Computer Architecture & Organization
(2019-20) final —

1. a) Differentiate between SRAM and DRAM.

SRAM (Static Random Access Memory) and DRAM (Dynamic Random Access Memory) are two types of RAM (Random Access Memory) used in computers and other electronic devices. While both serve as temporary storage for data that the CPU needs to access quickly,

SRAM	DRAM
i. It stores information as long as the power is supplied	i. It stores information as long as the power supplied or a few milliseconds when the power is switched off.
ii. Transistors are used to store information (no refreshing is required)	ii. Capacitors are used to store data. (to be refreshed periodically)
iii. SRAM is faster compared to DRAM	iv. DRAM provides slow access speed and

v. It does not have refreshing unit	vi. It has a refreshing unit.
vi. These are expensive	vii. These are cheaper.
vii. low density devices.	viii. high density devices
viii. bit are stored in voltage form	ix. in electrical energy.
ix. Consume less power and generates less heat	x. Uses more power and generates more heat.
xi has lower latency, more resistant to radiation, higher data transfer rate, high-speed Cache memory	xi. more latency, less resistant, lower data transfer rate, lower-speed main memory
xii. These are used in Cache memory, used in high performance application (L1, L2, L3)	xii. Main memories, general purpose application

b) Explain Moores Law in Computer architecture

Moore's law is "the observation that" the number of transistors on an integrated circuit will double every two years with minimal rise in cost. Intel co-founder Gordon Moore provide this law in 1965. It state that the growth of microprocessors is exponential also.

- Transistor Density: Moores law primarily focuses on the number of transistors that can be placed inexpensively on an IC (now transistors are smaller fit on chip)
- Performance and Cost:
  - The increase in transistor density generally leads to a corresponding increase in the performance of microprocessors and other ICs, as more transistors typically means more processing power, memory, and

faster computations.

- The cost per transistor also decrease over time, making it cheaper to produce more powerful chips.

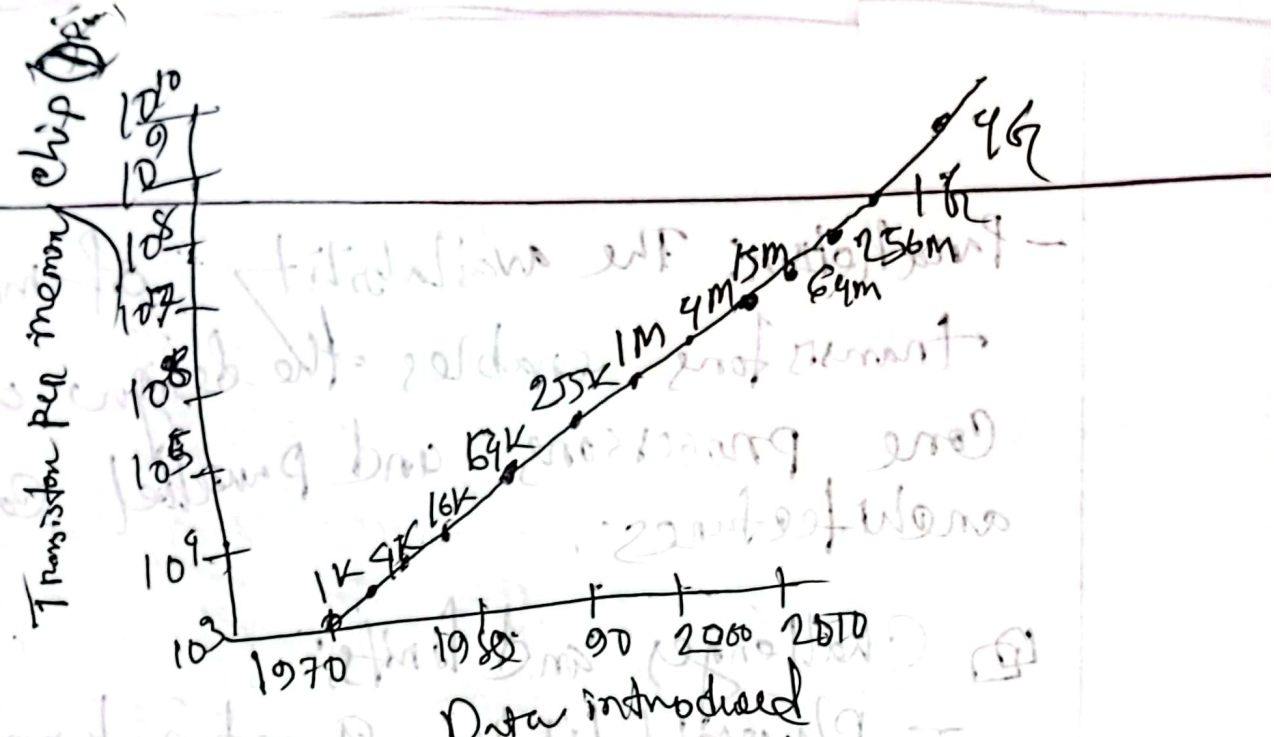
Implications for Computer Architecture

- Increased processing power: With more transistors available, processors can perform more complex computation and run more sophisticated algorithm.
- Energy efficiency: Smaller transistors generally consume less power, allowing for more energy efficient designs.
- Miniaturization: As transistors shrink, devices can become smaller and more portable.

- Parallelism: The availability of more transistors enables the design of multi-core processors and parallel computing architectures.

- Challenges and limits:
  - Physical limits: Quantum tunneling and heat dissipation.
  - Economic factors: The increasing cost of research and development for advanced semiconductor manufacturing processes.

