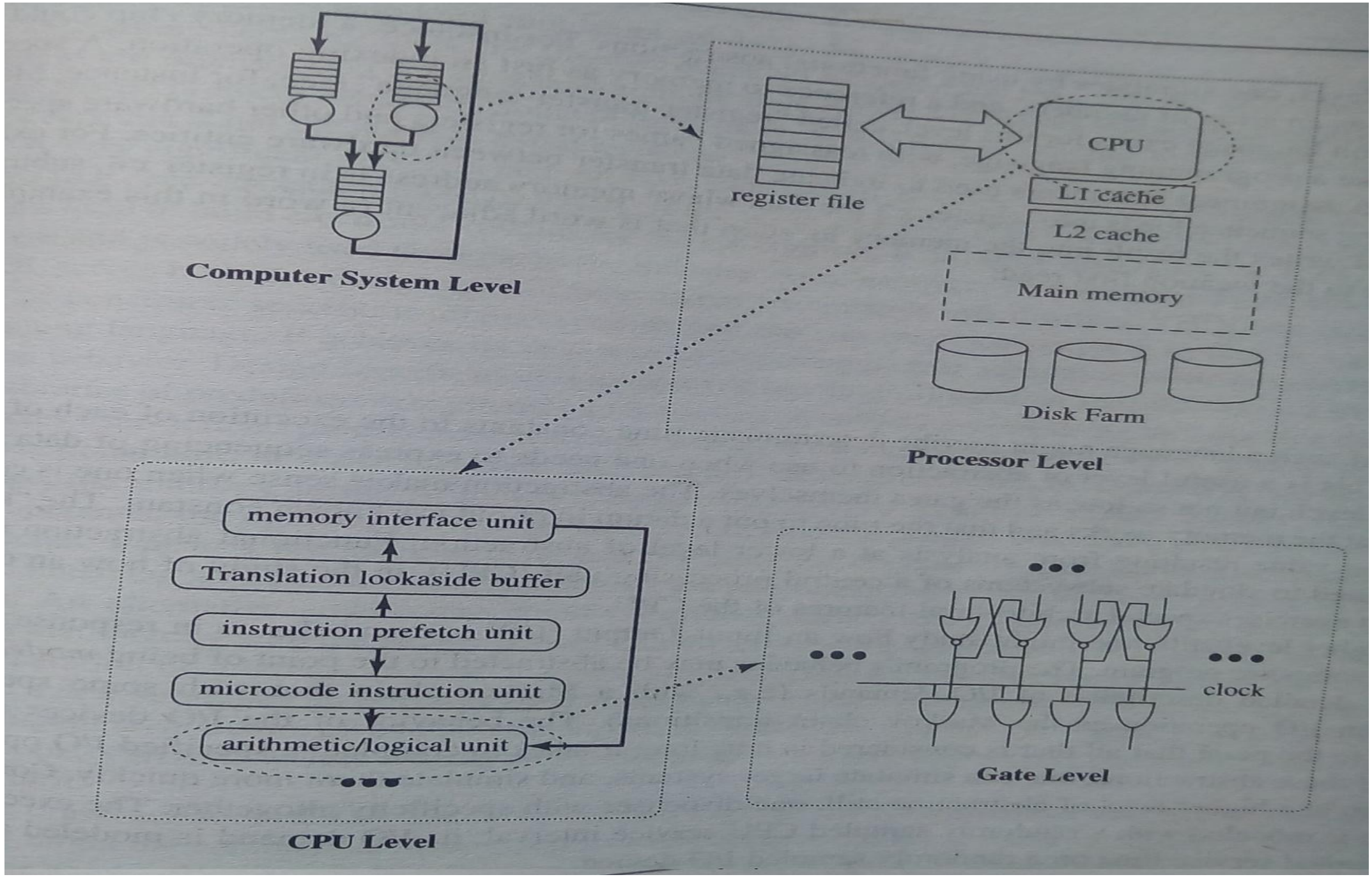
Chapter 10
Simulation of Computer System

Level of Abstraction in Computer System

- Level of Abstraction is defined as the amount of complexity by which a system is viewed or programmed.
- Computer system have complex time scale behaviour from **time to flipping transistor's state** to **time for human interaction**.
- It is designed hierarchically.
- The high level of abstraction is system level. In this level, one can view computational activity in terms of tasks circulating among servers, queuing for service when a server is busy.
- Below the system level is Processor level in which one can view components of the processor used.

