable with

no scheduling overhead. Let T be the total execution time of the program using a single processor. Then the speedup using a parallel processor with N processors that fully exploits the parallel portion of the program is as follows:

$$\text{Speedup} = \frac{\text{Time to execute program on a single processor}}{\text{Time to execute program on N parallel processors}}$$

$$= \frac{T(1-f)+Tf}{T(1-f)+\frac{Tf}{N}} = \frac{1}{(1-f)+\frac{f}{N}}$$

Two important conclusions can be drawn:

1) when f is small, the use of parallel processors has little effect.
2) As N approaches infinity, speedup is bound by $1/(1-f)$, so that there are diminishing returns for using more processors.

PART : H :-

ANSWER :-

MULTICORE :-

Multicore refers to an architecture in which a single physical processor incorporates the core logic of more than one processor. A single integrated circuit is used to package or hold these processors. These single integrated circuits are known as a die. Multicore architecture places multiple processor cores and bundles these as a singles physical processor. The objective

is to create a system that can complete more tasks at the same time, thereby gaining better overall system performance. This technology is most commonly used in multicore processors, where two or more processor chips or cores run concurrently as a single system. Multicore-based processors are used in mobile devices, desktops, workstations and servers. The concept of multicore technology is mainly centered on the possibility of parallel computing, which can significantly boost computer speed and efficiency by including two or more central processing units (CPUs) in a single chip. This reduces the system's heat and power consumption. This means much better performances with less or the same amount of energy.

MIC :-

Chip manufacturers are now in the process of making a huge leap forward in the number of cores per chip, with more than 50 cores per chip. The leap in performance as well as the challenges in developing software to exploit such a large number of cores has led to the introduction of new term known as many integrated core (MIC).

GPGPU :-

A general-purpose GPU (GPGPU) is a graphics processing unit (GPU) that performs non-specialized calculations that would typically be conducted by the CPU (central processing unit). Ordinarily, the GPU is dedicated to

graphics rendering.
GPGPUs are used for tasks that were formerly the domain of high-power CPUs, such as Physics calculations, encryption/decryption, scientific computations and the generation of crypto currencies such as Bit coin. Because graphic cards are constructed for massive parallelism, they can dwarf the calculation rate of even the most powerful CPUs for many parallel processing tasks. The same shader cores that allow multiple pixels to be rendered simultaneously can similarly process multiple streams of data at the same time. Although a shader core is not nearly as complex as a CPU, a high-end GPU may have thousands of shader cores; in contrast, a multicore CPU might have eight or twelve cores.

PART: I:-

ANSWER:-

Quick Path Interconnect (QPI) protocol layers:-

QPI is defined as a four-layer protocol architecture, 3 encompassing the following layers.

Physical:-

Consist of the actual wires carrying the signals, as well as circuitry and logic to support ancillary features required in the transmission and receipt of the 1s and 0s. The unit of transfer at the physical layer is 20 bits, which is called a Phit (physical unit).

LINK :-

Responsible for reliable transmission and flow control. The link layer's unit of transfer is an 80-bit Flit (Flow control unit)

ROUTING :-

Provides the framework for directing packets through the fabric.

PROTOCOL :-

The high-level set of rules for exchanging packets of data between devices. A packet is comprised of an integral number of Flits.

