1.1 Terminologies:

1.1.1. System Concepts

→ A system is a mapping of a set of inputs into a set of outputs.

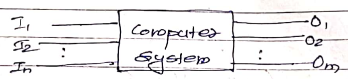

fig: A system with n-inputs & m-outputs.

Response Time:-

The time between the presentation of a set of inputs to a system and the realization of required behaviour, including the availability of all associated outputs is called response time of the system.

1.2. Real-Time Definitions:

→ A real-time system is defined as those systems in which correctness of system depends not only on the logical result of computation but also on time at which results are produced.

→ Depending upon temporal behaviour of Real Time System. They can be characterized as

- Hard RTS
- Soft RTS
- Firm RTS

a) Hard-Real Time System:-

→ A hard real-time system is the one in which failure to meet a single deadline may lead to complete and catastrophic system failure.

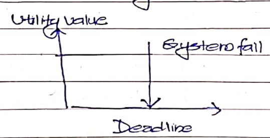

fig:- Hard RTS.

→ A system crashes if the deadline is not met by the tasks.

→ A hard deadline is imposed on such jobs in which late result produced by the job after the deadline have disastrous consequences.

Example:- Rocket launching system.

:- Air bag control system in car.

b) Soft RTS

→ A soft RTS is one in which performance is degraded but n destroyed by failure to meet response time constraints.

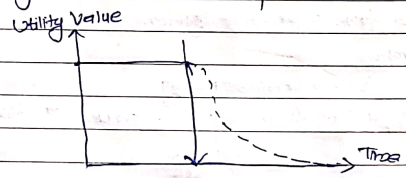

fig: Soft RTS

- Those real time system where missing a deadline doesn't create the system failure but degrades the system performance are called soft RTS
- Generally, system with no strict deadlines ~~are~~ can be thought of as soft RTS

  eg:- ExamHall (Delay by 10 or 20 minute is allowed)
  More delay results in less time to write.

c) FIRM - Real Time System.

- A firm real time system is one in which a few missed deadlines will not lead to total failure, but missing more than a few may lead to complete and catastropic system failure.

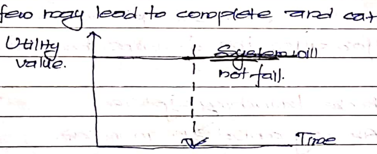

fig: Firm Real Time System

- Those system where, the missing of deadline doesn't cause the system to crash, but cause the result to be omitted are called firm real time system.

  Example:
  Video Conferencing
  (missing few frames won't cause a disaster)

Parts of Typical Real-Time System:-
→ Controlling System → air bag controlling sys
→ Controlled System → air bag
→ Operating Environment → car

Job: Job are unit of work that are scheduled and executed by the system.

Tasks: The set of related jobs that can be solved by the same algorithm is called a task

Deadline:- A deadline 'd' is a time at which a task or job must be complete.
* A hard deadline means that it is important for the safety that this deadline always be met.
* A soft deadline means that it is desirable to finish executing the task/job by the deadline, but that no catastrophe occurs if completion is late.
* A firm deadline means there is no value in completing the task after its deadline.

1.1.3 Types of Real-Time Tasks:-

① Periodic Tasks:
- Periodic task are activated regularly at fixed rate
- These tasks are time driven
- The time between two successive activation is called the period.

- consists of a sequence of identical jobs.
- Usually a periodic task can be characterized by its computation time 'C' and deadline 'D'.

Example:

Monitoring temperature of a patient.

Aperiodic Tasks:-

- Aperiodic tasks are activated irregularly at some unknown and possibly unbounded rate.
- Jobs have soft or no deadline
- These tasks are event driven
- Example:-

An event is activated when condition of patient changes.

Sporadic Tasks:-

- Sporadic tasks are activated irregularly with some known and bounded rate.
- Sporadic tasks may make a request at any time, but two successive requests must be separated in time by at least P(minimum separation) time units.
- Jobs have hard deadlines.
- Example:-

A patient can call doctor at anytime but the interval between two successive call must be at least 1 hour.