- Below the Processor level is the CPU level in which one can view the activity of functional units that together make up a central processing unit.
- The lowest level is Gate level in which one can view the logical circuitry that is responsible for all the computations carried out by the computer system.
- Simulation is used in each level and the results of one level is used by another level.


Computer System Level
register file
CPU
L1 cache
L2 cache
Main memory
Disk Farm
Processor Level
memory interface unit
Translation lookaside buffer
instruction prefetch unit
microcode instruction unit
arithmetic/logical unit
CPU Level
clock
Gate Level

Simulation Tools

- Simulation tools are the tools that are used to perform and evaluate simulations at different abstraction levels of computer system.
- There are a number of powerful simulation tools available, all of them have advantages and disadvantages.
- An important characteristic of a tool is how it supports model building.
- The tools commonly used for simulation are:

1. CPU network simulation (Queueing network, Petri net simulators)
2. Processor simulation (VHDL(Very High Scale Integrated Circuit Hardware Description Language))
3. Memory simulation (VHDL)
4. ALU simulation (VHDL)
5. Logic network simulation (VHDL)
6. System Architecture Simulator(CSIM)

Activity, Process and Event Oriented Simulation

Activity Oriented Simulation

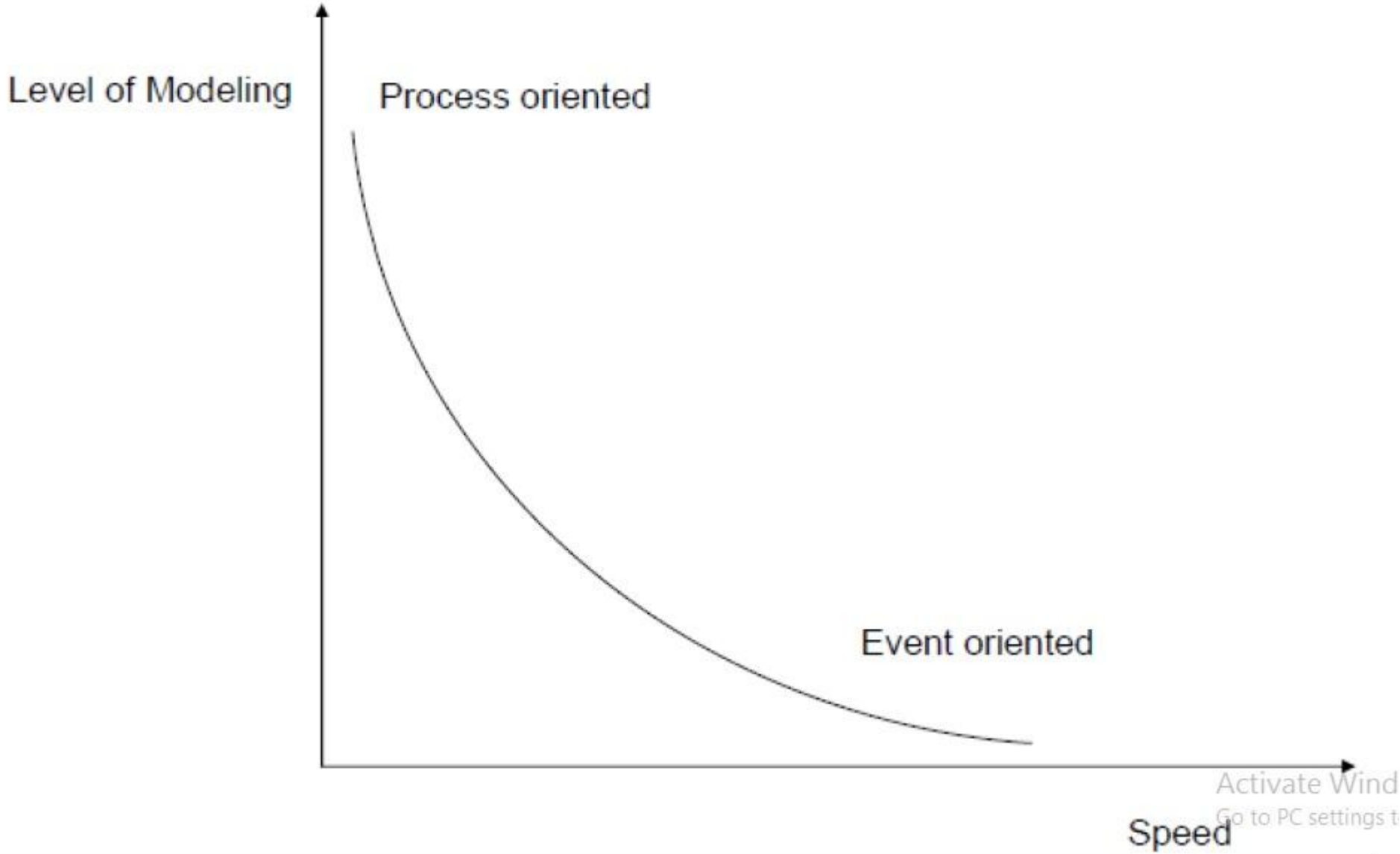
- The programmer defines the activities that are satisfied when certain conditions are satisfied.
- In many cases, this type of simulation uses a simulated clock which advance in constant increments of time.
- With each advance, list of activities is scanned and those which have become eligible are started.
- This type of model is used more often with simulating physical device.