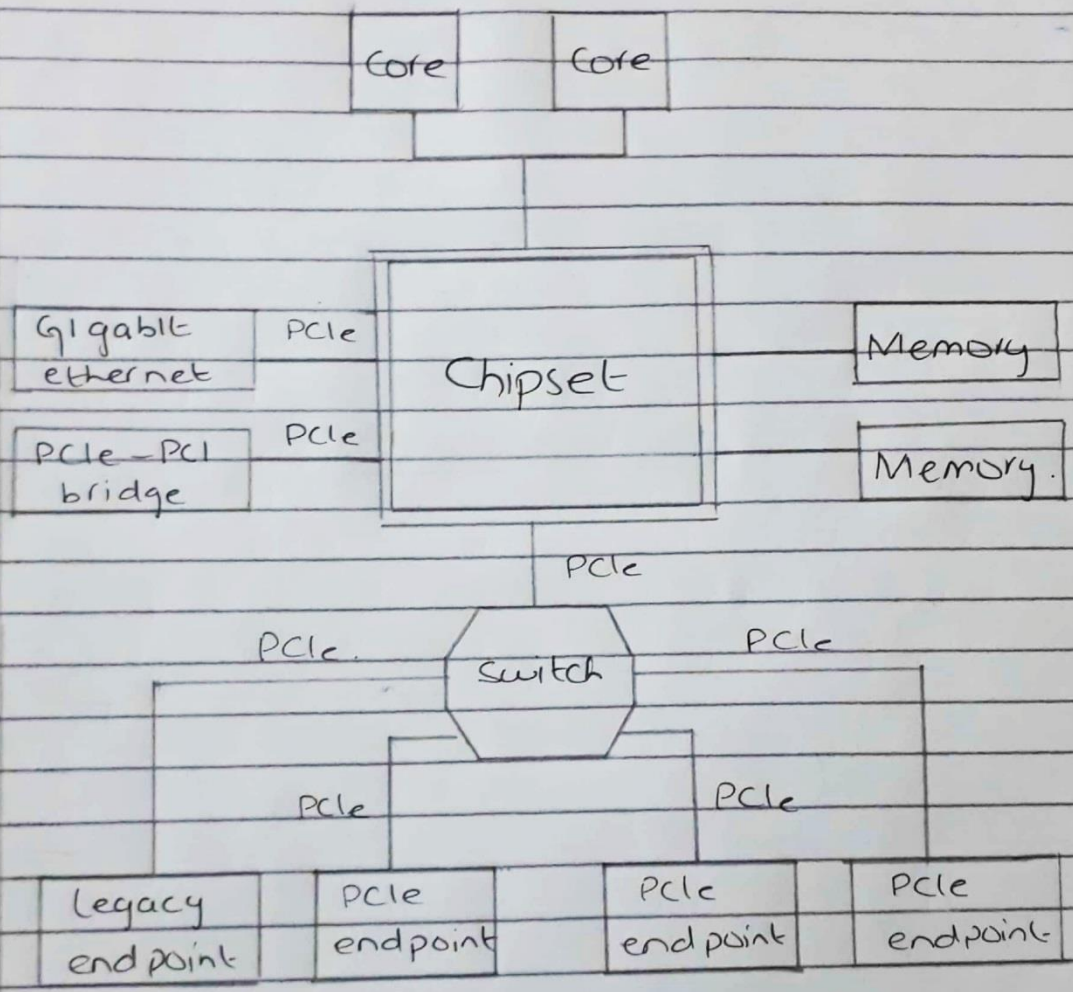
PART : J :-

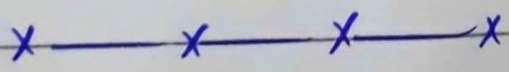
ANSWER :-

Physical and logical architecture of PCIe :-

A root Complex device, also refered to as a chipset or a host bridge, connects the processor and memory subsystem to the PCI Express switch fabric comprising one or mor PCIe and PCIe switch devices. The root complex acts as a buffering device, to deal with difference in data rates between I/O controllers and memory and processor components. The root complex also translates between PCIe transaction formats and the processor and memory signals and control requirement. The chipset will typically support multiple PCIe ports, some of which attach directly to a PCIe device, and one or more that attach to a switch that manages multiple PCIe streams.

11533. (13)


Core
Core
Chipset
Gigabit ethernet
PCIe
PCIe-PCI bridge
PCIe
Memory
Memory.
PCIe
PCIe
Switch
PCIe
PCIe
PCIe
Legacy end point
PCIe endpoint
PCIe endpoint
PCIe endpoint



11533 (14)

QUESTION: 2:-

Write a detail note on each of the following:

PART: A:-

STRUCTURAL COMPONENTS OF A COMPUTER:-

A computer has four main components: the central processing unit or CPU, the primary memory, input units and output units. A system bus connects all four components, passing and relaying information among them. This type of computer organization and architecture is called a "Von Neumann machine" after John von Neumann, who finalized the theory and design of the first modern digital computer.

⇒ C.P.U:-

Computer scientists typically call the CPU the "brain" of the computer because this is where programs are executed. A program is a set of instructions that tells the computer how to accomplish a specific task, such as sending a file to the printer, opening a browser window, or playing music or video.

The CPU is further broken up into three smaller components: the arithmetic unit handles all the simple mathematical computation; the control units interpret the instructions in a computer program; and the instruction decoding unit converts computer programming instructions into machine code.

Machine code is the basic language understood by all the components in a computer.

⇒ **MEMORY :-**

Once the CPU converts a specific set of computer program instructions into machine code, it stores that machine code in primary storage or memory. The machine code will be treated as either data or instructions. The CPU fetches data and instructions from memory, uses an instruction to manipulate the data, and then sends the result and the next set of instructions back to memory.

⇒ **INPUT UNITS :-**

Input units are all the devices you use to feed information to the computer, such as a keyboard, a hard drive or a networking card. These devices, in essence, bring data from the "outside world" into your computer, in much the same way that your eyes and ears bring information to your brain. Each input device has its own hardware controller that connects to the CPU and primary memory, and it has a set of instructions that tells the CPU how to use it.

⇒ **OUTPUT UNITS :-**

Output units are the devices your computer uses to relay information to the user, such as a printer, monitors and speakers. For example, everything you see on your computer monitor stacks as machine code in memory. The CPU takes that

machine code and converts it into a format required by your monitor's hardware. Your monitor's hardware then converts that information into different light intensities so that you see words or pictures.

THE SYSTEM BUS :-

The system bus lets the four components of the computer communicate with one another. The system bus transmits data and instructions. It also sends addresses that tell the CPU where in primary memory the data and instructions are coming from and where the results should go.

PART : B :-

THE KEY CHARACTERISTICS OF A COMPUTER FAMILY :-

⇒ Similar or identical instruction set: In many cases, the exact same set of machine instructions are supported on all members of the family. Thus, a program that executes on one machine will also execute on any other. In some cases, the lower end of the family has an instruction set that is a subset of that of the top end of the family. This means that programs can move up but not down.

⇒ Similar or identical operation system: The same basic operating system is available for all

family members. In some cases, additional features are added is the higher-end members.
⇒ Increasing speed: The rate of instruction execution increases in going from lower to higher family members.
⇒ Increasing number of I/O ports: In going from lower to higher family members.
⇒ Increasing memory size: In going from lower to higher family members.
⇒ Increasing cost: In going from lower to higher family members.

