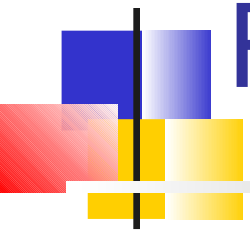
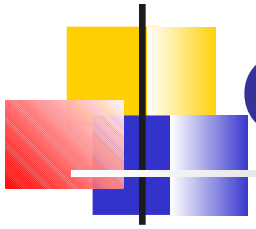


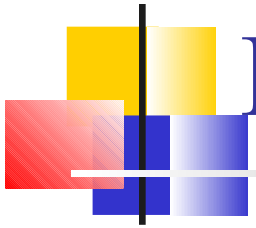
# Real-Time Task Scheduling

*Basic concepts..*



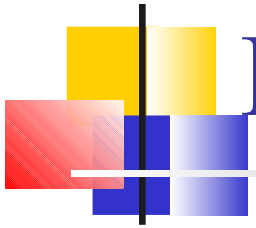
# Outline

- Introduction
- Important concepts
- Types of real-time task & their characteristics
- Task scheduling
  - Basic concepts
  - Classification of scheduling algorithms



# Introduction

- Real-time tasks get generated in response to some events that may either be external or internal to the system.
- Every real-time system usually consists of a number of real-time tasks.
- The time bounds on different tasks may be different.
- Selection of appropriate task scheduling algorithm is central to the proper functioning of a real time system(to meet the time constraint of the task).

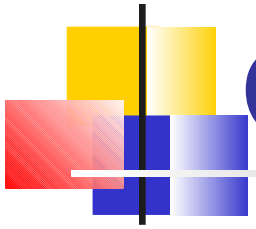


# Important concepts

- Task instance
- Relative deadline Vs Absolute deadline
- Response time
- Task precedence
- Data sharing

# Task instance

- A task is generated when some specific event occurs.
- Real time tasks normally recur a large number of times at different instants of time depending on event occurrence time.
- Most real-time tasks recurs with certain fixed periods.
  - E.g. : A temperature sensing task in chemical plant might recur indefinitely with a certain period because the temperature is sampled periodically, whereas a task handling a device interrupt might recur at random instants.



# Cont..

- Each time a task recurs, it is called an instance of the task. The first time a task occurs, it is called the first instance of the task.
- The next occurrence of the task is called its second instance, and so on.
- The jth instance of a task Ti would be denoted as Ti(j).
- Each instance of a real-time task is associated with a deadline by which it needs to complete and produce results.

# Relative Deadline Vs Absolute Deadline

**Absolute Deadline**

- The absolute deadline of a task is the ab-solute time value (counted from time 0) by which the results from the task are expected.
- Thus , absolute deadline is equal to the interval of time between the time 0 and the actual instant at which the deadline occurs as measured by some physical clock.

**Relative Deadline**

- Relative deadline is the time in-terval between the start of the task and the instant at which the deadline occurs.
- In other words, relative deadline is the time interval between the arrival of a task and the corresponding dead-line.
- the relative deadline of the task Ti(1) is d, whereas its absolute dead-line $\phi + d$.