Process Oriented Simulation

- The programmer defines the processes and the model in terms of interacting processes.
- A process is an independent program or procedure which can execute in parallel with other processes**.**
- The process will use the resource of the system.
- It implies that the tool must support separately schedulable threads of control.
- It allows continuous description with suspensions.

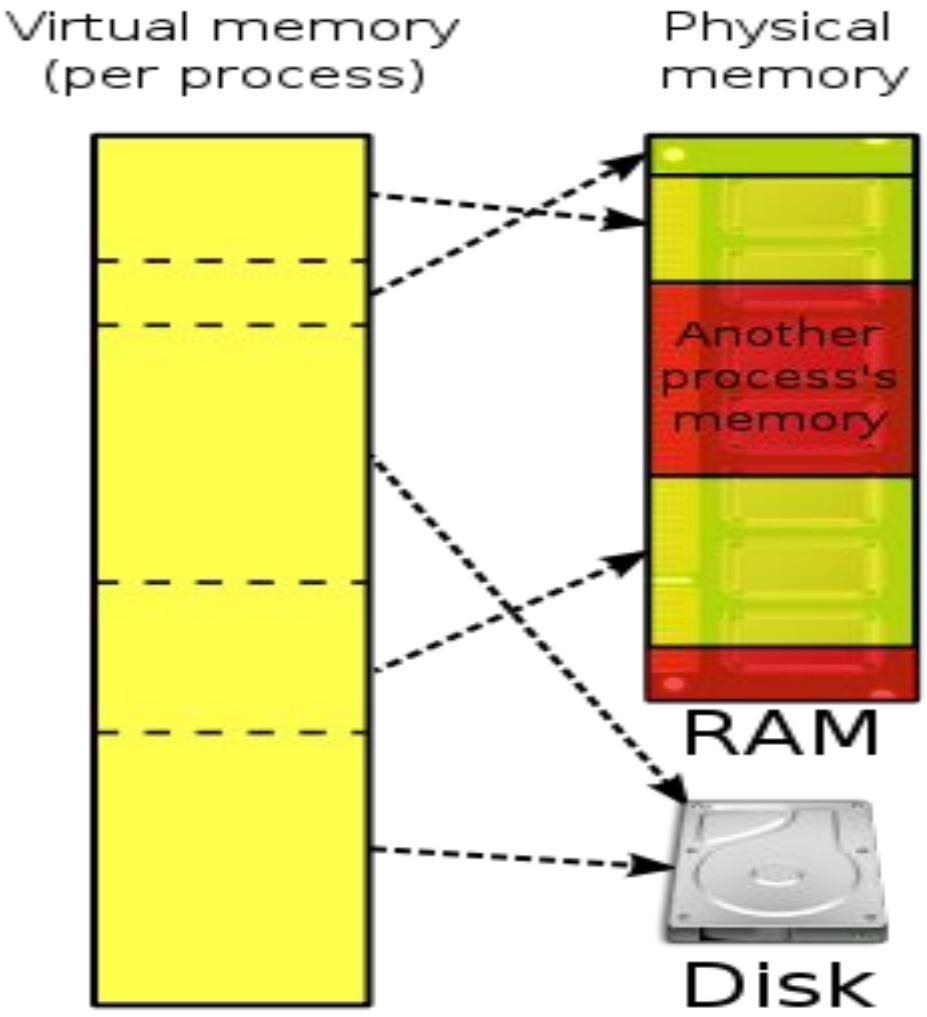
Event Oriented Simulation

- The simulation programmer defines events and then writes routines which are invoked as each kinds of events occur.
- It implies that the tool must support model description.
- It does not allows continuous description with suspensions.
- Usually a priority queue is used.



Virtual Memory Referencing

- Program is organized on units called pages.
- Physical memory is divided into page frames.
- Mapping is done by OS
- Replacement policy are used
- We can use computer simulation to find hit ratio(**ratio** of number of **hits** is divided by the total CPU reference of memory)