- Beyond Moore's law:
  - The semiconductor industry is exploring new materials, technologies and architectures (quantum computing, neuromorphic computing, 3D stacking) to continue the trend of performance improvements even if traditional scaling slows down.



c) Regarding Cache design, write short notes on the following:

- hit time
- miss penalty

Hit time is the time it takes to access data from the cache when a requested data item is found there, known as a "cache hit". This metric is the time required to determine if the cache and to access the from the cache memory.

- Speed: Hit time is typically very fast because caches are designed to be accessed quickly compared to main memory.
- Components: It includes the time taken for -
  - The processor to send the address to the cache
  - The cache to check if the address is present (cache lookup)
  - The cache to access the data and deliver it to the processor.

  (Typical numbers
  1-2 clock cycle - L1
  5-20 " - L2)
- Importance: Minimizing hit time is crucial for high-performance systems, as cache hits should ideally happen frequently, given that accessing data from the cache is much faster than from main memory.

Miss penalty: refers to the additional time required to fetch data from a lower level of the memory hierarchy when a cache miss occurs. A cache miss happens when the data requested by the processor is not found in the cache, necessitating access to a slower, larger memory, such as main memory or another cache level.

Components: The miss penalty typically includes
- The time to send the request to the lower memory level
- The latency to access the data in the lower memory level
- The time to transfer the data back to the cache and/or the processor
- The time to possibly update the cache

2. a) In Interrupt handler, analyze the steps:
i. no interrupt ii) interrupt pending

An interrupt is a signal that prompts the operating system to stop its current tasks and execute a special routine known as an interrupt handler or interrupt service routine (ISR)

i. No Interrupt: The system operates normally without any interrupt signals being received.

- Normal Operation -

1. The CPU executes instructions sequentially from the main program or the currently running process.
2. The interrupt controller monitors for interrupt signals from hardware devices or software.
3. The system's control flow is uninterrupted, meaning that the CPU fetches, decodes and executes instructions from the

main memory or cache.

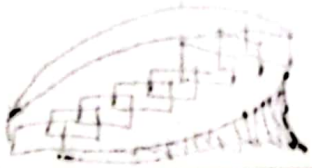
- Polling and Monitoring

- The interrupt controller or specific hardware components continue to monitor for any potential interrupt requests.
- In system that use polling instead of interrupts, the CPU periodically checks the status of various devices to see

- Idle State:

- If the system is idle the CPU might enter a low-power state to conserve energy. The CPU will wake up from this state if an interrupt occurs.

- The system maintain its normal execution flow with no special action required for handling interrupts;



Interrupt Pending: An interrupt signal is received, and the system needs to handle it.

1. Interrupt Detection:
   - The interrupt controller detects an interrupt signal from a hardware device (keyboard, mouse) or a software condition.
   - The controller sends an interrupt request (IRQ) to the CPU, indicating that an interrupt is pending.
2. Interrupt Acknowledgment:
   - The CPU finishes executing the current instruction and acknowledges the interrupt.
   - If interrupts are enabled, the CPU saves the current state of execution, including the program counter, registers, and other necessary context, to resume execution later.

3. Interrupt vector and ISR

b) How much memory space 8080 will provide that has 16 bit address bus?

The Intel 8080 microprocessor uses a 16-bit address bus. This means it can address memory locations with 16 bit wide addresses.

Total Memory space $= 2^{\text{Num\_of\_Address lines}}$

for the 8080, with a 16 bit address bus

$= 2^{16} = 65,536$ bytes

$= \frac{65,536}{1024}$ KB

$= 64$ KB

Thus the Intel 8080 microprocessor with a 16 bit address bus can address up to 64KB of memory.

(c) Describe with figure : Asynchronous timing diagram for system bus Read/write bus.

Asynchronous timing diagrams illustrate the sequence of control signals and data flow in a system where operations are not synchronized with a common clock signal

3. a) Analyze the concept of "Write Back" in order to avoid the bottleneck of "Write Through".

The Concepts of "Write Back" and "Write Through" are primarily associated with caching mechanisms in Computer systems (memory management). Here's an analysis of the "Write Back" strategy and how it can help avoid the bottleneck associated with "Write Through".

Write Through:

In a "Write Through" caching strategy, every time data is written to the cache, it is simultaneously written to the backing store (main memory). This ensure that the data in the cache and the backing store are always consistant.

- Performance bottleneck: Since every write operation requires a write to both the

Cache and the main memory, it can significantly slow down performance (heavy workloads). The system must wait for the slower main memory write to complete before proceeding.

- Increased latency: The time taken for write operations increases, leading to higher latency for applications that require frequent updates to data.
- Resource Contention: Frequent writes to the main memory can lead to contention for memory bandwidth, further degrading performance.

Write Back: Every time data is written to the cache, it is simultaneously written to the backing store (main memory).

Write Back: allows data to be written to the cache only, with the write to the backing store occurring later.

- Reduce latency: Since writes are only performed to the cache, the system can return control to the application more quickly, reducing the overall latency of write operations.
- Batching Writes: The system can defer writing to the backing store until necessary (when the cache line is evicted), allowing multiple writes to be combined into a single write operation. ("reduce write operation of main memory)
- Improved Performance: By minimizing the number of writes to the slower backing store, "write back" can improve overall system

Performance, especially in scenarios with frequent updates.

- Cache Coherency: While "Write Back" can lead to inconsistencies between the cache and the backing store, modern system implement mechanisms (like dirty bits) to track which cache lines have been modified. This ensure that data is eventually written back to the main memory when necessary, maintaining data integrity.

Avoiding Bottlenecks:

- Asynchronous Writes: By allowing writes to be asynchronous, the system can continue processing other tasks while waiting for the write-back to occur, improving overall throughput.