X——X——X——X

PART: C:-

STORED PROGRAM COMPUTER:-

A stored-program computer is a computer that stores program instructions in electronic memory. This contrasts with machines where the program instructions are stored on plug boards or similar mechanisms.

A computer with a von Neumann architecture stores program data and instruction data in the same memory; a computer with a Harvard architecture has seperate memories for storing program and data. Both are stored-program designs. Stored-program computer is sometime used as a synonym for von Neumann architecture, however Professor Jack Copeland considers that it is "historically inappropriate, to refer to electronic stored-program

digital computers as "von Neumann machines". Hennessy and Patterson write that the early Harvard machines were regarded as "reactionary by the advocates of stored-program computers."

X — X — X — X

PART: D :-

MOORE'S LAW :-

Moore's law is a computing term which originated around 1970; the simplified version of this law states that processor speeds, or overall processing power for computers will double every two years. A quick check among technicians in different computer companies shows that the term is not very popular but the rule is still accepted.

To break down the law even further, it specifically stated that the number of transistors on an affordable CPU would double every two years (which is essentially the same thing that was stated before) but 'more transistors' is more accurate.

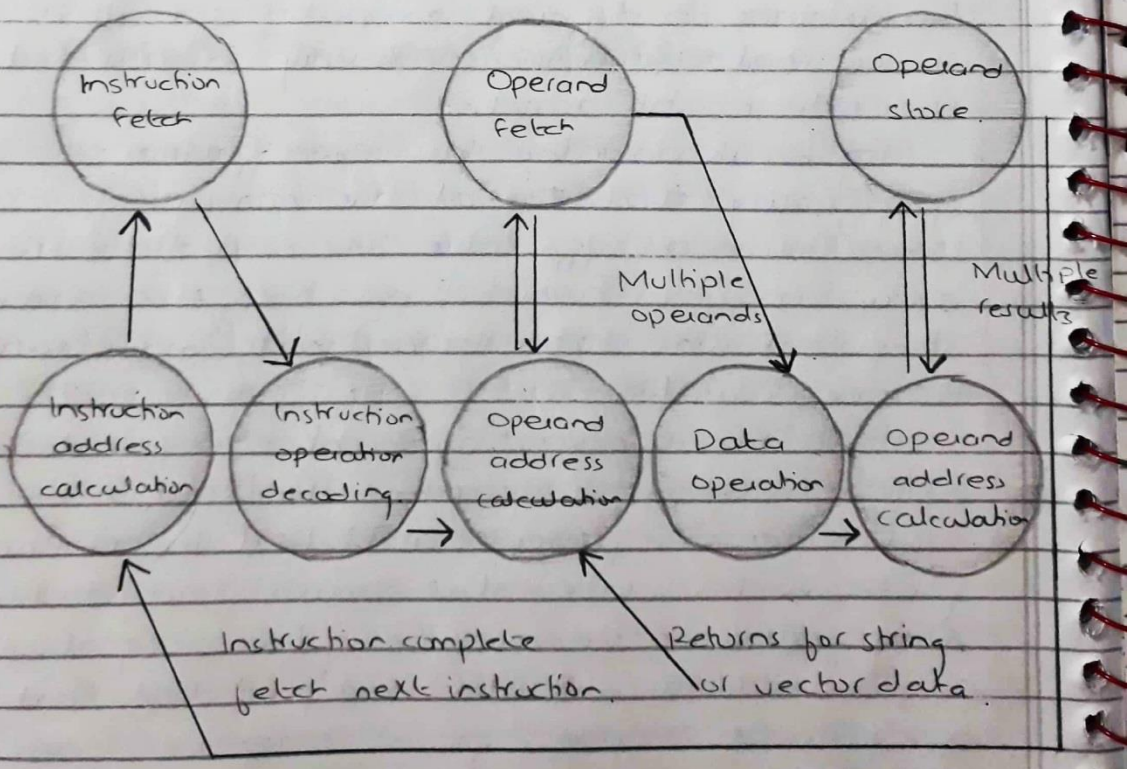
The law is named after Intel co-founder Gordon Moore, who described the trend in his 1965 paper. The paper stated that the number of components in integrated circuits had doubled every year from the invention of the integrated circuit in 1958 until 1965 and predicted that the trend would continue "for at least ten years". His prediction has proved very accurate. The law

is used in the semiconductor industry to guide long-term planning and to set targets for research and development.

The capabilities of many digital electronic devices are strongly linked to Moore's law: processing speed, memory capacity, sensors and even the number and size of pixels in digital cameras. All of these are improving at (roughly) exponential rates as well.

PART: E :-

INSTRUCTION CYCLE STATE DIAGRAM:-



⇒ Instruction address calculation (iac): Determine the address of the next instruction to be executed. Usually, this involves adding a fixed number to the address of the previous instruction. For example, if each instruction is 16 bits long and memory is organized into 16-bit words, then add 1 to the previous address. If instead, memory is organized as individually addressable 8-bit bytes, then add 2 to the previous address.

⇒ Instruction fetch (if): Read instruction from its memory location into the processor.

⇒ Instruction operation decoding (iod): Analyze instruction to determine type of operation to be performed and operand(s) to be used.

⇒ Operand address calculation (oac): If the operation involves reference to an operand in memory or available via I/O then determine the the address of the operand.

⇒ Operand fetch (of): Fetch the operand from memory or read it in from I/O.

⇒ Data operation (do): Perform the operation indicated in the instruction.

⇒ Operand store (os): Write the result into memory or out to I/O.