1.1.3 Events and Determinism

next inst ko address.

Events:-

Any occurance that causes the program counter to change non-sequentially is considered a change of flow-of-control and thus an event.

How change in flow control occurs:-

- A change in state results in a change in the flow-of-control
- The decision block suggest that the program flow can take alternative paths
- Case, If-then and While statements represents a possible change in flow-of-control.
- Invocation of procedures in Ada and C represents change in flow-of-control.
- In C++ and Java, instantiation of an object or invocation of a method causes change in flow-of-control.
- Events can also be periodic, aperiodic and sporadic.

Deterministic Systems:-

- A system is deterministic if for each possible state and each set of inputs ~~can~~ ~~data~~, a unique set of outputs and next state of the system can be determined.
- Event determination means the next state and output of a system are known for each set of inputs that trigger events.

h/w, s/w

- If in a deterministic system the response time for each set of output is known then the system also exhibits temporal determinism.
- A system is considered to be in control when the next state of the system, given the current state and a set of inputs is predicted.
- For any physical system, certain states exists under which the system is considered to be out of control.

1.15. CPU Utilization:-

Resource Use

- CPU utilization (or time loading factor) is a measure of the percentage of non-ideal processing.
- CPU utilization refers to a computers usage of a processing resources or the amount of work handled by a CPU.

- Suppose a system has $n \geq 1$ periodic tasks, each with an execution period of P_i and hence execution frequency $f_i = \frac{1}{P_i}$. If task 'i' is known to have a maximum execution time of e_i, then the utilization factor U_i for task i is;

$$U_i = \frac{e_i}{P_i}$$

Then overall system utilization is

$$U = \sum_{i=1}^{n} U_i = \sum_{i=1}^{n} \frac{e_i}{P_i}$$

1.2 Real-Time System Design Issues:-

1) The selection of hardware and software, and evaluation of the trade off needed for a cost-effective solution; including dealing with distributed computing systems and the issues of parallelism and synchronization.

→ selection of appropriate s/w language.

2) Understanding the detail of the programming languages and the real time implications resulting from their translation into machine code.

3) Specification and design of real-time systems and correct representation of temporal behaviour
time.

4) Maximizing of system fault tolerance and reliability through careful design.

5) Selection of tools and equipments for design and administration of tests design & administration garna kun tool & eqp use garne

6) Taking advantage of open systems technology (such as Linux) and interoperability (such as CORBA)

7) Estimating and measuring response time and reducing them (if needed) by performing a schedulability analysis

8) Selection of real time operating system.

1.3 Examples of Real Time System

Domain	Applications
1) Multimedia	Games & Simulators
2) Medicine	Remote / Robot surgery & Medical Imaging
Industrial Systems	Robotic assembly lines & automated inspection
Civilian	Elevator control & Automotive Systems
Avionics (related to aircraft)	Navigation & Displays.

- Aircraft inertia measurement system
- Nuclear Plant Control
- Airline Reservation System
- Traffic light control system for 4-way intersection

Common Misconceptions regarding Real Time Systems

① Real time System are synonymous with 'fast' system
⇒ Real Time system is not concerned with fast but is concerned with meeting the deadline

② There are universal, widely accepted methodology for real time system specification & design.
⇒ There is still no methodology available that answers all of the challenges of real-time specification and design all the time for all application.

③ There is no more a need to build a real time O.S because many commercial product exist.
⇒ As there are number of popular and cost-effective commercial RTOS, but also choosing the right one at the right time is very challenging.

④ Rate Monotonic Analysis has solved "the real time problem".
⇒ Although, rate monotonic system provide guidance in the design of real time system. However, it has not solved all the real time problems

⑤ The study of real-time systems is mostly about scheduling theory
⇒ It is required to study scheduling theory but the successful uses of scheduling theory in RTOS requires practical experience, simplification and power to see the future.

Diff RTOS & non RTOS.