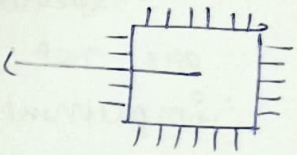
HPC : High Performance computing

Moore's law : No. of transistors on a microchip doubles every two year. [1965]

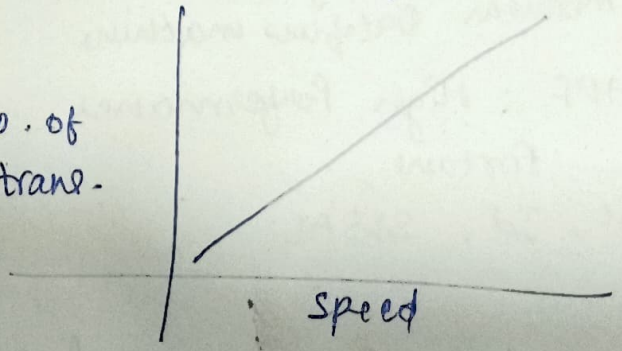
- It claims that speed & capability of our computers to increase every two yrs. & cost would be halved.
- Growth is expotential.
- With time the machinary size will be decreased, faster & cheaper with time.



contains carbon & silicon which runs electricity faster.

→ Ending of Moore's law ? ←

1. By 2020. it was belived that the physical limits of computer will come.
2. Cooling of transistor requires more energy



The Computational law Argument

or

→ Moore's law 1975 ←

In this law, moore revised the time frame from 24 yrs to 18 months.

↳ The computational power argument says that the logical resources rely on both

implict parallelism

↳ Seperate function calls are not required to implement parallelism,

↳ Ex: Pipelining, Vectorization, Multithreading

↳ Lang:

1. Axum
2. BMDFM (Binary modular Dataflow machine
3. HPF : High Performance Fortan,
4. SAC, Id, SFSAL

explicit parallelism

↳ extra programming efforts are needed to tell the compiler to use the resources parallely

↳ Parallel loops
↳ Message passing
↳ Data Parallism

↳ Lang

1. Occam
2. Erlang
3. Java.

Serial processors delays on implict parallelism

We can achieve parallelism in two ways

Parallelism
↙ ↘
Implicit- Explicit-

→ Memory Risk Argument ←

Clock Rates → 40% ↑ / year while.
DRAM → 10% ↑ / year
results in performance issues.

Here principle of locality works

↳ frequently access the same set of memory at same location

→ Data Communication Argument ←

- Large Volume of Data (→ in databases & data mining.) is not easy to move from one point to another
- Thus the movement of data is need to done with parallelism.
- Thus seti@home, folding@home uses this technique to large computing platforms.

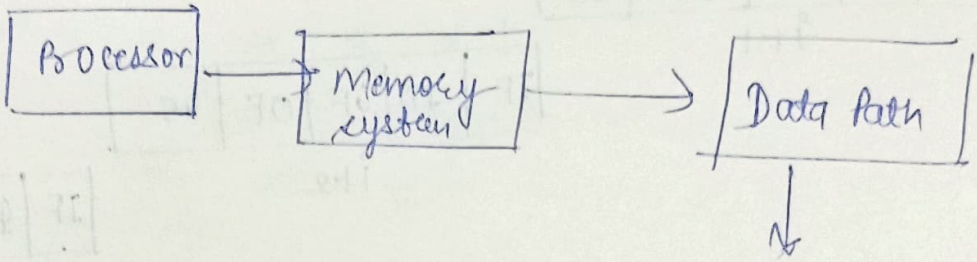
↓

Distributed computing project for disease research using molecular dynamics. through volunteer computing

Application

- Scientific
  - Genus & Proteins Characterization
  - Chemical pathways understanding
  - Astrophysics
  - weather modeling
  - Bioinformatics
- Commercial
  - Data mining
  - large scale servers
- Computer System Cyber
  - Intrusion Detect"
  - Automobiles

Conventional Architecture



H/W Responsible to perform ALU.