PART : F :-

CLASSES OF INTERRUPTS :-

PROGRAM :-

Generated by some condition that occurs as a result of a instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, or reference outside a user's allowed memory space.

TIME :-

Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis.

I/O :-

Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions.

HARDWARE FAILURE :-

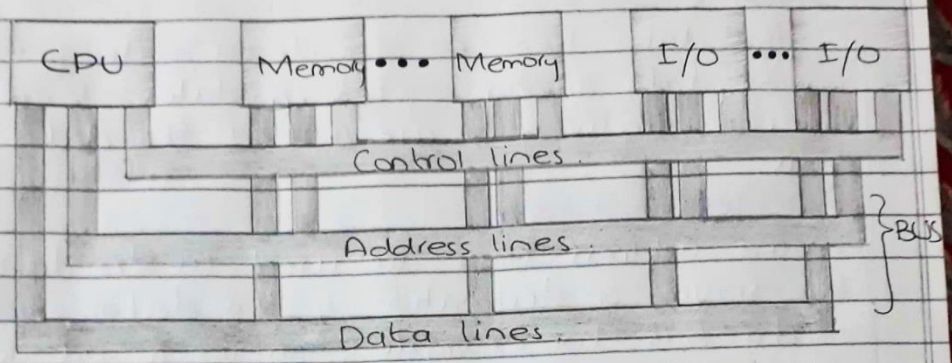
Generated by a failure such as power failure or memory parity error.

X —— X —— X —— X

PART : G :-

BUS INTERCONNECTION SCHEME :-

11533 (22)



BUS INTERCONNECTION SCHEME.

A bus is a communication pathway connecting two or more devices. A key characteristic of a bus is that it is a shared transmission medium. Multiple devices connect to the bus and a signal transmitted by any one device is available for reception by all other devices attached to the bus. If two devices transmit during the same time period, their signals will overlap and become garbled. Thus, only one device at a time can successfully transmit. Typically, a bus consist of multiple communication pathways, or lines. Each line is capable of transmitting signals representing binary 1 and binary 0. An 8-bit unit of data can be transmitted over eight bus lines. A bus that connects major computer components (processor, memory, I/O) is called a system bus.

DATA LINES :-

The data lines provide a path for moving data among system modules. These lines, collectively, are called the data bus.

ADDRESS LINES :-

The address lines are used to designate the source or destination of the data on the data bus. For example, on an 8-bit address bus, address 01111111 and below might reference locations in a memory module (module 0) with 128 words of memory, and address 10000000 and above refers to devices attached to an I/O module (module 1)

CONTROL LINENS :-

The control lines are used to control the access to and the use of the data and address lines. Control signals transmit both command and timing information among system modules. Timing signals indicate the validity of data and address information. Command signals specify operations to be performed. Typical control lines include:

- **MEMORY WRITE :-**
  Causes data on the bus to be written into the addressed location.
- **MEMORY READ :-**
  Causes data from the addressed location

to be placed on the bus.

- **I/O Write:-**
  Causes data on the bus to be output to the addressed I/O port.
- **I/O read:-**
  Causes data from the addressed I/O port to be placed on the bus.
- **Transfer ACK:-**
  Indicates that data have been accepted from or placed on the bus.
- **Bus request:-**
  Indicates that a module needs to gain control of the bus.
- **Bus grant:-**
  Indicates that a requesting module has been granted control of the bus.
- **Interrupt request:-**
  Indicates that an interrupt is pending.
- **Interrupt ACK:-**
  Acknowledges that the pending interrupt has been recognized.
- **Clock:-**
  Is used to synchronize operations.
- **Reset:-**
  Initializes all modules.

The operation of the bus is as follows. If one module wishes to send data to another,

it must do two thing. To obtain the use of the bus, and to Transfer data via the bus. If one module wishes to request data from another module, it must obtain the use of the bus, and Transfer a request to the other module over the appropriate control and address lines. It must then wait for that second module to send the data.

X——X——X——X

QUESTION : NO:3 :-

Differentiate each of the following:

PART: A :-

Computer Organization and Computer Architecture :-

COMPUTER ORGANIZATION	COMPUTER ARCHITECTURE.
1) Computer Organization is concerned with the structure and behaviour of a computer system as seen by the user.	1) Computer Architecture is concerned with the way hardware components are connected together to form a computer system.
2) It deals with the components of a connection in a system.	2) It acts as the interface between hardware and software.
3) Computer Organization tells us how exactly all the	3) Computer Architecture help us to understand

units in the system are arranged and interconnected	the funtionalities of a system
4) whereas Organization expresses the realization of architecture.	4) A programmer can view architecture in terms of instructions, addressing modes and registers
5) An organization is done on the basis of architecture	5) While designing a computer system architecture is considered first.
6) Computer Organization deals with low-level design issues.	6) Computer Architecture deals with high-level design issues.
7) Organization involves physical Components (circuit design, Adders, Signals, peripherals	7) Architecture involves logic (instruction sets, Addressing modes, Data types, Cache optimization)

