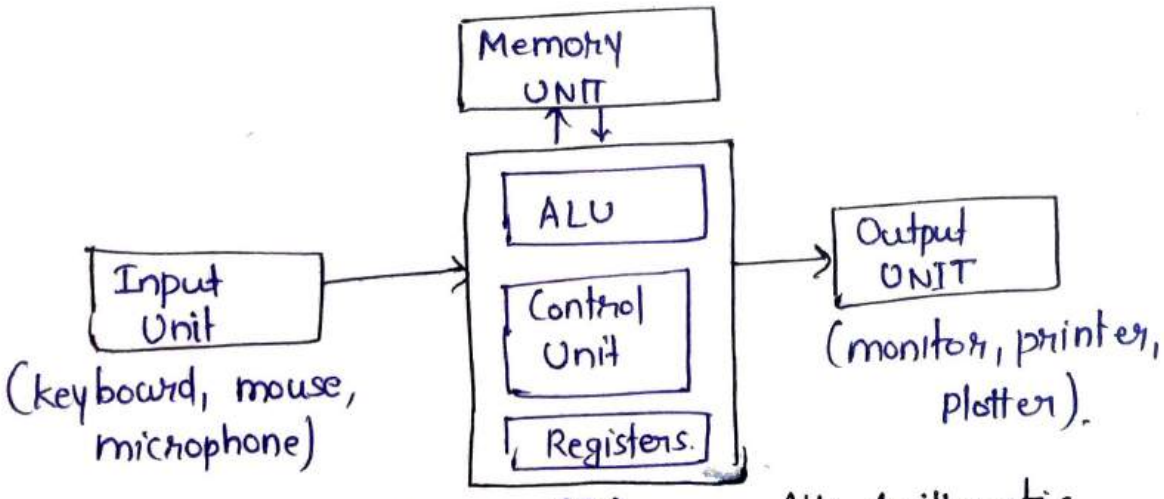
Computer Architecture & Orgnization

UNIT-I

Structure of Desktop computers



ALU - Arithmetic & Logic Unit.

Memory
- Primary (RAM) (Fast small, expensive)
  - → store programs & active data.
- Secondary (large, slow, cheap)
  - ↓
  - magnetic tapes, magnetic disks

ALU
- add subtract, division, multiplication (arithmetic operation)
- (Logical operations) (AND, OR, Inverting).

Control Unit
- Control & Coordinate activities amongst all functional units.
- function
  - fetch instruction from memory.
  - decode (identify the operation)
  - execute instruction (sending control signals to corresponding unit.

Stack implementation

- Register stack.
- Memory stack.

Register stack.

It is organized as a collection of finite number of CPU registers.

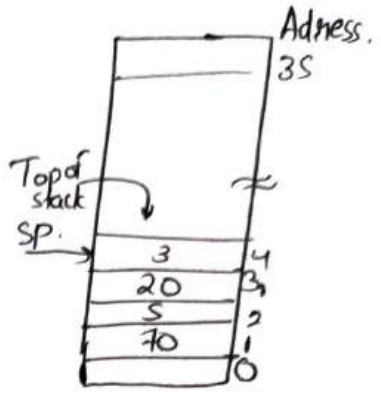


Fig: word Register stack.

- stack pointer holds the address of the register that is currently the top of stack.
- In above figure, 4 elements are stored in stack
- Data element 3 is top of stack, therfore content of SP=4.
- stack pointer is a 5 bit register, because there are 32 registers in stack ($2^5=32$)
- SP=0 means stack is empty
- When data element is pushed on the stack, SP incremented.

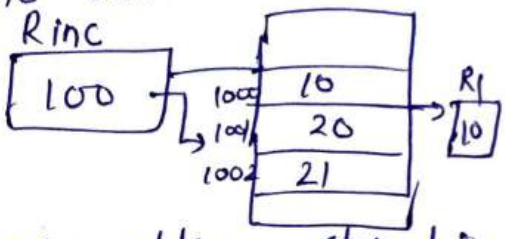
Memory stack

- It is implemented using computer memory.
- operation of memory stack is exactly similar to register stack.
- Number of registers in the CPU is limited & it restricts the size of stack.
- Using memory, size of stack can be extended upto size of memory.

– Or Address of the operand is in a register whose value is incremental / decremented after fetching the operand from the address.

Ex: Load (R inc)+, R_1

① $R_1 \leftarrow M[R_{inc}]$



Load the contents of the memory address stored in register R inc.

② $[R_{inc}]_{++}$ or $R_{inc} = R_{inc} + 1$

↳ increment the contents of Rinc. (memory address)

OR INR R_1 will increment the Register R1

-When the address stored in the register refers to a table of data in memory, it is necessary to increment / decrement the register after every access to the table.