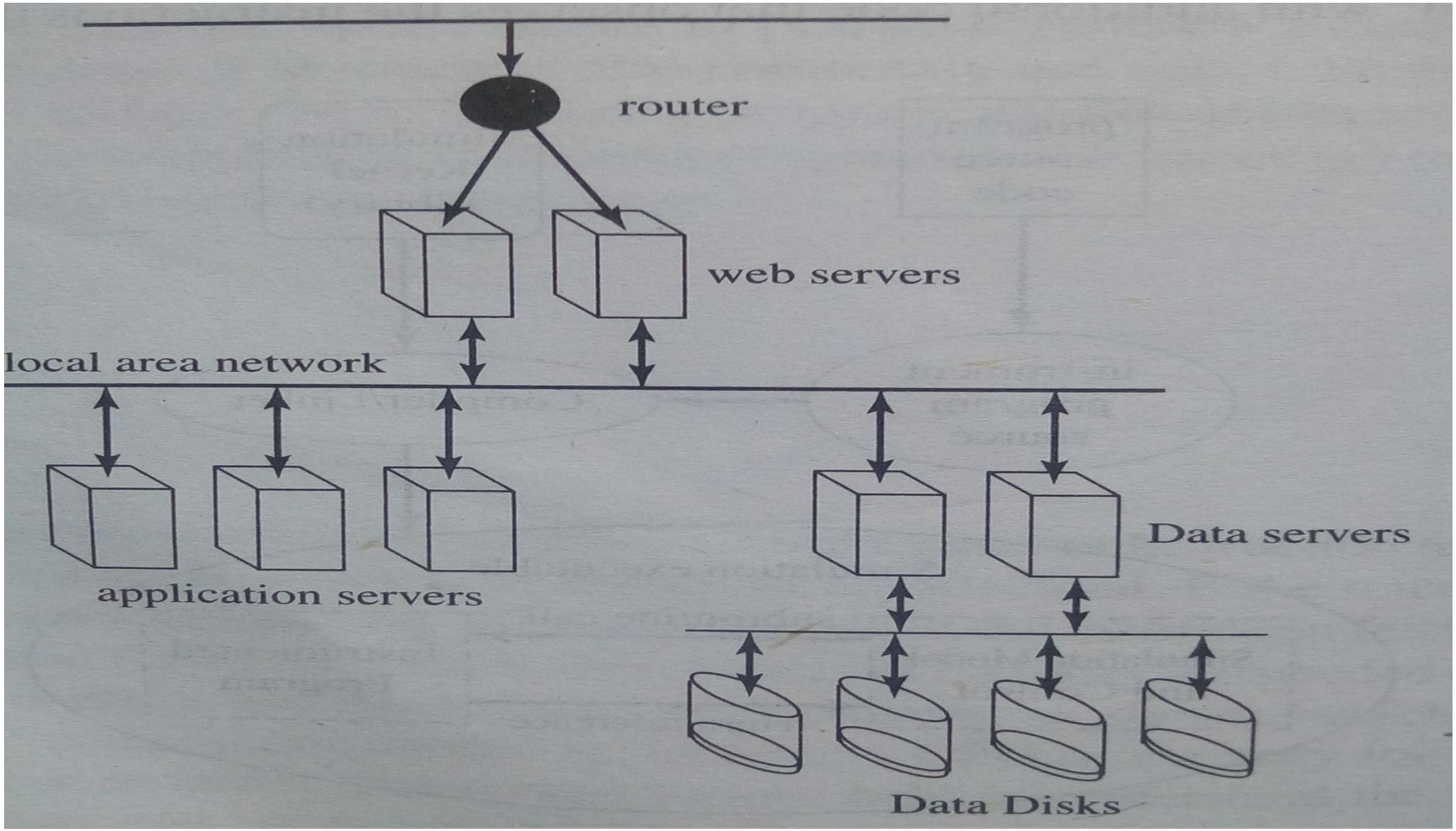


High Level Computer Simulation-System Simulation

Problem Definition

Consider a company provides a website for searching and links to sites for certain facilities. At the back end, there is **data servers** that handles specific queries and updates databases. Data servers receive requests for service from application servers. At front end, there is **web servers** that manage interaction of applications with the WWW. The whole system is connected with the users through the router. Let us consider that we need to study site's ability to handle load at peak periods i.e desired output is empirical distribution of the access response time.

Now, for this we need to focus on impact of timing at each level, factors that affect timing and effects of timing on contention for resources for designing this high level simulation model.


router
web servers
local area network
application servers
Data servers
Data Disks

Simulation Model

- All entries into the system are through dedicated router. It examines the request and forwards it to some web server.
- It takes some time to decide whether the request is a **new request or part of ongoing session.**
- One switching time is assumed for a pre-existing request and different time for a new request.
- It outputs the web server selection and enqueues request for service to the web server.
- Web server consists of one queue for **new requests**, one for **suspended requests that are waiting for response from application server** and **one for requests that are ready to process response from application server**.
- It is assumed that web server has enough memory to handle all the requests. It also has queuing policy.
- Associated application server is identified for each new requests.
- A request for service is formatted and forwarded to the application server and the request joins the suspended queue.

- Application server organizes the request for services. The new request for service joins the new-request queue.
- An application request is modeled as a sequence of sets of requests (organized in a burst) from data servers.