- Optimized Memory Access: With fewer write operations to the main memory the system can optimize memory access patterns, reduce latency and improving performance.

- Batching Writes: Write back caching can accumulate multiple write operations and execute them in batches. (reduce the frequency of write operation in main memory) ~~minimizing~~

- Reduced Write Amplification: By only writing back modified data, write back caching can significantly reduce write amplification. (important in SSDs, faster wear and reduce lifespan).

- Improved Data locality: keeping frequently accessed data in

the cache allows for quicker access and reduces the need to fetch data from slower storage.

b) Let's design a cache of 64KB. Main memory is 16 Mbytes. Cache block is 4 bytes (2 bit word identifier) and rest of bits is block identifier. Use the concept of direct mapping address

Given -

Cache memory size = 64 KB
Main memory size = 16 MB
Cache block size = 4 B

Step 1: Calculate the number of cache blocks:

$= (64 \times 1024) / 4$

$= 16,384$

Step 2: (2)

Number of Bits for Indexing
= log2(Number of cache blocks)
= 16,384 = 2^{14}
= 14 bits for the index

Step 3:

Number of Bits for the block offset

Since each block is 4 bytes, we need 2 bits to identify the words within the block: 2 bits.

Step 4: ~~Number of Bits for the Block offset~~ = Total address Space =

The main memory is 16 MB,

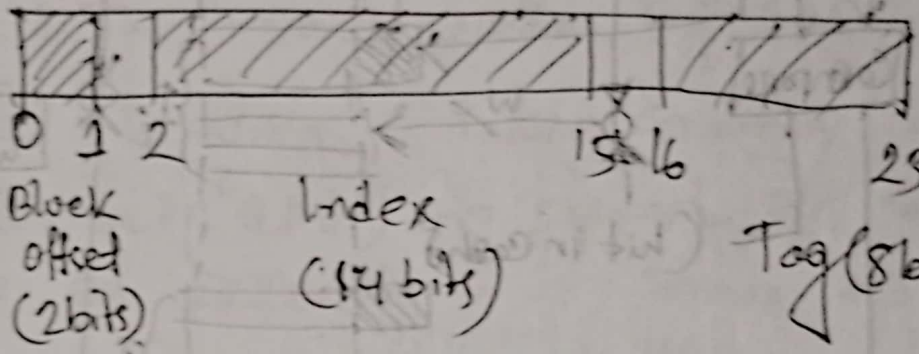
= 16 × 1024 × 1024
= 16777216

TAS = log2(16777216 = 2^{24} = 24

Step 5: Tag size = Add size − Index size − Block offset size

= 24 − 14 − 2

= 8



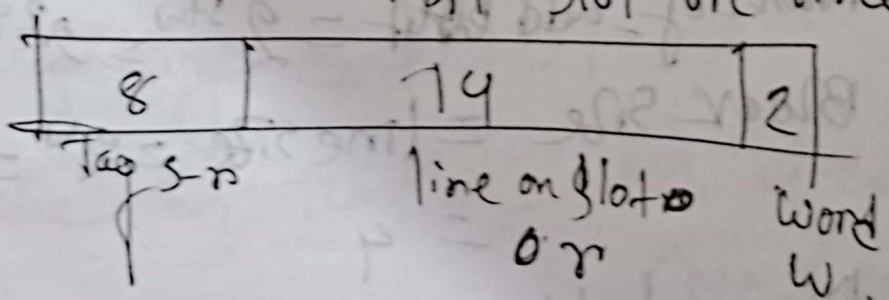
24 bit address

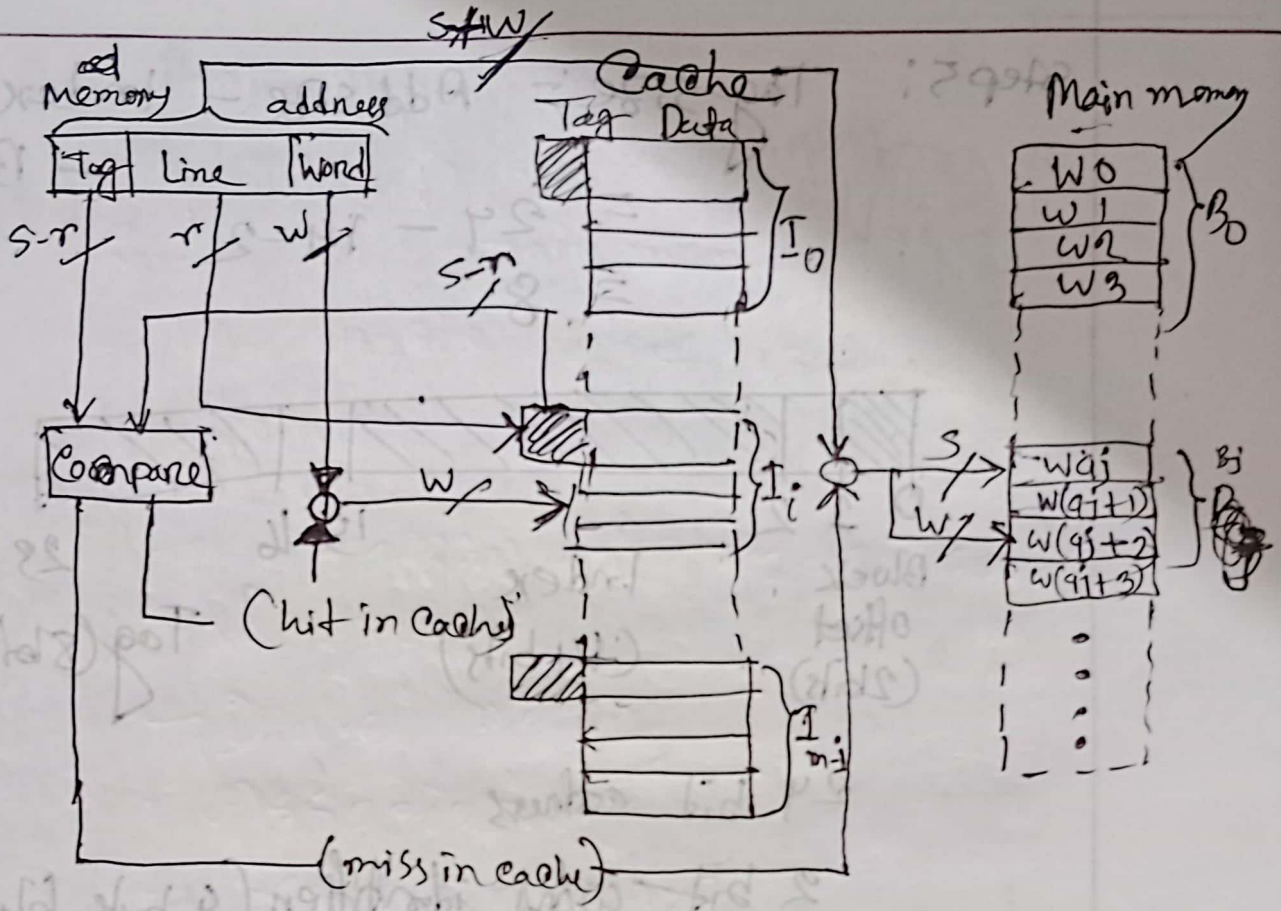
2 bit word identifier (4 byte block)