-This can be achieved using increment or decrement instructions.

DCR R2 -will decrement the register R2

Auto decrement.

Ex Load $(R_{dec})-, R_1$

② $R_1 \leftarrow M[R_{dec}]$

$[R_{dec}]_{--}$ or $[R_{dec}] \leftarrow [R_{dec}] - 1$ [Before access of execution of instruction]

★ This mode is specify useful when we want to access a table of data.

① Arithmetic instructions -

Name	Mnemonic	Description.
Increment	INC	Add 1 to the value stored in a register or memory word
Decrement	DEC	Sub 1 from the value stored in a register or memory word.
Add	ADD	ADD 2 number (binary, flt pt, deci)
substract	SUB	Subtract 2 no.
Mutiply	MUL	multiply 2 no
Divide	DIV	DIV 2 no.
Add withy carry	ADDC	Performs the addition of 2 operand plus the value of the carry from previous computation.
Subtract with borrow	SUBB	Sbtracts 2 words & a borrow which may have resulted from previous subtract operation
Negtive 2's Component	NEG	From the 2's complement of a number effectively.

② Logical & Bit Manipulation instructions.-

Name	Mnemonic	Description.
Clear	CLR	0's transferred to the destination to clear selected bits.
Complement	COM	
AND	AND	
OR	OR	
Exclusive OR	XOR	
Clear carry	CLRC	
set carry	SETC	
Complement carry	COMC	
Enable Interrupt	EI	- Enable interrupt flip flop that control is the interrupt facility is enabled.
Disable interupt	DI	- Flipflop that controls the interrupt facility is disabled.

- The pupose of micro progam sequencer is to present an address to the control memory so that a microinstruction may be read & executed.
  - Here multiplexer selects an address from 4 sources & routes it into control address register.
  - The output from CAR provides the address for the control memory.
  - The contents of CAR are incremented & applied to the multiplexer & to the stack register file.
  - The register selected in the stack is determined by the stack pointer.
  - Inputs I_2, I_1, I_0 & T (test) derived from CD & BR fields of microinstruction specify the operation for the sequencer.
  - They specify the input source to the multiplexer also generate push & pop signals required for stack operation.

Typical Sequencer operations are

- Increment
- Branch.
- Jump.
- Call & return from subroutine
- load an external address
- push or pop the stack & other address sequencing operations.

- The stack pointer is a 3 bit register & it gives the address of stack register file consists of ($2^3 = 8$) eight registers.

③ If it is, then the instruction is fetched from the cache - a very fast process (Cache-hit)

④ If not, the CPU has to fetch next instruction from main memory - a much slower process (cache miss)

Hit Ratio – The performance of cache memory is measured in terms of quantity called hit ratio.

- When the CPU refers to memory & finds the word in cache it is said to provide a hit.
- If the word is not found in cache, it is in main memory then it counts as a miss.
- The ratio of the number of hits to the total CPU refrencess to memory is called hit ration.

$$\text{Hit ratio} = \frac{\text{Hit}}{\text{Hit} + \text{miss}}$$

CACHE MAPPING TECHNIQUES.

Cache memory bridges the speed mismatch b/w processor / CPU & the main memory.

When cache
- Hit occurs - requires word present in cache memory
  - word delivered to the CPU from cache memory.
- Miss occurs - required word is not present in the cache memory.

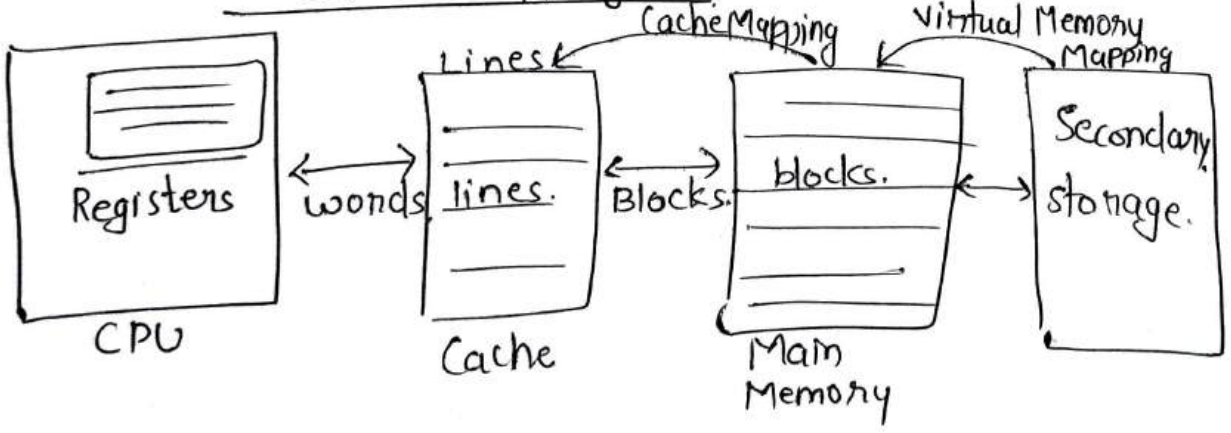
So when cache miss occurs, the page containing the required word has to be transferred from the main memory to cache memory.