PART : B :-

RISC and CISC :-

RISC	CISC
1) RISC stands for Reduced instruction set.	1) CISC stands for complex Instruction set computer.
2) RISC processors have simple instructions taking about one clock cycle. The average clock cycle per instruction (CPI) is 1.5.	2) CSIC processor has complex instruction that takeup multiple clocks for execution. The average clock cycle per instruction (CPI) is in the range of 2 and 15.
3) Performance is optimized	3) Performance is optimized

with more focus on software	with more focus on hardware.
4) It has no memory unit and uses seperate hardware to implement instructions.	4) It has a memory unit to implement complex instructions.
5) It has a hard-wired unit of programming.	5) It has a micropogramming unit.
6) The instruction set is reduced i-e it has only a few instruction in the instruction set. Many of these instruction are very positive.	6) The instruction set has a variety of different instructions that can be used for complex operation
7) The instruction set has a variety of different instructions that can be used for complex operations.	7) CISC has many different addressing modes and can thus be used to represent higher-level programming language statements more efficiently.
8) Complex addressing modes are synthesized using the software.	8) CISC already supports complex addressing modes.
9) Multiple register sets are present.	9) Only has a single register set.
10) RISC processors are highly pipelined.	10) They are normally not pipelined or less pipelined.
11) The complexity of RISC lies with the compiler that executes the program.	11) The complexity lies in the microprogram.
12) Execution time is very less.	12) Execution time is very high.
13) Code expansion can be a	13) Code expansion is not a problem.

problem.

14) The decoding of instruction is simple.

15) It does not require external memory for calculations

16) The most common RISC microprocessors are Alpha, ARC, ARM, AVR, MIPS, PA-RISC, PIC, Power Architecture, and SPARC.

17) RISC architecture is used in high-end application such as video processing, telecommunications and image processing.

14) Decoding of instruction is complex.

15) It requires external memory for calculations

16) Examples of CISC processors are the System/360, VAX, PDP-11, Motorola 68000 family, AMD, and Intel x86 CPUs.

17) CISC architecture is used in low-end applications such as security systems, home automation, etc.

PART: C :-

MICROPROCESSOR AND MICROCONTROLLERS :-

MICROPROCESSORS	MICROCONTROLLERS :
1) Microprocessor is the heart of computer system.	1) Micro controller is the heart of embedded.
2) It is only a processor, so memory and I/O components need to be connected externally.	2) Micro Controller has a processor along with internal memory and I/O components.
3) Memory and I/O has to be connected externally, so the circuit becomes large.	3) Memory and I/O are already present, and the internal circuit is small.

4) You can't use it in compact system.	4) You can use it in compact system.
5) Cost of the entire system is high.	5) Cost of the entire system is low.
6) Due to external components, the total power consumption is high. Therefore, it is not ideal for the devices running on stored power like batteries.	6) As external components are low, total power consumption is less. so it can be used with devices running on stored power like batteries.
7) It is mainly used in personal computers.	[illegible]
8) Microprocessors has a smaller numbe of registers, so more operations are memory based.	8) It is used mainly in a washing machine, MP3 players, and embedded system.
9) Microprocessors are based on Van Neumann model.	8) Microcontroller has more register. Hence the programs are easier to write.
10) It is a central processing unit on a single silicon based integrated chip.	9) Microcontrollers are based on Harvard architecture.
11) It has no RAM, ROM, Input-Output units, timers and other peripherals on the chip.	10) It is a by product of the development of micro-processor with a CPU along with other peripherals.
12) It uses an external bus to interface to RAM, ROM, and other peripherals.	11) It has a CPU along with RAM, ROM, and other peripheral embedded
13) Microprocessor-based systems can run at a very high speed because of the	

technology involved.	or a single chip.
14) It's used for general purpose applications that allow you to handle loads of data.	12) It uses an internal controlling bus.
15) It's complex and expensive, with a large number of instructions to process.	13) Microcontroller based system run up to 200 MHz or more depending on the architecture.
16) Most of the microprocessors do not have power saving features.	14) It is used for application-specific systems.
	15) It's simple and inexpensive with less number of instructions to process.
	16) Most of the microcontrollers offer power-saving mode.

PART : D :-

CORTEX-A, CORTEX-R, AND CORTEX-M :-

CORTEX-A	CORTEX-R	CORTEX-M
⇒ Application processors.	⇒ Real time processors.	⇒ Micro-controllers.
⇒ Used in wide range of devices that have fully functional processors	⇒ Used in critical system where data interpretion is essential.	⇒ Designed for small devices and mixed signal processing.
⇒ High performance, high efficiency and	⇒ High performance, real time and safe	⇒ low performance point, higher