22 bit block identifier

8 bit tag (22-14)

14 bit slot or line





Address length = (s+w) bits = 22+2 = 24 bits

Num_of_addressable = $2^{s+w} = 2^{24} = 16$ MB

Block Size = line size = $2^w = 2^2$

$= 4$

Num_of_blocks in main memory = 2^s

$= 2^{22}$

Num. of lines in cache = $m = 2^r$

Size of cache = words or bytes

4. a) What are the difference among direct mapping, associative mapping and Set associative mapping?

Cache: The small section of SRAM memory, added between main memory and processor (CPU) to speed up the process of execution, is known as cache memory. It includes a small amount of SRAM & more amount of DRAM. It is a high speed & expensive memory.

Cache hit ratio: It measures how effectively the cache fulfills the request for getting content.

Cache hit ratio = No. of cache hit / (No. of cache hit + No. of cache miss)

If data has been found in the cache it is a cache hit else a cache miss.

4. a) What are the difference among direct mapping, associative mapping and Set associative mapping?

Cache: The small section of SRAM memory, added between main memory and processor (CPU) to speed up the process of execution, is known as cache memory. It includes a small amount of SRAM & more amount of DRAM. It is a high speed & expensive memory.

Cache hit ratio: It measures how effectively the cache fulfills the request for getting content.

Cache hit ratio = No. of cache hit / (No. of cache hit + No. of cache miss)

If data has been found in the cache it is a cache hit else a cache miss.

Cache Mapping:

The process / technique of bringing data of main memory blocks into the cache block is known as cache mapping.

The mapping techniques can be classified as:

- Direct mapping
- Associative
- Set Associative

⊞ Direct mapping: Each block from main memory has only one possible place in the cache organization in this technique

e.g: every Block i of the main memory can be mapped to block j of the cache using the formula:

$j = i \bmod m$

where, i = main memory block num

j = cache block num

m = num of Block in the cache

The address here is divided into 3 fields : Tag, Block & Word.

b) Assume that there are three (03) small caches, each consisting of eight (08) one word blocks. One cache is fully associative, a second is two-way set associative. and third is direct mapped. Now. find the num of misses of each cache organization given the following sequence of block address:

0, 8, 0, 6, 8

* Direct-mapped cache

Address of memory block accessed	Hit or miss	Content of cache block after reference			
		0	1	2	3
0	Miss	Mem[0]			
8	Miss	Mem[8]			
0	miss	Mem[0]			
6	Miss	Mem[0]		Mem[6]	
8	Miss	Mem[8]			

i=j modulov

Block add	Cache Set
0	0 modulo 2 = 0
6	= 0
8	= 0

(2) way Set associative cache

Address of memory block address Accessed	Hit on miss	Content of cache block after reference			
		set. 0	set. 0	Set 1	Set 1
0	Miss	mem[0]			
8	Miss	mem[0]	mem[8]		
0	Hit	mem[0]	mem[8]		
6	miss	mem[0]	mem[6]		
8	miss	mem[8]	mem[6]		

The 2way has 4 misses, one less than the direct mapped cache

Fully associative cache

		Block 0	Block 1	Block 2	Block 3
0	Miss	mem[0]			
8	Miss	mem[0]	mem[8]		
0	Hit	mem[0]	mem[8]		
6	miss	mem[0]	mem[8]	mem[6]	
8	Hit	mem[0]	mem[8]	mem[6]	

The fully associative cache has the best performance with only 3 misses

c) List and briefly define three (03) techniques for performing I/O

It is the technique of communication between memory and I/O devices. I/O techniques are catagorized in three types based on how information is transfer btw memory and I/O devices that whether it is using CPU interaction or interrupt interaction.