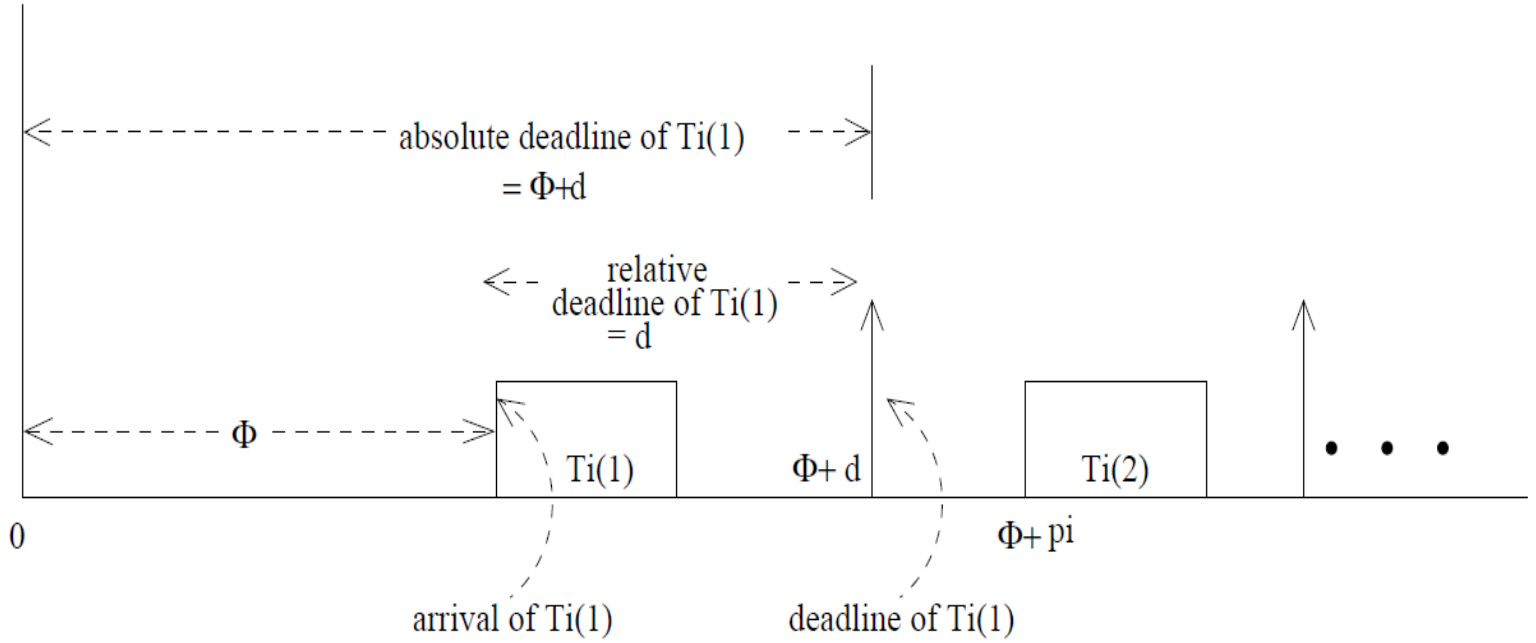
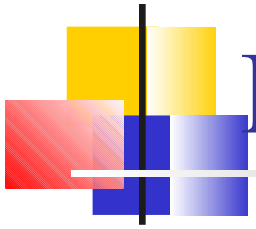



Figure 1: Relative and Absolute Deadlines of a Task



# Response Time

- The response time of a task is the time it takes (as measured from the task arrival time) for the task to produce its results.
- As already remarked, task instances get generated due to occurrence of events.
- These events may be internal to the system, such as clock interrupts, or external to the system such as a robot encountering an obstacle.

# Task Precedence

- A task is said to precede another task,if the first task must complete before the second task can start.
- When a task Ti precedes Tj , then each instance of Ti precedes corresponding instance of Tj .
- A precedence order defines a partial order among tasks.
- E.g. : partial ordering among tasks - T1 precedes T2, but we cannot relate T1 with either T3 or T4.

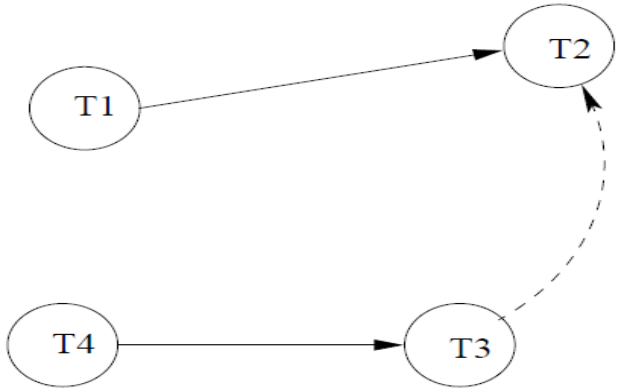
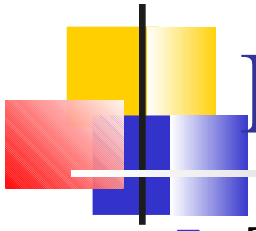
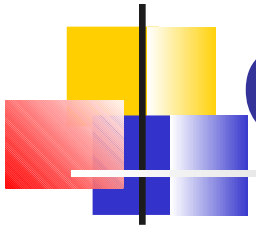



Figure 2: Precedence Relation Among Tasks



# Data Sharing

- Tasks often need to share their results among each other when one task needs to share the results produced by another task; clearly, the second task must precede the first task.
- In fact, precedence relation between two tasks sometimes implies data sharing between the two tasks (e.g., first task passing some results to the second task). However, this is not always true.
- A task may be required to precede another even when there is no data sharing.
- E.g. in a chemical plant it may be required that the reaction chamber must be filled with water before chemicals are introduced.