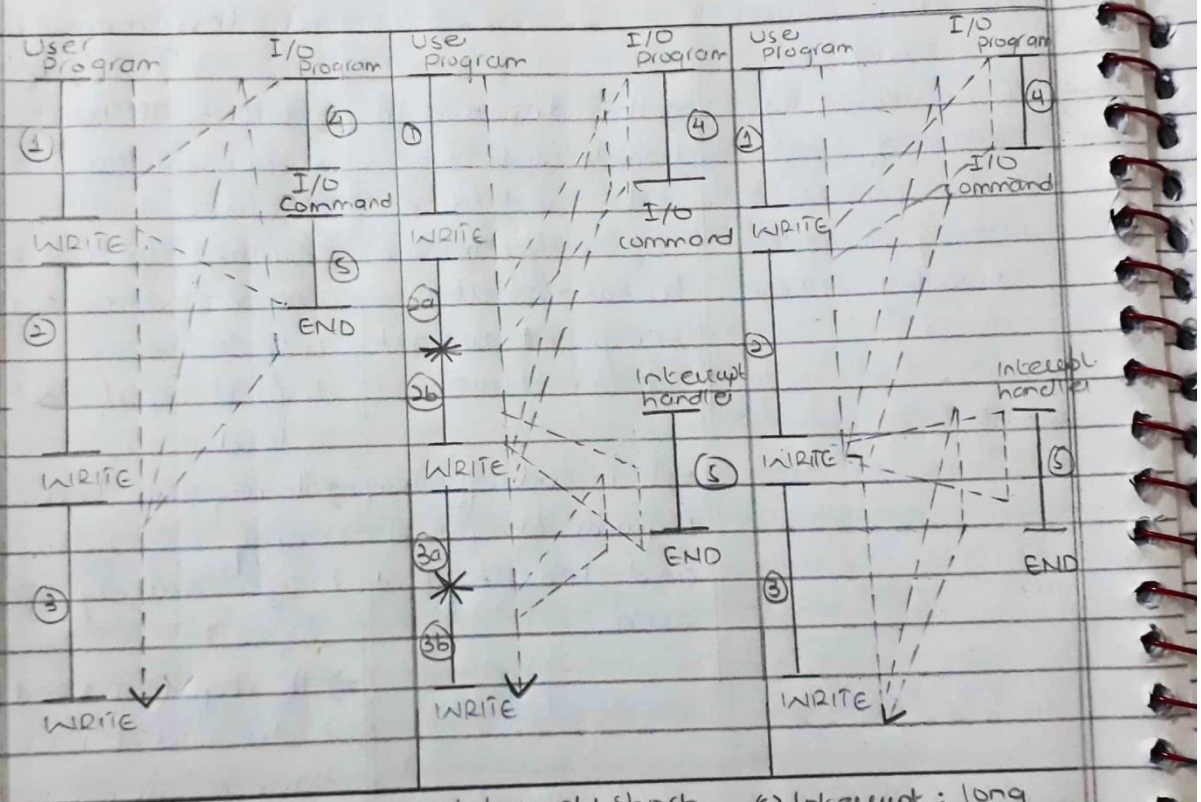
high computational power.		efficiency.
⇒ It runs at relatively high clock frequency.	⇒ It runs on high clock frequency.	⇒ It runs at slower clock speed.
⇒ It is connected to large amount of memory.	-------	⇒ It is connected to less memory.
⇒ It handles large amount of applications and is capable of running complex operating system directly.	⇒ It is designed to handle fast changing data, and to be sufficiently responsive to handle data through put without slowing down.	⇒ It is built into micro controller with I/O lines and designed for small factor systems that rely on heavy digital input and output.
-------	⇒ It is cost effective, requires low power and less physical area.	⇒ It requires less energy and has longer battery life.
-------	-------	⇒ It requires small code and it is easy to use.
⇒ Applications - Mobiles, telephone, tablets, laptops, etc.	⇒ Applications - Medical devices, car systems.	⇒ Applications - Robotic systems, small consumer electronics.

11533 (32)

PART : E :-

Program flow of control without interrupt and with interrupt :-

PROGRAM FLOW CONTROL :-



(a) No interrupts (b) Interrupt; short I/O wait. (c) Interrupt; long I/O wait.

1, 2 and 3 - code segments refer to sequences of instruction that do not involve I/O.
The WRITE calls are calls to an I/O program that is a system utility and that will perform the actual I/O operation.

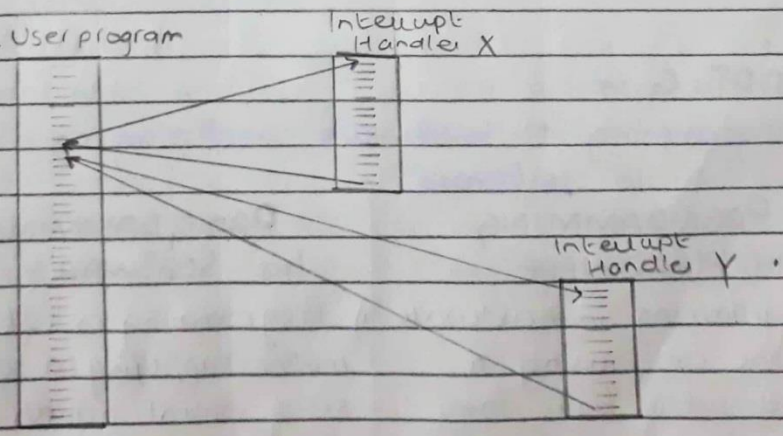
The I/O consists of three sections:
4-A sequence of instructions which prepare for the operation.
I/O command - the actual I/O command.
5-A sequence of instruction which complete the operation.

PART: 4F:-

Disabled interrupt and nested interrupt processing:

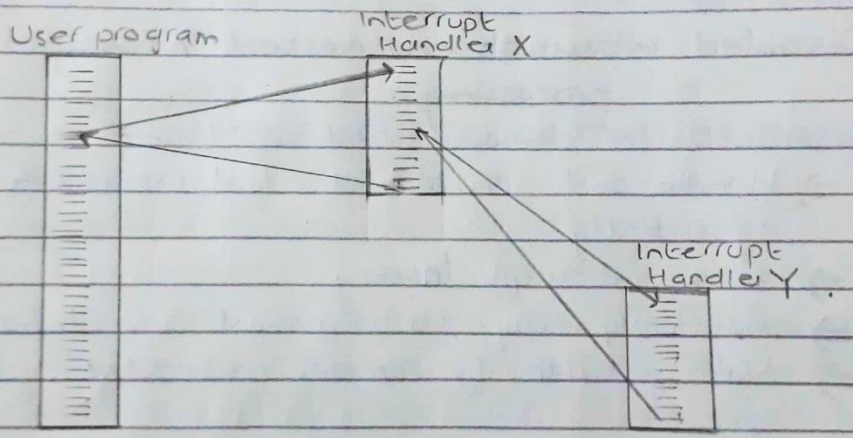
DISABLED INTERRUPT PROCESSING:-

⇒ Handle and service individual interrupts sequentially.
⇒ High interrupt latency.
⇒ Relatively easy to implement and debug.
⇒ Not suitable for complex embedded systems.



