UNIT IV

1. **What is meant by hardware and software clock?**
   Clock devices can be programmed to generate interrupts at regular intervals in orders that, for example, time slicing can be implemented. The operating system reads the node’s hardware clock value, \( H(t) \), scales it and adds an offset so as to produce software clock \( C(t) = \alpha H(t) + \beta \) that approximately measures real, physical time \( t \) for process \( p_i \).

2. **What is clock resolution?**
   Note that successive events will correspond to different timestamps only if the clock resolution—the period between updates of the clock-value—is smaller than the time interval between successive events. The rate at which events occur depends on such factors as the length of the processor instruction cycle.

3. **What is clock drift?**
   Clock drift, which means that they count time at different rates and so diverge. The underlying oscillators are subject to physical variations, with the consequence that their frequencies of oscillation differ. Moreover, even the same clock’s frequency varies with temperature. Designs exist that attempt to compensate for this variation, but they cannot eliminate it. A clock’s drift rate is the change in the offset (difference in reading) between the clock and a nominal perfect reference clock per unit of time measured by the reference clock.

4. **What is IAT?**
   Computer clocks can be synchronized to external sources of highly accurate time. The most accurate physical clocks use atomic oscillators, whose drift rate is about one part in 10^13. The output of these atomic clocks is used as the standard for elapsed real time, known as International Atomic Time.

5. **What is CUT?**
   Coordinated Universal Time—abbreviated as UTC (From the French equivalent)—is as international standard for timekeeping. It is based on atomic time, but a so-called ‘leap second’ is inserted—or, more rarely, deleted—occasionally to keep it in step with astronomical time.
6. **What is meant by external synchronization?**

In order to know at what time of day events occur at the processes in our distributed system—it is necessary to synchronize the processes’ clocks, \( C \), with an authoritative, external source of time. This is external synchronization.

We define these two modes of synchronization more closely as follows, over an interval of real time \( I \):

- For a synchronization bound \( D > 0 \), \(|C_i(t) - C_j(t)| < D\) for \( i = 0, 1, 2, \ldots N \) and for a source \( S \) of UTC time, \(|S(t) - C_i(t)| < D\), for \( i = 1, 2, \ldots \) and for all real times \( t \) in \( I \).

- The clocks \( C_i \) are accurate to within the bound \( D \).

7. **What is internal synchronization?**

And if the clocks \( C \) are synchronized with one another to known degree of accuracy, then we can measure the interval between two events occurring at different computers by appealing to their local clocks, even though they are not necessarily synchronized to an external source of time. This is internal synchronization.

- For a synchronization bound \( D > 0 \) and for a source \( S \) of UTC times, \(|S(t) - C_i(t)| < D\), for all real times \( t \) in \( I \).

- Clocks \( C \) agree with in the bound \( D \).

8. **Define NTP and its design aims.**

Cristian’s method and the Berkeley algorithm are intended primarily for use within intranets. The Network Time Protocol (NTP) [Mills1995] defines an architecture for a time service and a protocol to distribute time information over the Internet.

NTP’s chief design aims and features are as follows:

- To provide a service enabling clients across the Internet to be synchronized accurately to UTC:

- To provide a reliable service that can survive lengthy losses of connectivity:

- To enable clients to resynchronize sufficiently frequently to offset the rates of drift found in most computers: To provide protection against interference with the time service, whatever malicious or accidental.
9. **What is strata?**

The NTP service is provided by a network of servers located across the Internet. Primary servers are connected directly to a time source such as a radio clock receiving UTC; secondary servers are synchronized, ultimately, with primary servers. The servers are connected in a logical hierarchy called a synchronization subnet whose levels are called strata.