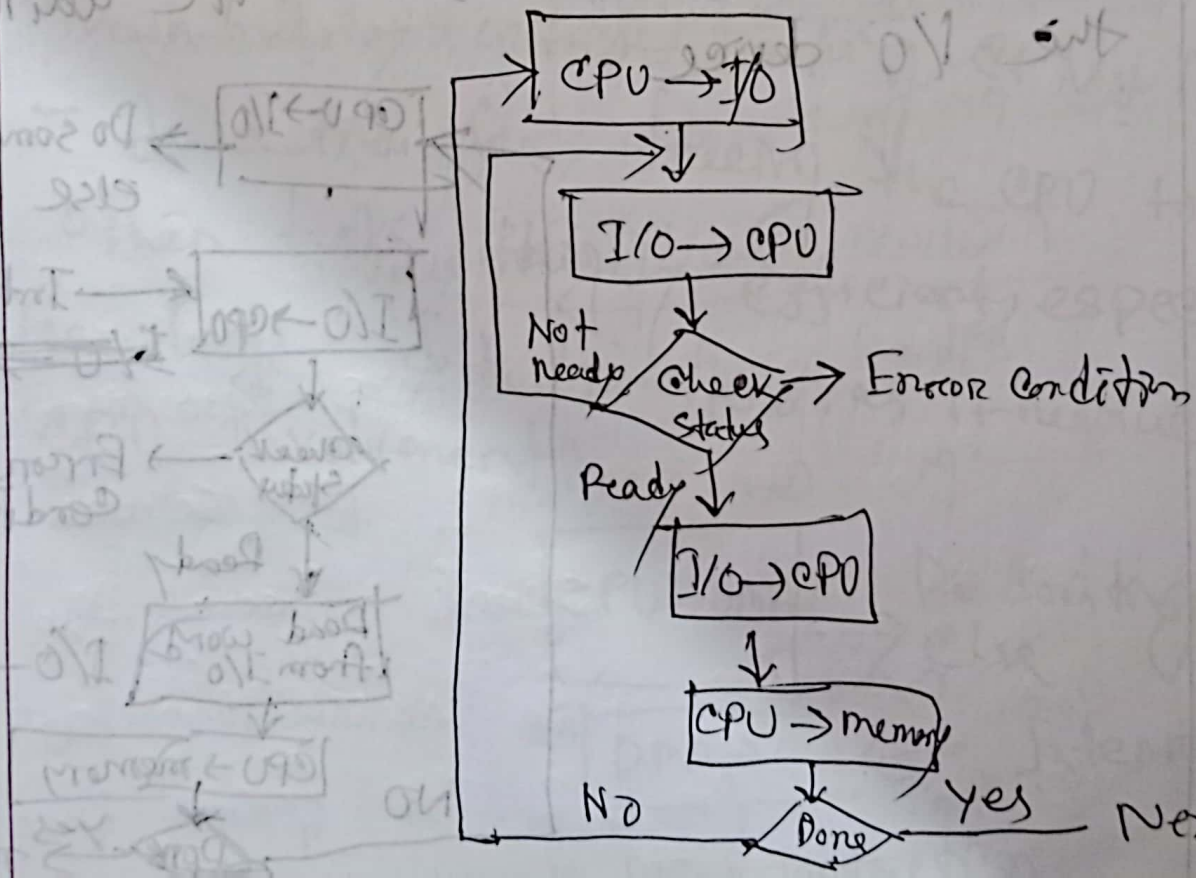
[Users interact with the computer sys through input and output (I/O) devices, such as keyboard, mouse, monitor and so on. I/O devices are also called peripherals. exchange information btw ~~I/O devices and~~ User & CPU.]

[I/O module is intermediate between I/O devices and CPU (I/O devices cannot directly sys with sys buses)
i. Control & Timing
ii. Processor Communication]

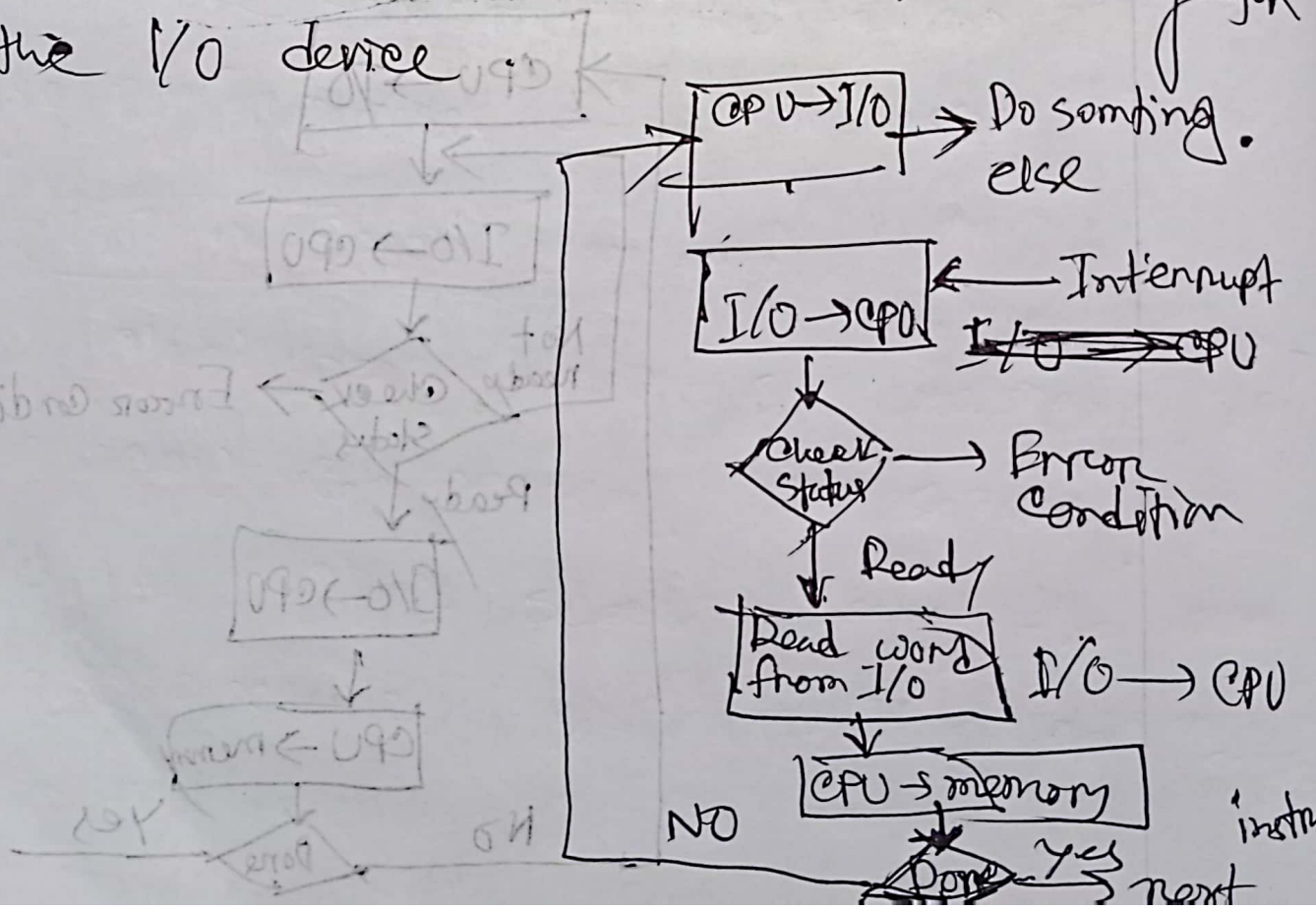
Techniques of I/O:

- Programmed I/O: The CPU issues a command then wait for I/O operation to be complete. The CPU is fasten than the I/O module then method is useful.

In this technique, the CPU continuously checks the status of an I/O device to see if it is ready to transfer data. The CPU actively wait until the device is ready, then performs the data transfer.

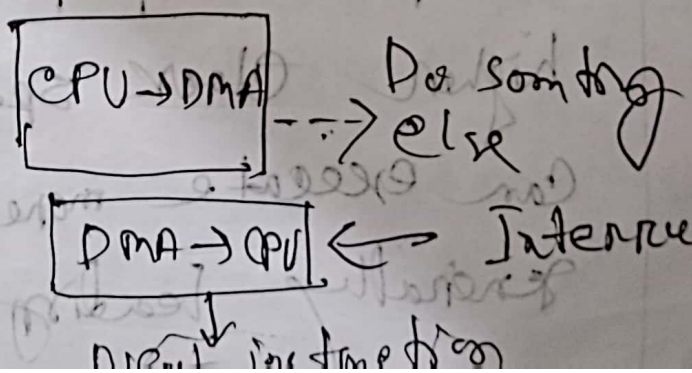


Interrupt Driven I/O: The CPU issue command then proceeds with its normal work until interrupted by I/O device on completion of its work. In interrupt driven I/O, the CPU not continuously check the I/O device. So, the I/O devices sends an interrupt signal to the CPU when it is ready for data transfer. This methods is more efficient than Programmed I/O as it allows the CPU to perform other tasks while waiting for the I/O device.



- DMA: In this CPU and I/O module exchange data without involvement of CPU.
- Memory mapped I/O: Memory and I/O are treated as memory only, It means no signal like IO/M.
- Isolated I/O: Address space of memory I/O is isolated It uses IO/M signal.

Direct Memory Access (DMA): I/O devices directly transfer data to or from the main memory without involving CPU. Manages data transfer, freeing the CPU to perform other tasks. Highly efficient, especially for large data transfers, as it reduce the CPU involvement.



5. a) Explain the performance of a processor.

CPU performance refers to the effectiveness and efficiency of a computers central Processing Unit in executing instructions. High CPU performance implies that a computer can process a large number of instructions per unit time, enabling it to operate more quickly and handle more simultaneous tasks. There are several key factors influence the performance of a processor:

1. Clock Speed (Frequency):

Clock speed indicates how many cycles a CPU can perform per second. A higher clock speed means the CPU can execute more instructions per second, generally leading to faster performa-

nce. measured in Gigahertz (GHz)

ii. Num_of_Cores: Modern CPUs have multiple cores, with each core capable of executing its own sequence of instructions, handle multiple tasks (Parallel processing), improve performance (multitasking, multi-threaded applications)

iii. Threads: are the smallest unit of processing that can be scheduled by an OS. A CPU with multiple threads per core can handle more tasks concurrently, improving performance in parallel work loads.

iv. Instruction Set Architecture (ISA): ISA defines the set of instructions the CPU can execute. Some processor have instruction set that allow them to perform tasks more quickly or efficiently.

v. Cache size: Cache is a small, fast memory located on the CPU that stores frequently accessed data & instructions. Larger cache sizes reduce the need to fetch data from slower RAM, improving performance.

vi. Pipeline Depth: allows the CPU to work on several instructions simultaneously at diff stages of execution. Deeper pipelines can improve performance but introduce latency if instructions depend on each other.

vii. Thermal Design Power (TDP): represents the max amount of heat the CPU is expected to generate.

viii. Power efficiency: Power efficient processors can deliver better performance per watt of power consumed.

ix. Fabrication Process: The size of transistors on the CPU, often referred to as the process node (nanometers)

x. Memory Bandwith: The speed at which data can be transferred btw the CPU and RAM, impacts performance.