- The transformation of data from main memory to cache memory is called as mapping process.

This mapping is performed using cache mapping techniques.

Defⁿ Cache mapping is a technique by which the contents of main memory are brought into the cache memory.

Cache Mapping



Mapping Process.

During cache mapping, block of main memory is simply copied to the cache & the block is not actually brought from the main memory.

There are three types of cache Mapping Techniques.

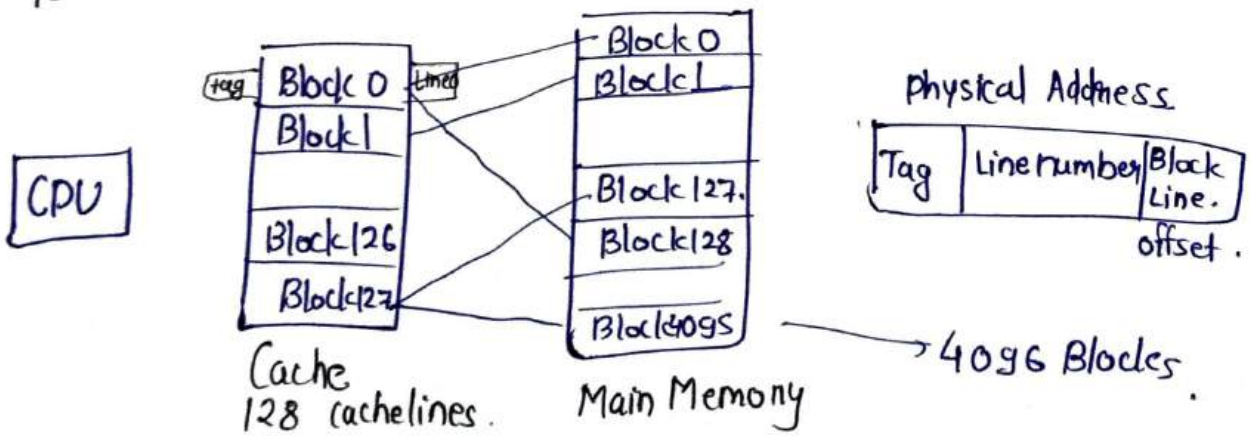
1. Direct Mapping.
2. Fully Associative Mapping.
3. k-way set associative Mapping.

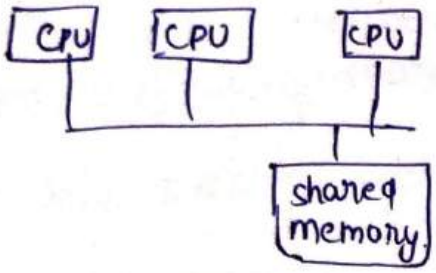
1. Direct Mapping- In direct mapping, a particular block of main memory can map only to a particular line of the cache.

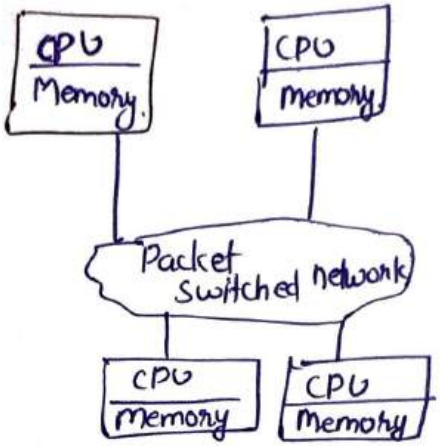
- The line number of cache to which a particular memory block can map is given by -

$$\text{Cache line number} = (\text{main memory Block Address}) \text{ Modulo } (\text{Number of lines in Cache})$$

Ex. Consider cache memory is divided into n number of lines then, block 'j' of main memory can map to the line number (j mod n) only of the cache -







Difference between Tightly Coupled and Lossely coupled multiprocessor.