10. **Enumerate the mode of synchronization in NTP servers.**

- NTP servers synchronize with one another in one of three: multicast, procedure-call and symmetric mode.
- *Multicast mode* is intended for use on a high-speed LAN. One or more servers periodically multicasts the time to the servers running in other computers connected by the LAN, which set their clocks assuming a small delay. This mode can achieve only relatively low accuracies, but ones that nonetheless are considered sufficient for many purposes.
- *Procedure-call mode* is similar to the operation of Cristian’s algorithm. In this mode, one server accepts requests from other computers, which it processes by replying with its timestamp (current clock reading). This mode is suitable when higher accuracies are required than can be achieved with multicast, or where multicast is not supported in hardware.
- *In symmetric mode* is intended for use by the servers that supply time information in LANs and by the higher levels of the synchronization subnet, where the highest accuracies are to be achieved.

11. **What is filter dispersion?**

NTP servers apply a data filtering algorithm to successive pairs which estimates the offset $o$ and calculates the quality of this estimates as a statistical quantity called the filter dispersion.

12. **What is synchronization dispersion?**

Peers with lower stratum numbers are more favoured than those in higher strata because they are ‘closer’ to the primary time sources. Also, those with the lowest synchronization dispersion are relatively favoured. This is the sum of the filter dispersions measured between the server and the root of the synchronization subnet.
13. What is meant by HB relation?
   - Lamport called the partial ordering obtained by generalizing these two relationships the happened-before relation. It is also sometimes known as the relation of causal ordering or potential causal ordering.
   - We can define the happened-before relation, denoted \( \rightarrow \) by as follow:
     
     HB1: If \( p_i : e \rightarrow e' \) , then \( e \rightarrow e' \)
     
     HB2: For any message \( m \), send (m) receive (m)
     
     HB3: IF \( e, e' \) and \( e'' \) are events such that \( e \rightarrow e' \) then \( e \rightarrow e'' \)

14. What is logical clock?
   - Lamport [1978] invented a simple mechanism by which the happened before ordering can be captured numerically, called a logical clock.
   - A Lamport logical clock is a monotonically increasing software counter, whose value need bear no particular relationship to any physical clock.
   - Each process \( p \) keeps its own logical clock, \( L \), which it uses to apply so-called Lamport timestamps to a events.
   - We denote the timestamp of event \( e \) at \( p_i \) by \( L(e) \) , and by \( L(e) \) we denote the timestamp of event \( e \) at whatever process it occurred at.

15. Define Vector clock

Vector clocks for a system of \( N \) processes is an array of \( N \) integers

- Shortcoming of Lamport clocks:
  
  \[ L(e) < L(e') \] doesn't imply \( e \rightarrow e' \)

- Vector clock: an array of \( N \) integers for a system of \( N \) processes
  
  - Each process keeps its own vector clock \( Vi \) to timestamp local events
  
  - Piggyback vector timestamps on messages

- Rules for updating vector clocks:
  
  - \( Vi[i] \) is the number of events that \( p_i \) has timestamped
  
  - \( Vi[j] \) ( \( j \neq i \)) is the number of events at \( p_j \) that \( p_i \) has been affected by

  VC1: Initially, \( Vi[j] := 0 \) for \( p_i, j=1..N \) (\( N \) processes)
  
  VC2: before \( p_i \) timestamps an event, \( Vi[i] := Vi[i]+1 \)
  
  VC3: \( p_i \) piggybacks \( i = Vi \) on every message it sends
VC4: when $pi$ receives a timestamp $t$, it sets $V_i[j] := \max(V_i[j], t[j])$ for $j=1..N$ (merge operation)

16. What do you meant by distributed garbage
An object is considered to be garbage if there are no longer any reference to it anywhere in the distributed system. The memory taken up by that object can be reclaimed once it is known as to be garbage.

17. Define Global History
Let us return to our general system $p$ of $N$ processes $p_i (i=1,2,3,.....N)$
Here a series of events occurs at each process, and that we may characterize the execution of each process by its history.