→ Pipelining ←

1. Connected in series.
2. Data elements are operated.
3. In OP/IP → O/P of Elem → I/P of elem.

Process in 5 phases

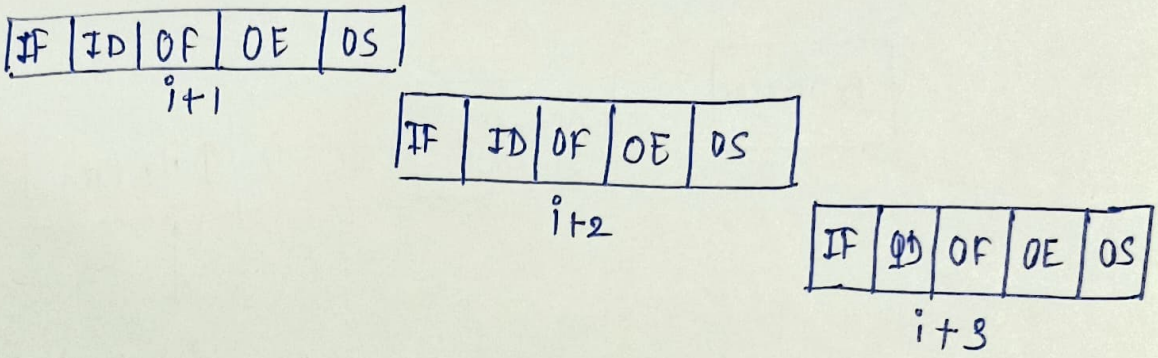
IF → Instruction fetch.
ID → Instruction decode
OF → Operand fetch
OE → Operand Execute.
OS → Operand Store.

IF	ID	OF	OE	OS.

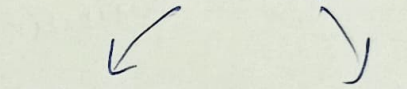
Reason to implement Pipeline.

1. To improve the performance of CPU improving faster circuits
2. Arrange the hardware in such a way that instruction executes at same time.

Non pipelining structure



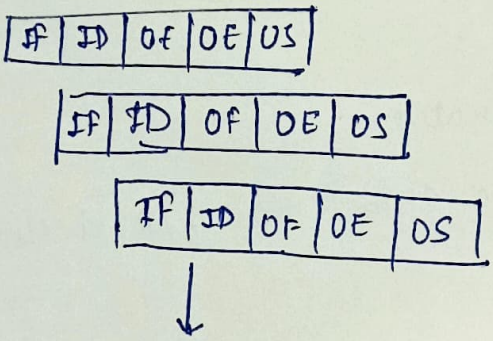
Pipelining



Arithmetic Pipelining → 4 stages

Instruction Pipelining → 5 stages / 6 stages

Pipelining structure



Interface ~~Intermediate~~ Registers are used to store the intermediate outputs -

Intermediate Register / Latch / Buffer.

① Instruction Fetch → CPU reads → memory → value in Program Counter

② Int. Decode : Register file is accessed. to get the values from Instruction

③ Ins. Execute : ALU operations are performed.

Pipelining $= (K \times n - 1)\ TP$.

where n = no. of inst.

K = no. of stages

TP : clock cycle.

Non. Pipelining : Non $n \times K \times TP$.

$$\text{Speed up} = \frac{\text{Perf. of NP}}{\text{Perf. of P.}} = \frac{n \times K \times TP}{(K + n - 1)\ TP}$$

$$= \frac{n \times k}{(k + n - 1)}$$

$\because n >>> k$:

$$\therefore \frac{n \times k}{n}$$

$$\boxed{S = k}$$

Super scalar Processor

(1) Uses Instruction level parallelism

↳ There is single thread of execution of a process with its own set of resources.

↳ Two approaches of Instruction level parallelism

↳ Hardware : Dynamic parallelism

↓

processor decide at run time that which instruction to execute in parallel

↳ Software : Static Parallelism

↳ Compiler decide which instruction to execute in parallel.

(2) To improve the rate of more than one instruction per clock cycle for a single sequential program on a single processor