xi. Instruction level Parallelism (ILP): refers to the CPU's ability to execute multiple instructions simultaneously: (Out-order execution & Superscalar architecture improve ILP)

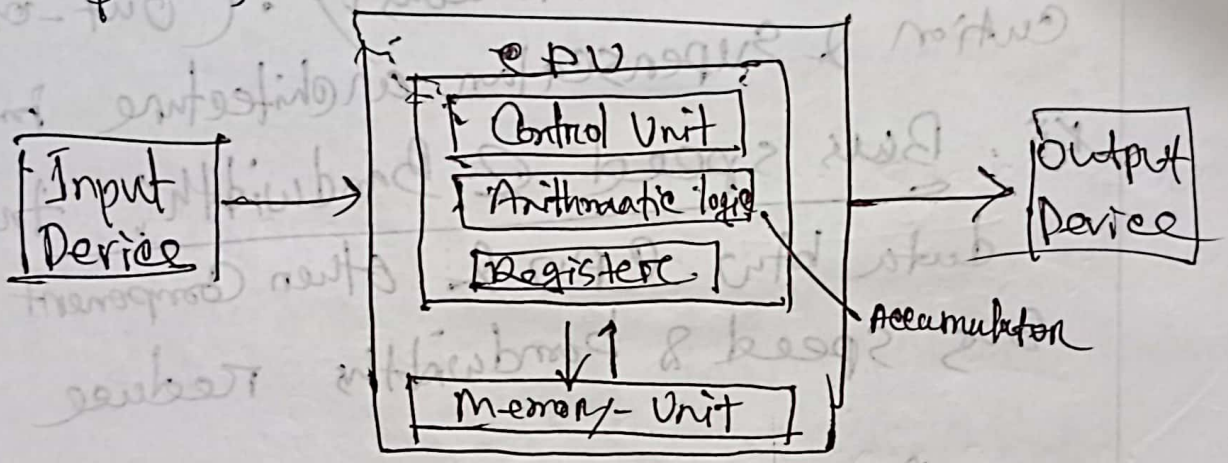
xii. Bus speed & Bandwidth: transfer data btw CPU & other component. Faster Bus speed & bandwidths reduce bottleneck

xiii. Microarchitecture: The design of the CPU's internal components plays a crucial role in determining how efficiently the processor performs tasks.

xiv. Workload & Software Optimization: Performance can very depending on the type of workload and how well the software is optimized for the CPU.

(b) Von Neumann computer architecture is the basic computer architecture for the modern computer. Explain this.

⇒ is a fundamental computer architecture that serves as the basis for most modern computers.



CPU: The CPU is the brain of the computer; responsible for executing instructions. It typically consists of two main parts.

ALU: Performs Arithmetic and logic operations (Sub, Add etc).

CU: Directs the operation of the processor; fetching instruction from

memory, decoding them & executing them.

Memory :- (RAM) is used to store both the data and the instructions that the CPU needs to execute. (single memory space for both data & instructions)

I/O devices:- These devices allow the computer to interact with the external world, receiving input from users and sending output to display devices, storage. (keyboard, monitor, mouse etc)

Registers :- Registers refer to high-speed storage areas in the CPU. The data processed by the CPU are fetched from the registers. Types -

i. Accumulator - Store the results (from ALU)

ii. Program Counter :- track of the memory

iii. Memory Add Register; Store memory location Add.

iv. Memory data Register: Store instruction fetched from memory

v. Current Instruction Register (CIR): Store the most recently fetched instruction.

vi. Instruction Buffer Register (IBR): The instruction not be executed.

Busses – A communication system that transfer data btw the CPU, memory & I/O devices.

Types:

i. Data Bus: Carries data among the memory unit, I/O device & the processor

ii. Address Bus: Carries the add of data btw memory & processor

iii. Control Bus: Carries control commands from the CPU.

Key Principles:

i. Stored-Program Concept: the program and data are stored together in the same

memory space.

Sequential Execution: The CPU to fetch & execute instructions sequentially.

1. Fetch
ii. Decode
iii. Execute

Single Memory Space: The same memory is used to store both data and instruction.

Bottleneck:

The CPU can only fetch one instruction or piece of data at a time, the speed at which data can be transferred btw the CPU & memory becomes a limiting factor in performance.

Processors become faster but memory speeds do not increase at the same rate. It can cause delays & reduce the overall efficiency.

divided into

Advantages:

- Simplicity: Single memory space for instruction and data, making it easier to implement and understand.
- Flexibility: Programs can be modified by simply changing the instructions in memory, allowing software to be run on the same hardware.
- Cost effective: The architecture's simplicity & efficiency make it...

Limitations:

- Bottleneck Issues: Can slow down the processing speed.
- Security: Since instructions and data share the same memory, there is a risk,

Modern Variations & Enhancements:

- Harvard Architecture: reduce bottleneck
- Pipelining: overlap fetch, decode, execute
- Caching: reduce the time needed to fetch them from the main memory.

(c) Describe the multiplexed bus with advantages & disadvantages.

A multiplexed bus is a communication system that transmits multiple signals over a single data path or channel. In a computer system, a bus is a set of wires or traces on a circuit board through which data, addresses, and control signals are transmitted between different components like the CPU, memory, and peripherals. When the bus is multiplexed, the same set of physical lines is used to transmit different types of information (such as data, address and control signals) at different times, reducing the number of lines required.

How a multiplexed Bus Works:

— Time Division multiplexing: The bus is divided into time slots, allocated to

to a different type of signal. For instance, during one clock cycle, the bus may carry address information, and in the next cycle it may carry data.