- For each application, a list of ready to execute and a list of suspended threads are maintained.
- Data servers create a new thread to respond to data request and places it in a queue of ready threads.
- When service is received, the thread requests data from a disk and then places in a suspended queue.
- Disk completes its operation for data request and the thread in data server, on receiving response from disk moves to ready list and reports back to application server associated with the request.
- The thread suspended at application server responds and finishes; then reports its completion to the web server.
- The thread in web server that initiates that request then communicates the results back to the Internet.

Note:

- Router have table of sessions
- Web server has three queues of threads
- Application server has two queues of threads
- Goal is to find response time distribution
- First we find bottleneck and then look how to reduce load at bottleneck during change of scheduling policy, biding applications to servers, increasing CPU and I/O devices.

Response Time

- Query-response-time distribution is estimated by measuring between the time at which a **request first hits the router** and the time at which **web server thread communicates the result.**
- The system can be analyzed by measuring behaviour at each server of each type.
- To assess system capacity at peak loads, we would simulate to identify bottlenecks, then look to see how to reduce load at bottleneck devices by changing various settings of simulation like scheduling policy, queue discipline and so on.

CPU Simulation

- In CPU simulation, we focus on discovering execution time and bottleneck situations that may appear.
- A bottleneck occurs when the **capacity of an application or a computer system is severely limited by a single component**, like the neck of a bottle slowing down the overall water flow.
- For CPU simulation, the input is the **stream of instructions** and the simulation must model the **logical design on what happens in response to the instruction stream.**
- Main challenges in CPU Simulation is to avoid stalling(Main challenges is to avoid stalling).