| IF | ID | OF |

| IF | OID | OF |

| FF | ID | OF | IE | WB | NA |

Instruction level Parallelism

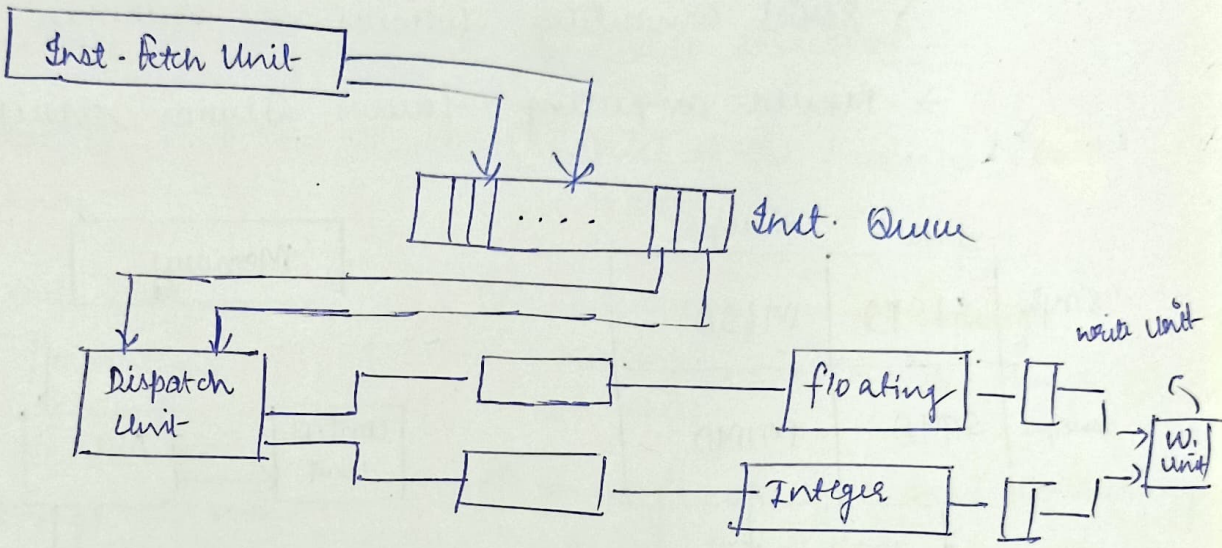
↙ Sequential

↓ Dependency

↘ Independence

:→ Superscalar is works in single processor.

:→ Executes more than one instruction during clock cycle by dispatching multiple instruction at same time.



Adv. : Can make the optimize use of resources.

Dis : Detrimental Effect of on performance
↳ harmful effect

Superscalar Archi., CPU manages several instructions pipeline to perform multiple Inst. at same time

Types of super scalar

↳ Intel core i7
↳ Intel Pentium
↳ IBM Power PC 601

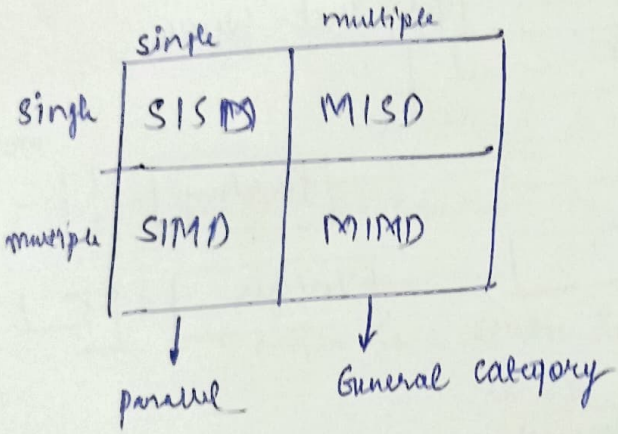
→ Instruction stream ←

Control Structure of Parallel Platforms

Parallel computing provide more resources & memory in order to tackle the problems that could not be solved through serial computing.

→ Serial computing follows von Neumann Architecture

→ Parallel computing follows Flynn's classification



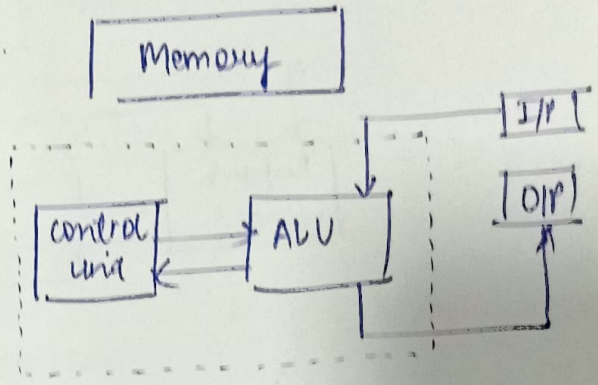


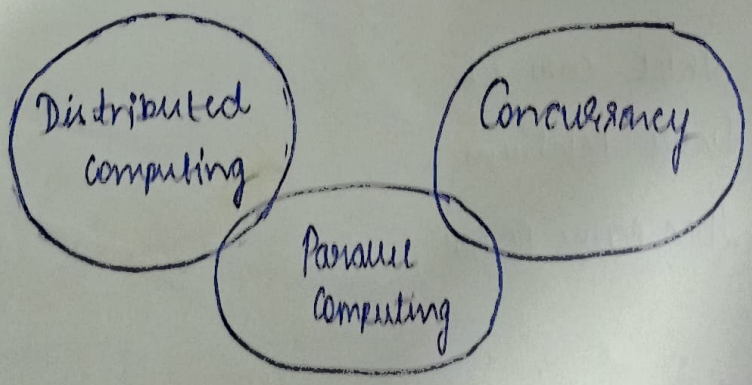
Fig (a) Neuman

SISD : Traditional computers.

