

# Real Time Operating Systems

LY ETRX  
Sem VIII

Arati Phadke  
AY 2021-22



**SOMAIYA**  
VIDYAVIHAR UNIVERSITY

K J Somaiya College of Engineering



# Computer Organization and Architecture

## Course Outcomes

At the end of successful completion of the course the student will be able to

**C01:** Understand real time operating system concepts.

**C02:** Create various instances of a task and send and receive data to and from a queue

**C03:** Implement the APIs within an Interrupt Service Routine

**C04:** Develop algorithms for real time processes using FreeRTOS

**C05:** Understand memory allocation schemes

# Module 1

## Operating System Concepts

**1.1** Basics of OS: Real time concepts, hard real time and soft real time, difference between general purpose operating system and real time OS, components of an OS, kernel, tasks and threads, Free RTOS

# Need of RTOS

- Embedded Systems
- Producing quality software
- Stages in producing code
- Simple System
- Nature of its Software
- Multitasking Systems
- Need for using Interrupts
- Need for RTOS
  - Provides ready platform for code developer
  - Detailed knowledge of interrupts, timers, analogue-to-digital converters, etc is no longer needed.

# Operating System Concepts

- An operating system (OS) is a collection of software that manages computer hardware resources and provides common services for computer programs.
- Creates platform for application softwares



**Functions of  
OS ???**

# Operating System Functions

- Memory Management
- Processor Management
- Device Management
- File Management
- Security
- Control over system performance
- Job accounting
- Error detecting aids
- Coordination between other software and users

# Operating System Functions

## Memory Management

Memory management refers to management of Primary Memory or Main Memory. Main memory is a large array of words or bytes where each word or byte has its own address.

Main memory provides a fast storage that can be accessed directly by the CPU. For a program to be executed, it must be in the main memory. An Operating System does the following activities

- Keeps tracks of primary memory, i.e., what part of it are in use by whom, what part are not in use.
- In multiprogramming, the OS decides which process will get memory when and how much.
- Allocates the memory when a process requests it to do so.
- De-allocates the memory when a process no longer needs it or has been terminated.

# Operating System Functions

## Processor Management

In multiprogramming environment, the OS decides which process gets the processor when and for how much time. This function is called process scheduling. An Operating System does the following activities for processor management –

- Keeps tracks of processor and status of process. The program responsible for this task is known as traffic controller.
- Allocates the processor (CPU) to a process.
- De-allocates processor when a process is no longer required.



# Operating System Functions

## Device Management

An Operating System manages device communication via their respective drivers. It does the following activities for device management –

- Keeps tracks of all devices. Program responsible for this task is known as the I/O controller.
- Decides which process gets the device when and for how much time.
- Allocates the device in the efficient way.
- De-allocates devices.

# Operating System Functions

## File Management

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions.

An Operating System does the following activities for file management –

- Keeps track of information, location, uses, status etc. The collective facilities are often known as file system.
- Decides who gets the resources.
- Allocates the resources.
- De-allocates the resources.

# Operating System Activities

- **Security** – By means of password and similar other techniques, it prevents unauthorized access to programs and data.
- **Control over system performance** – Recording delays between request for a service and response from the system.
- **Job accounting** – Keeping track of time and resources used by various jobs and users.
- **Error detecting aids** – Production of dumps, traces, error messages, and other debugging and error detecting aids.
- **Coordination between other softwares and users** – Coordination and assignment of compilers, interpreters, assemblers and other software to the various users of the computer systems.