NESTED INTERRUPT PROCESSING :-

⇒ Handle multiple interrupts without a priority assignment.
⇒ Medium or high interrupt latency.
⇒ Enable interrupts before the servicing of an individual interrupt is complete.
⇒ No prioritization, so low priority interrupts can block higher priority interrupts.



PART: G :-

Programming in hardware and programming in software :-

PROGRAMMING IN HARDWARE	PROGRAMMING IN SOFTWARE :
1) Programming in hardware means configuring a small set of basic logic components specifically	1) Programming in software means supplying a specific set of control signals to a general-purpose hard-

for a particular computation.	ware.
2) Hardware programming languages are concurrent in nature and executed a piece of code in parallel.	2) A software programming language are sequential in nature and executed a piece of code sequentially.
3) Hardware programming language are often used to model a digital circuits and the circuits sythesized to a hardware.	3) Software programming languages are often executed as a piece of instructions to CPU and the code is not synthesizable.
4) Hardware programming languages are used to find the Timing Delay of the circuit.	4) Software languages cannot be used for to find the timing Delay of the circuit.
5) Hardware languages are faster and the usage of memory allocation is critical considering the formation of chip.	5) Software languages are slower and the consideration of memory usage is not critical.
6) For hardware language, Knowledge of digital and hardware circuit is needed.	6) For software language, Algorithm and processor Knowledge is needed.
7) Some of the examples of HDL (Hardware Description language) which are used as Hardware language are VHDL, Verilog, System Verilog.	7) Some examples of software languages are C, C++.

QUESTION : 4 :-
Solve Each of the following.

PART : A :-
ANSWER :-
(a) Show the assembly language code for the program, starting at address 08A.

Address	Contents
08A	LOADM(0FA)
	STOR M(0FB)
08B	LOAD M(0FA)
	JUMP+M(08D)
08C	LOAD - M(0FA)
	STOR M(0FB)
08D	

(b) Explain what this program does.
ANSWER:
This program is to store the absolute value of content at memory location 0FA into memory location 0FB.

X——X——X——X

PART : B :-
ANSWER :-

Opcode	Operand
00000001	000000000010

In the beginning, the CPU have to fetch the instruction

from the memory. Then, the instruction will include the address of the data which is required to load. Through the execution time, the memory will be accessed in that time to load the data contents which is located at that address for a total of two trip to memory.

PART: C :-

GIVEN :-

Clock speed of the processor = 60MHz

Number of instructions the executed program consist = 104,000

Instruction Type	Instruction Count	Cycles per instruction
Integer arithmetic	46000	1
Data Transfer	33000	2
Floating point	16000	2
Control Transfer	9000	2

TO FIND :-

CPI = ?

MIPS rate = ?

Execution time = ?

SOLUTION :-

Calculating the CPI is :

$$CPI = \frac{\text{Instruction count} \times \text{Cycles per second}}{\text{Number of instruction the executed program consist}} \quad \text{--- (1)}$$

Substitute the values of "instruction count" and "cycles per second" from the above table in equation (1).

11533 (38)

$$CPI = \frac{(46000 \times 1) + (33000 \times 2) + (16000 \times 2) + (9000 \times 2)}{104,000}$$

$$CPI = \frac{46000 + 66000 + 32,000 + 18000}{104,000}$$

$$= \frac{162,000}{104,000}$$

$$= 1.55$$

Therefore, the CPI for this program is 1.55

Now Calculating MIPS :-

$$MIPS = \frac{f}{CPI \times 10^6}$$

Frequency is given as 60MHz, putting values (converted MHz to Hz)

$$MIPS = \frac{60 \times 10^6}{1.55 \times 10^6}$$

$$= 38.70$$

Calculating execution time :-

Execution time = CPI x Instruction count x clock time

$$= \frac{CPI \times \text{Instruction Count}}{\text{frequency}}$$

$$= \frac{1.55 \times 104000}{60 \times 10^6}$$

$$= 0.0026$$

Execution time = 0.0026

PART : D :-

ANSWER :-

(a) Determine the average CPI :-

Since we have the same instruction mix, that means the additional instructions for each task could be allocated appropriately between the instruction types. Therefore, the following table be gotten.

Instruction Type	CPI	Instruction Mix
Arithmetic and logic	1	60%.
Load/store with cache hit.	2	18%.
Branch	4	12%.
Memory reference with cache miss	12	10%.

The average CPI = $(1 \times 0.6) + (2 \times 0.18) + (4 \times 0.12) + (12 \times 0.1)$

$= 2.64$.

Therefore, the CPI has been increased since the time for memory access is also increased.