SIMD : Image Processing & Signal Processing

MISD : Not Applicable.

MIMD : Distributed Computing & Parallel processing & high performance computing



Parallelism

1. Implementation Property
2. Task run / execute simultaneously.
3. Multi processor
4. Sub class of concurrency
5. Concurrent task can be parallelized

" Parallel is doing lot of things at once

Concurrency

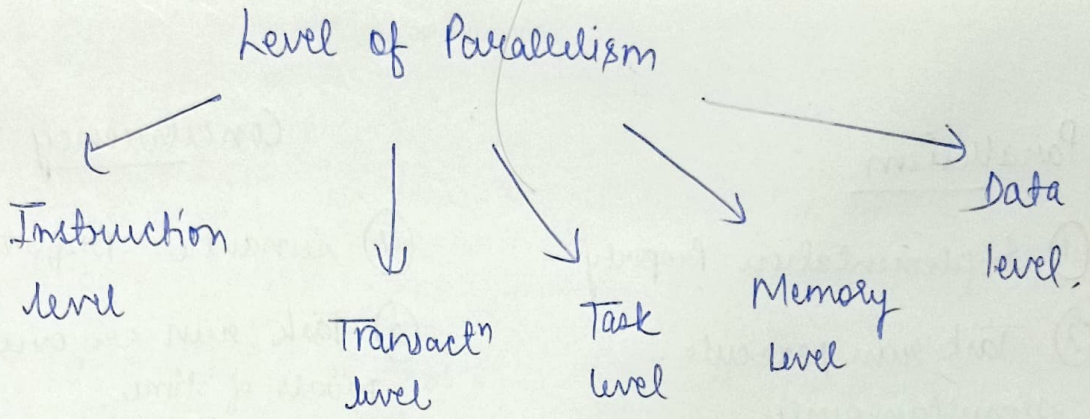
1. Semantic Property
2. Task run on overlapping periods of time
3. Single processor
4. Order of the processes doesn't matter.
6. Concurrency is ~~managing~~ dealing with lots of things at once.
7. In single core you have concurrency but not parallelism.

Superscalar

1. Use instruction level parallelism
2. several ins. simultaneously / multiple pipeline.
3. Spatial Parallelism → Duplicate Resource execute multiple ins.
4. Run con. on sever resources
5. Duplicating Hardware Resources. like

Pipelining

1. Overlapped within the execution
2. Single pipeline
3. Temporal Parallelism
4. Several operations on common h/w.
5. Executing units pipelined more deeply with very fast CLK


Level of Parallelism
Instruction level
Transactn level
Task level
Memory Level
Data level.

→ Very Long Instruction Word (VLIW) Archi ←

→ It optimizes the ILP. for improved performance.
→ It is simple way to build a superscalar microp.
→ CPU processor allows program to execute in sequence but VLIW processor allows to explicit specify instruction to ~~use~~ execute in parallel.
→ It gives high performance without complexity.
→ It is much more advanced to superscalar but the functioning is just like the superscalar.

Examp:

Intel i860, NXP (previous philips semiconductors) C6000 digital signal processor.

→ The architecture is depending upon the compiler.
→ The program decides the parallel flow of the instructers.
→ This increases the compiler complexity but decreases the hardware complexity.

Word = Register size = Sequence of bits that can be transformed from the working memory to the register of proces.