- Bus Control logic: The system's control logic ensures that the correct signals are placed on the bus at the right times, coordinating the transfer of information between components.

Advantages:

i. Reduce Pin Count: Because fewer lines are needed, the number of pins on connecting integrated circuits (ICs) is reduced, which simplifies the design and reduces costs.

ii. Lower Cost and Complexity: Fewer physical connections means that the

Overall complexity of the circuit board layout is reduced, which can lower manufacturing costs and improve reliability.

iii. Space Savings: Fewer wires or traces on the circuit board means that the design can be more compact, which is particularly beneficial in systems where space is at a premium.

iv. Flexibility: allows for more flexibility in the design and allocation of bus resources.

- Communication requires only two bus lines (wires)
- A simple master/slave relationships exist between all components
- No strict baud rate required
- Hardware is less complicated than UART
- Supports multiple masters & multiple slaves

- ACK/NACK bit gives confirmation that each frame is successfully transferred
- I2C is a "true multi-master bus" providing arbitration and collision detection
- Each device connected to the bus is software addressable by a unique address
- Most I2C device communicate at 100kHz or 400kHz.

Disadvantages:

i. Increased latency: Since multiple types of information share the same bus, time must be allocated for each type, leading to delays, slow down overall sys performance (high-speed applications)

ii. Complexity in control logic: The control logic required to manage a multiplexed bus is more complex. (design complexity means that the

power consumption :)

iii. Potential for Data Collisions: If the control logic fails/ timing is not managed correctly then collisions will occure, (multiple signals attempt → leading errors)

iv. Limited Bandwidth: The total bandwidth of the bus is shared among different signals (burden → high speed data transfer)

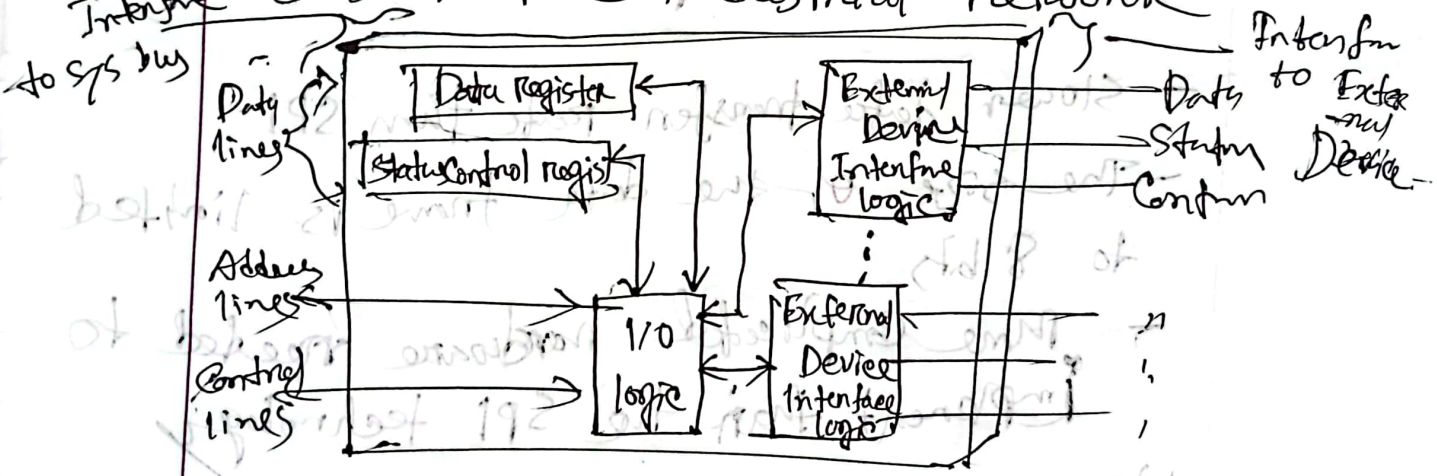
- Slower data transfer rate than SPI
- The size of the data frame is limited to 8 bits
- More complicated hardware needed to implement than the SPI technology

Use Cases:

1. Embedded Systems: used in embedded sys where space & cost contraints are significant
ii. Micro controller: Use in multiplexed busses to connect the CPU to memory, Peripherals, reduce num-ch ... in synchronization

6. a) I/O module necessity ?

I/O modules, manage the communication btw a CPU and a network, including the transfer data, the management of power loads, and the control of machine functions. This enables system integrators to connect disparate devices, allowing greater control of the industrial network



necessity :

1. I/O devices are most of case usually electrical/mechanical device

where processor is an electronic device. Also the data transfer rates of I/O are often slower than the processor and memory. So it is significant that the speed and electrical characteristics of I/O are different from CPU.

iii. There are a variety of peripherals that exist and may ~~be difficult~~ need to be connected to the same system bus. But it may be difficult to incorporate all the peripheral device logic into CPU. This reduce flexibility & creates hinderances in new developments

iv. Peripheral often use different data formats and word lengths that used by the CPU.

- Communicate with external Devices (Interfacing, data Conversion)
- Control of external hardware (Actuators & motors)
- Timing and Synchronization)

- System expansion & flexibility (Scalability & modularity)
- Isolation & Protection (Electrical Isolation fault tolerance)
- Improved performance (Dedicated processing parallel processing)

b) In direct memory access (DMA) operation, describe about "cycle stealing".

DMA is a feature of computer systems that allows certain hardware subsystems to access main system memory independently of the CPU. Cycle stealing is a technique used in DMA operation where the DMA controller temporarily takes control of the system bus to transfer data directly between an I/O device and the system memory, without

involving the CPU for each byte.