11533 (40)

(b) determine the average MIPS rate:

MIPS = 400/2.64 = 152
MIPS = 152
There is a corresponding drop in the MIPS rate.

(C) Calculate the speed up factor.
The speedup factor equals to the ratio of the execution times. The execution time is calculated as the following : $T = I_c / (MIPS \times 10^6)$
For the one processor, $T_1 = (2 \times 10^6)/(178 \times 10^6)$
$= 11 ms.$
For the 8 processor, each processor executes 1/8 of the 2 million instructions plus the 25,000.

$$T_8 = \frac{\frac{2 \times 10^6}{8} + 0.025 \times 10^6}{152 \times 10^6}$$

$= 1.8 ms$
Therefore we have,
Speed up = $\frac{\text{time to execute program on a single processor}}{\text{time to execute program on N parallel processor.}}$

$= \frac{11}{1.8} = 6.11$

(d) Compare the actual speedup factor with the theoretical speedup factor determined by Amdahl's law.

In fact, there are two inefficiencies in the parallel system.
The first one is that there are more additional instructions which is added to coordinate between threads.
The second one is that there is contention for memory access. Thus, none of the code is inherently serial, and all of it is parallelizable but with scheduling overhead. It could be said that the memory access conflict means some extent memory reference instructions are not parallelizable.
By depending on the information given, it is not obvious how to qualify this effect in Amdahl's equation.
Therefor, if it is supposed that the fraction of code, which is parallelizable, is f = 1, then Amdahl's law decreases to Speedup = N = 8. Therefore, the actual speedup is only about 75% of the theoretical speedup.

PART : E :-

ANSWER :-

STEP : 1 :-

(a) The PC contains 300, the address of the first instruction. This value is loaded in to the MAR.
(b) The value in location 300 (which is the instruction with the value 1940 in Hexadecimal) is loaded into the MBR, and the PC is incremented. These two steps can be done in parallel.
(c) The value in the MBR is loaded into the IR.

STEP: 2 :-

(a) The address portion of the IR (940) is loaded into the MAR.

(b) The value in location 940 is loaded into the MBR.

(c) The value in the MBR is loaded into the AC.

STEP: 3 :-

(a) The value in the PC (301) is loaded in to the MAR.

(b) The value in location 301 (which is the instruction with the value 5941) is loaded into the MBR, and the PC is incremented.

(c) The value in the MBR is loaded into the IR.

STEP: 4 :-

(a) The address portion of the IR (941) is loaded into the MAR

(b) The value in location 941 is loaded into the MBR.

(c) The old value of AC and the value of location MBR are added and the result is stored in the AC.

STEP: 5 :-

(a) The value in the PC (302) is loaded in to the MAR.

(b) The value in location 302 (which is the instruction with the value 2941) is loaded into the MBR, and the PC is incremented.

(c) The value in the MBR is loaded into the IR.

STEP:6 :-

(a) The address portion of the IR (941) is loaded into the MAR.

(b) The value in the AC is loaded into the MBR

(c) The value in the MBR is stored in location 941.

PART, F :-

ANSWER :-

(A) What is the maximum directly addressable memory capacity (in bytes)?

$2^{(32-8)} = 2^{(24)}$

= 16,777,216 bytes = 16MB (8 bits = 1 byte for the opcode).

(B) Discuss the impact on the system speed if the microprocessor bus has:

(b.1) 32-bit local address bus and a 16-bit local data bus, or.

A 32-bit local address bus and a 16-bit local data bus. Instruction and data transfers would take three bus cycle each, one for the address and two for the data. Since if the address bus is 32 bits, the whole address can be transferred to memory at once and decoded there; however, since

the data bus is only 16 bits, it will require 2 bus cycles to fetch the 32-bit instruction or operand.

(b.2) 16-bit local address bus and a 16-bit local data bus.

A 16-bit local address bus and a 16-bit local data bus. Instruction and data transfers would take few bus cycles each, two for the address and two for the data. Therefore, that will have the processor perform two transmissions in order to send to memory the whole 32-bit address; this will require more complex memory interface control to latch the two halves of the address before it performs an access to it. In addition to this two-steps address issue, since the data bus is also 16-bits, the microprocessor will need 2 bus cycles to fetch the 32-bit instruction or operand.

(c) How many bits are needed for the program counter and the instruction register?

For the PC needs 24 bits (24-bit addresses), and for the IR needs 32 bits (32-bit addresses).

X——X——X——X

PART: G :-

ANSWER:-

First we need to find the time taken to fetch one operand from one memory location, frequency and bus cycle.

11533 (45)

Given clock rate = 4MH₂

$$\text{Frequency} = \frac{1}{\text{Clock Rate}}$$

$$= \frac{1}{4\text{MH}_2}$$

$$= 0.25\text{MS}$$

Therefore, frequency or bus cycle is 0.25MS
Then, memory cycle will take 0.25MS x 4 = 1MS
Hence to fetch one operand from memory 1MS is required.
By applying this " if an odd-aligned word is referenced, two memory cycles, each consisting of four bus cycles, are required to transfer the word" from the Question, so we will have 3 cases:
First case: is if both operands are even-aligned, so the time required is 1x1 = 2MS to fetch both operands.
Second case: is if both operands are odd-aligned, so the time required is 1x4 = 4MS to fetch both operands.
Third case: is if only one is odd-aligned, so the time required is 1x3 = 3MS to fetch both operands.

X—X—X—X

::THE END::