# Cont…

- In this case, the task handling filling up the reaction chamber with water must complete, before the task handling introduction of the chemicals is activated.
- It is, therefore, not appropriate to represent data sharing using precedence relation

# Cont…

- Data sharing among the tasks does not necessarily impose any particular ordering among the tasks.
- Here, T2 uses the results of T3.T2 may even start executing first , after sometimes it may receive some data from T3, and continues its execution.

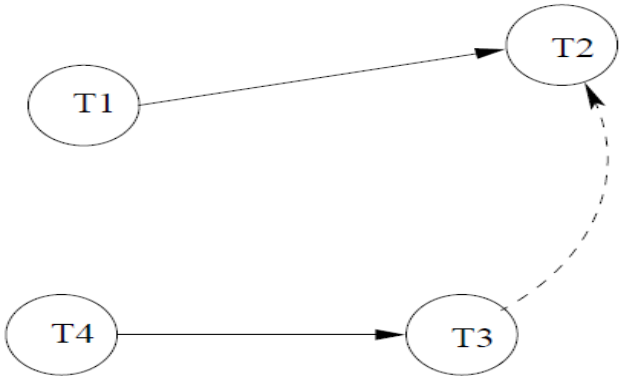



Figure 2: Precedence Relation Among Tasks

# TYPES OF REAL-TIME TASKS & THEIR CHARACTERISTICS

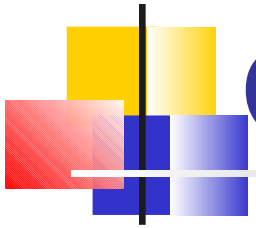
- Based on the way real-time tasks recur over a period of time, it is possible to classify them into three main categories:
  - Periodic tasks
  - Sporadic tasks
  - Aperiodic tasks.

# Periodic Tasks

- A periodic task is one that repeats after a certain fixed time interval.
- Periodic tasks are sometimes referred to as clock-driven tasks.
- The fixed time interval after which a task repeats is called the period of the task.
- If Ti is a periodic task, then the time from 0 till the occurrence of the first instance of Ti(i.e., Ti(1)) is denoted by $\phi_i$ ; and is called the *phase of the task.*
- Periodic task Ti can be represented using four tuples:

$$(\phi_i, p_i, e_i, d_i)$$

  - Pi -is the period of task,
  - ei - is the worst case execution time of the task,
  - di - is the relative deadline of the task.



# Cont..

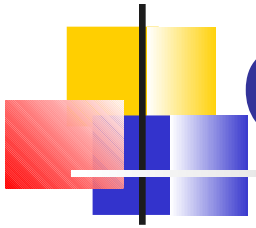
- A vast majority of the tasks present in a typical real-time system are periodic.
- for example, monitoring certain conditions, polling information from sensors at regular intervals to carry out certain action at regular intervals.
- The instances of the temperature, pressure, and chemical concentration monitoring tasks are normally generated through the interrupts received from a periodic timer.

# Sporadic Task

- A sporadic task is one that recurs at random instants.
- A sporadic task Ti can be is represented by three tuples:

$$T_i = (e_i, g_i, d_i)$$

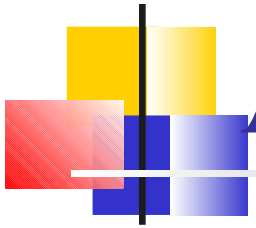
  - ei - worst case execution of instance of the task.
  - gi - the minimum separation between two consecutive instances of the task.
  - di - relative deadline.
- The minimum separation gi indicates that once an instance of a sporadic task occurs , the next instance cannot occur before gi time unit have elapsed.



# Cont..

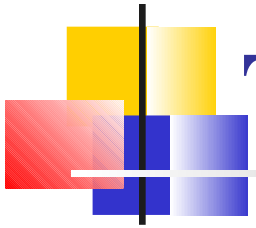
- E.g. In a robot a task that gets generated to handle an obstacle that appears suddenly is a sporadic task.
- In a factory , the task that handles fire condition is a sporadic task.

- Criticality varies from highly critical to moderately critical.
  - E.g. an I/O interrupt or DMA interrupt is moderately critical.
  - Task handling the reporting of fire condition is highly critical.