Tightly Coupled	Loosely Coupled.
① Tightly coupled multiprocessor has shared memory	① Loosely coupled multiprocessor has distributed memory.
② The degree of coupling between processor is high	② The degree of coupling blw processors is low.
③ In tightly coupled multi-processors CPU's connected in close communication.	③ In loosely coupled multi-processors CPU/systems located at different locations.
④ In this system, CPU's share computer Bus, memory & I/O devices.	④ In this system there is no such sharing.
⑤ It is efficient when high degree of interaction between processes & for high speed & real time processing.	⑤ It is efficient when tasks running on different processors has minimal interaction blw them (parallel application)
⑥ Memory conflict occurs due to use of shared memory.	⑥ No memory conflit occurs.

⑦ Processors communicate with each other using shared memory.	⑦ Processor communicate by using message passing Techniques.
⑧ Data rate is high	⑧ Data rate is low
⑨ More expensive	⑨ Less expensive.
⑩ Compact in size	⑩ Larger in size.
⑪ Less scalable - Limited number of CPU can be added	⑪ Scalability is high. - more no. of CPU's can be added to improve the system performance.
⑫ Less fault to learnt - If shared memory corrupted then causing all processor to fail.	⑫ more fault to learnt - A fault in a single system/ module does not lead to a complete system breakdown.
⑬ Less complex	⑬ Complex due to additional hardware is required to provide communication between individual processors.
⑭ Portable	⑭ Less portable.

Now P_i makes an exit from the CS, it will execute "signal" operation $\boxed{S=1}$.

Then one of the processes, looping in "while" statement of "wait" operation → $\boxed{S=0}$ - enter its CS.

As "wait" operation is to be executed automatically, so other waiting processes will find the $\boxed{S=0}$ & continue to loop in the "while" loop.

So, at a time only one of the cooperating process can enter CS, subject to satisfaction of the condition that "wait" operation is executed automatically.

★ So the requirement of mutual exclusion is met.

(ii) Progress - $\overset{S}{\boxed{1}}$ - no process is executed in CS.

One of the waiting processes looping in the "while" loop of "wait" operation will find the $\boxed{S=1}$, exit from the "while" loop decrement the semaphore to 0 & enter CS.

So if no process is executing in CS & some are waiting to enter, then one of the waiting processes will enter its CS immediately. Thus the requirement of progress is met.

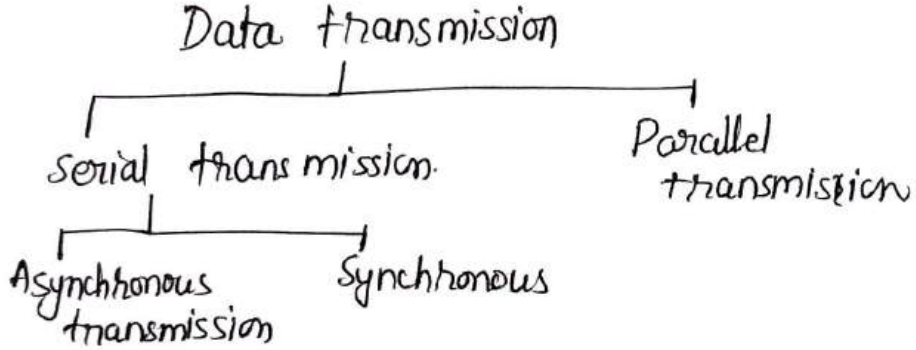
(iii) Bounded Waiting -

Arbitrary, one of the waiting processes, will get entry into its critical section, when a cooperating process, executing in its critical section exists.

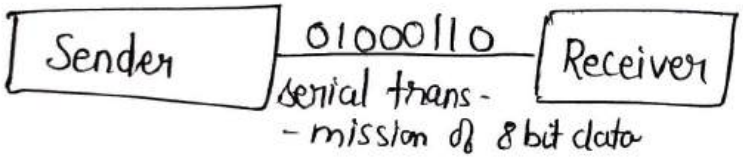
DATA TRANSMISSION

Data transmission refers to the process of transeferring data blw 2 or more digital devices.

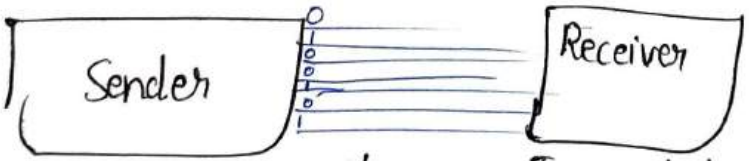
- Data tramitted from 1 device to another in analog or digital format.
- Data is transferred in the form of bits blw 2 or more digital devices.



Serial transmission



- Parallel transmission.



simultaneous Transmission of 8 bit Data

Serial data transmission	Parallel Data transfer
① Serial data transmission sends data bits one after another over a single channel.	① Parallel data transmission sends multiple data bits at the same time over multiple channels.