1 Word = 16 bits / 2 bytes.

→ Multiple Independent Operations are grouped together in single VLIW instruction

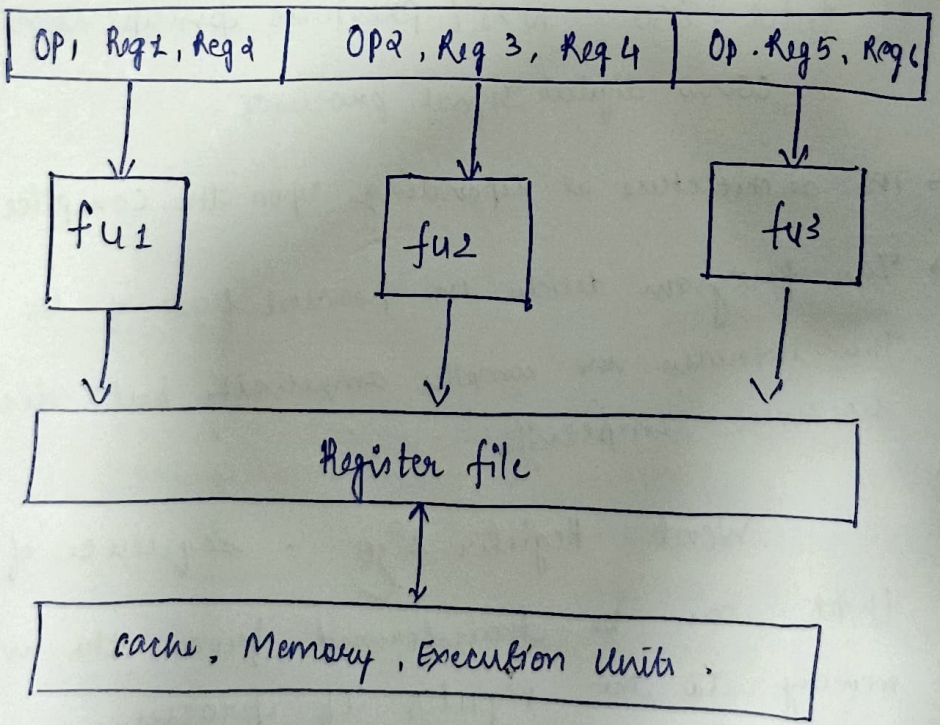
→ Each operation has independent functional unit.

→ functional unit shares common register file.

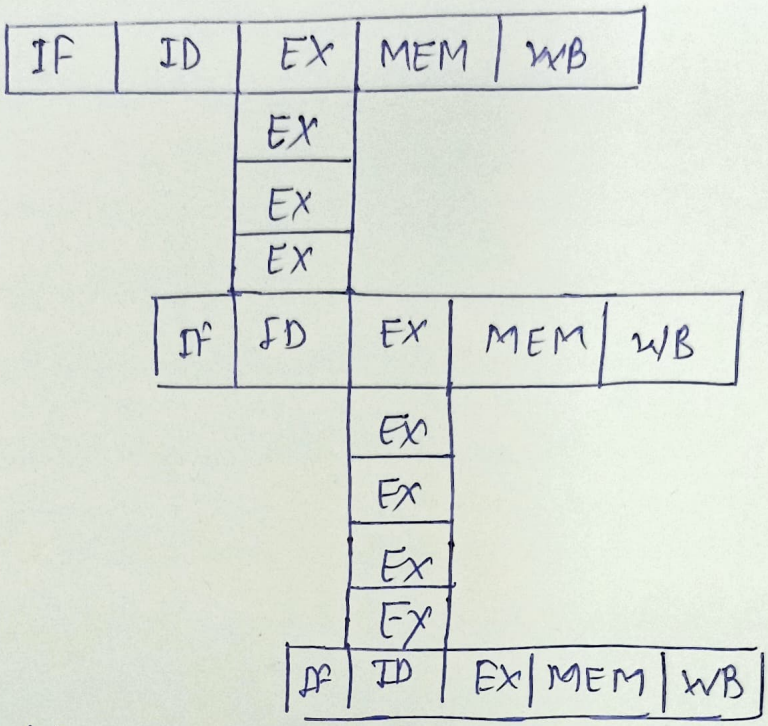
→ Ins. word. 64 – 1024 bits : depend upon no. of execution unit & code ~~unit~~ length.

→ Instruction scheduling & dispatching of word is done statistically by the compiler.

→ compiler checks for dependencies before scheduling parallel execution of the instructions.



Annexure No :



→ Advantages ←

→ Reduce H/w complexity

→ Reduce Power ~~loss~~ consumption

→ Simplifies decoding & instruction issues.

Disadvantage :

① Complex compilers Required

② Increased Program code size.

Applications

① Digital signal Processing

② Multimedia Processing

③ Scientific computing

④ Embedded system.