18. What is meant by cut?
Consider the events occurring at processes $p_1$ and $p_2$ shown in figure

19. Define Global state predicate
- A Global State Predicate is a function that maps from the set of global process states to True or False.
- Detecting a condition like deadlock or termination requires evaluating a Global State Predicate.
- A Global State Predicate is stable: once a system enters a state where it is true, such as deadlock or termination, it remains true in all future states reachable from that state.
- However, when we monitor or debug an application, we are interested in non stable predicates.
20. **List the assumption considered in snapshot algorithm**
   - Neither channels nor processes fail
   - Reliable communications ensure every message sent is received exactly once
   - Channels are unidirectional
   - Messages are received in FIFO order
   - There is a path between any two processes
   - Any process may initiate a global snapshot at any time
   - Processes may continue to function normally during a snapshot

21. **Define Failure detector.**
   A failure detector is a service that processes queries about whether a particular process has failed. It is often implemented by an object local to each process that runs failure detection algorithms in conjunction with its counterparts at the other processes.

22. **List the properties of failure detector**
   A failure detector is not necessarily accurate. Most falls into the category of **unreliable failure detectors.**
   - A result of unsuspected
   - A result of Suspected

23. **Define critical section problem**
   The application – level protocol for executing a critical section is as follows
   - `enter()` - enter critical section – block if necessary
   - `resourceAccesses()` - access shared resources in critical section
   - `exit()` - leave critical section other processes may now enter.

24. **What is meant by election**
   Election: choosing a unique process for a particular role is called an election
   - All the processes agree on the unique choice
   - For example, server in dist. mutex

25. **List the famous mutual exclusion algorithms**
   - Center server algorithm
   - Ring- Based algorithms
   - Mutual Exclusion using multicast and Logical Clocks
26. What do you meant by bully algorithms and types of messages

Assumption: Each process knows which processes have higher identifiers, and that it can communicate with all such processes

• Compare with ring-based election
  – Processes can crash and be detected by timeouts

• synchronous

• timeout \( T = 2T_{transmitting} \) (max transmission delay) + \( T_{processing} \) (max processing delay)

Three types of messages

a. Election: announce an election
b. Answer: in response to Election
c. Coordinator: announce the identity of the elected process

27. What are the types of ordering in multicast

Three types of message ordering

• **FIFO (First-in, first-out) ordering:** if a correct process delivers a message before another, every correct process will deliver the first message before the other

• **Casual ordering:** any correct process that delivers the second message will deliver the previous message first

• **Total ordering:** if a correct process delivers a message before another, any other correct process that delivers the second message will deliver the first message first

28. Define Consensus

Consensus more precisely and relates it to three related Problems of agreement. For processes to agree on a value (consensus) after one or more of the processes has proposed what that value should be. Covered topics: **byzantine generals, interactive consistency, totally ordered multicast**

• The byzantine generals problem: a decision whether multiple armies should attack or retreat, assuming that united action will be more successful than some attacking and some retreating

• Another example might be space ship controllers deciding whether to proceed or abort.

Failure handling during consensus is a key concern
29. What are the requirements of interactive consistency?

Three requirements of a consensus algorithm

- **Termination**: Eventually every correct process sets its decision variable
- **Agreement**: The decision value of all correct processes is the same: if pi and pj are correct and have entered the \textit{decided} state, then \(d_i=d_j\) for all \((i,j=1,2,\ldots,N)\)
- **Integrity**: If the correct processes all proposed the same value, then any correct process in the \textit{decided} state has chosen that value

30. What are the requirements of consensus algorithms

The requirements of a consensus algorithms are that are following conditions should hold for every execution of it:

- Termination
- Agreement
- Integrity

**PART B (16 MARK QUESTIONS)**

1. Explain clocks, events and process state in details
2. Discuss in details any two algorithms used for synchronizing clocks in intranet
3. Explain logical clocks in details
4. Discuss the snapshot algorithms
5. Explain distributed mutual exclusion algorithms in details
6. Explain group and multicast communication in details
7. Explain ordering multicast communication and its types
8. Explain consensus and its related problems in details
9. Explain ring based election algorithms
10. Explain bully’s algorithms
11. Explain global states and consistent cuts in details