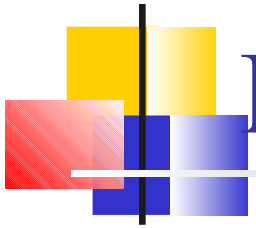
# Aperiodic Task

- An aperiodic task can arise at random instants.
- Two or more instances of an aperiodic task might occur at the same time instant because minimum separation gi,between two consecutive instances can be 0.
- Aperiodic task can recur in quick succession . It , therefore ,becomes very difficult to meet the deadlines of all instances of an aperiodic task.
- Soft real time systems can tolerate a few deadline misses .so ,aperiodic tasks are generally soft real time tasks.
- E.g. Logging task in a distributed systems.
  - The logging task can be started by different tasks running on different nodes.
  - The logging request from different tasks may arrive at the logger almost at the same time.
- Other egs are Keyboard presses, mouse movements etc..



# Task scheduling

- Basic concepts
  - Important concepts and terminologies for schedulers :
    - Valid schedule
    - Feasible schedule
    - Proficient scheduler
    - Optimal scheduler
    - Scheduling Points
    - Preemptive scheduler
    - Utilization
    - Jitter



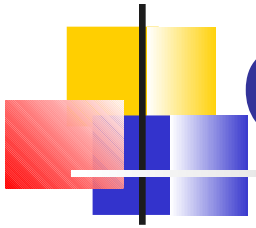
# Description

## Valid schedule

- A valid schedule for set of tasks is one where at most one task is assigned to a processor at a time, no task is scheduled before its arrival time, and the precedence and resource constraints of all tasks are satisfied.

## Feasible schedule

- A valid schedule is called a feasible schedule, only if all tasks meet their respective time constraints in the schedule.



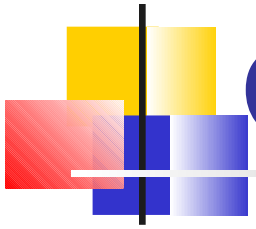
# Cont..

## Proficient scheduler

- A task scheduler sch. 1 is said to be ***more proficient*** than another scheduler sch. 2, if sch. 1can feasibly schedule all task sets that sch. 2can, but there exists at least one task set that sch. 2can not feasibly schedule, whereas sch. 1can.
- If sch. 1can feasibly schedule all task sets that sch. 2can feasibly schedule and vice versa, then sch. 1and sch. 2 are called ***equally proficient schedulers***.

## Optimal scheduler

- Optimal scheduler can feasibly schedule any task set that can be feasibly scheduled by any other scheduler.



# Cont..

- In other words, it would not be possible to find a more proficient scheduling algorithm than an optimal scheduler.
- If an optimal scheduler can not schedule some task set, then no other scheduler should be able to produce a feasible schedule for that task set.

## Scheduling Points

- The scheduling points of a scheduler are the points on time line at which the scheduler makes decisions regarding which task is to be run next.
- A task scheduler does not need to run continuously, it is activated by the operating system only at the scheduling points to make the scheduling decision as to which task to be run next.

# Cont..

## Preemptive scheduler

- A preemptive scheduler is one which when a higher priority task arrives , suspends any lower priority task that may be executing and takes up the higher priority task for execution.
- A preempted lower priority task can resume its execution only when no higher priority task is ready.

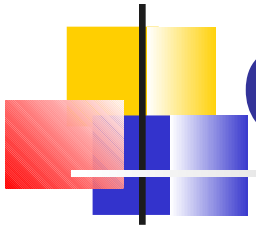
## Utilization

- The utilization of a task is the average time for which it executes per unit time interval.
- For a periodic task Ti , utilization Ui is

$$u_i = \frac{e_i}{p_i}$$

ei – execution time

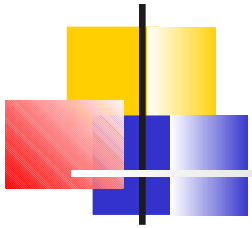
Pi – period of Ti



# Cont..

## Jitter

- It is the deviation of a periodic task from its strict periodic behavior.
- The arrival time jitter is the deviation of the task from arriving at the precise periodic time of arrival.
- It may be caused by imprecise clocks , or other factors such as network congestions.
- The completion time jitter is the deviation of task from its précised deadline.



# THANK YOU !!!!!!!