Problem Definition of ILP (Instruction Level Parallelism) CPU

 Pipelining has long been recognized as way of accelerating the execution of computer instructions.

The stages in an ILP CPU are as follows:

1. Instruction fetch - The instruction is fetched from memory.
2. Instruction decode - The memory word holding the instruction is interpreted to discover operations to be performed and registers involved.
3. Instruction Issue - An instruction is issued if no constraints hold it back from being executed.
4. Instruction Execute - The instruction operation is performed.
5. Instruction Complete - The results of instruction are stored in the destination register.
6. Instruction Graduate - Executed instructions are graduated in the order that they appear in the instruction stream.

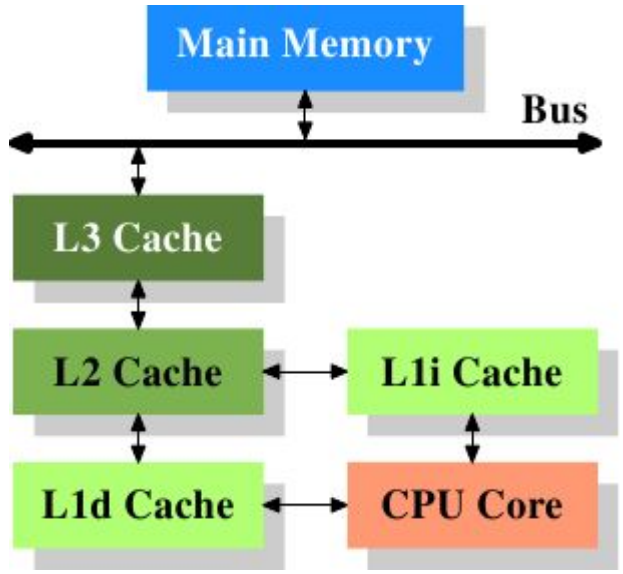
Simulation Model of ILP (Instruction Level Parallelism) CPU

- Instruction fetch interacts with the simulated memory system if present. If memory system is present, it can look into an instruction cache for the next referenced instruction, stalling if a miss is suffered. This stage makes instruction in the CPU's list of active instructions.
- Instruction Decode stage places an instruction in the list. A logical register that appears as the target of an operation is assigned a physical register. Registers used as operand are assigned physical registers that define their values. Branch instructions are identified and outcomes are predicted. Resources for the instruction execution are committed.
- Instruction Issue stage issue an decoded instruction for execution if values in its input registers are available and a functional unit needed to perform the instruction is available. It can be achieved by marking the registers and functional units as busy or pending. After the state is changed, the instruction waiting for that register or functional unit is reconsidered for issue.

- Instruction execute stage computes the result specified by the instruction. It means the actual operation intended by the instruction is performed.
- Instruction complete stage deposits the result into a register or memory as specified in the instruction.
- Instruction graduate reords the completed instruction in the same order as instruction stream. This is simulated by knowing the sequence number of the next instruction to be graduated.

Memory Simulation

- One of the great challenges of computer architecture is finding way to deal effectively with the increasing gap in operation speed between CPU and memory.
- Memory is arranged hierarchically with L1 cache, L2 cache, main memory and disks.
- Example: Cache Simulation
  a. The input is cache parameters and memory access tree.
  b. The output of simulation is cache hit rate or hit ratio.
- The **Cache Hit Ratio** is the **ratio** of the number of **cache hits** to the number of lookups(hit + miss), usually expressed as a percentage.



- Replacement Policy: Policy that determines which block in cache is removed in order to create space for coming block.
- Blocks can be removed in random fashion, using FIFO, LIFO,LFU(Least Frequently Used), LRU(Least Recently Used) strategies.
- LRU(Least Recently Used) is the widely used cache replacement strategy.

Simulation Model

- Maintain cache directory and LRU status of the lines within the set.
- When an access is made, update LRU status.
- If a hit, record it as such.
- If a miss, update the contents of the directory.
- Cache directory is implemented as an array, with array entries corresponding to directory entries.