# Operating System Basics

**Block Diagram  
of OS ???**

# Basic features of real-time operating systems

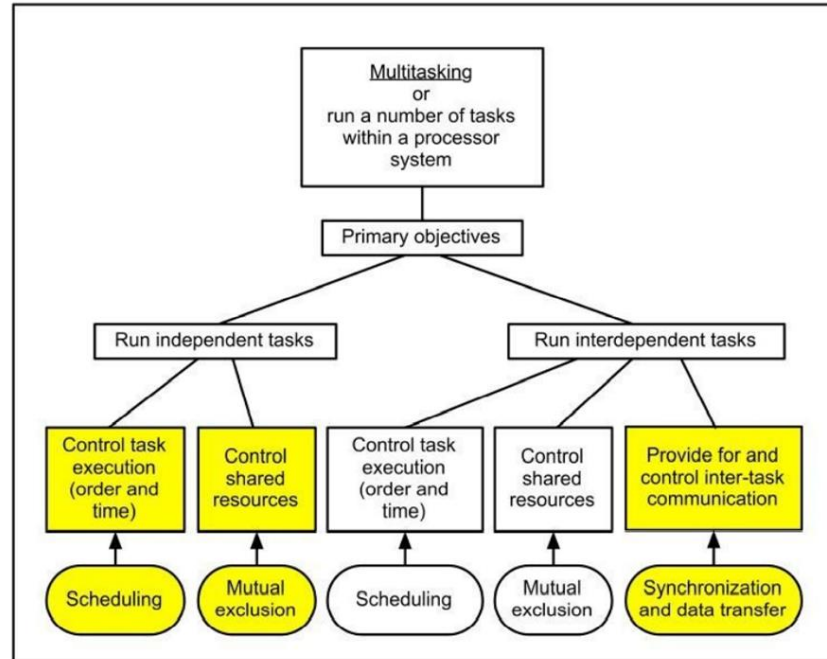
- Task structuring of programs.
- Task implementations as logically separate units (task abstraction).
- Parallelism (concurrency) of operations
- Use of system resources at predetermined times.
- Use of system resources at random times.
- Task implementation with minimal hardware knowledge.

Reduced costs and increased reliability: key uses of OS

# real-time operating systems

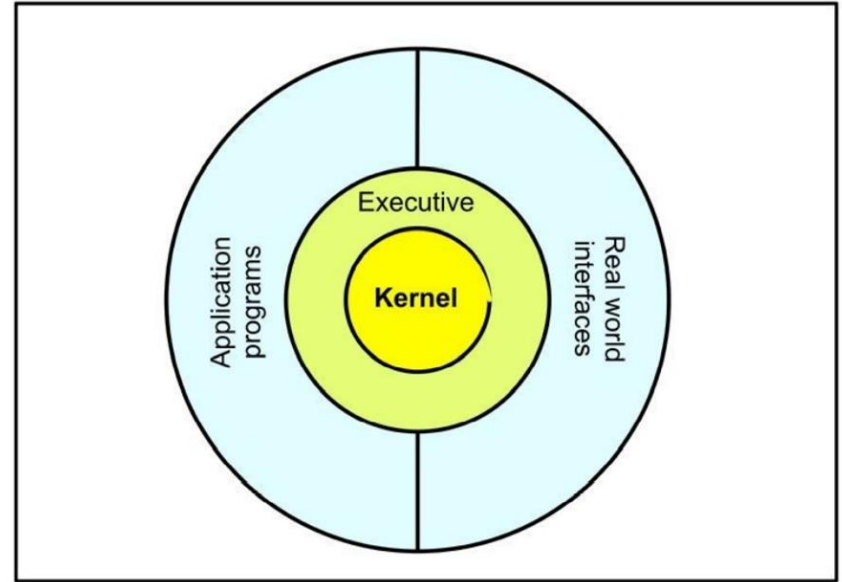
- Consider three separate but interdependent tasks.
- Decide WHEN and WHY tasks should run - 'task scheduling'.
- Monitor use of resources shared between tasks, to prevent damage or corruption to such resources - 'mutual exclusion'
- Task Communication and Synchronization
- Even in case of independent simultaneous tasks , communication and synchronization is required.

# real-time operating systems



# Real Time Operating Systems

- User tasks interface with other system activities (including other tasks) via the executive.
- Executive itself directly controls all scheduling, mutual exclusion, data transfer and synchronization activities.
- Detailed facilities are in kernel
- API
- Hardware interface

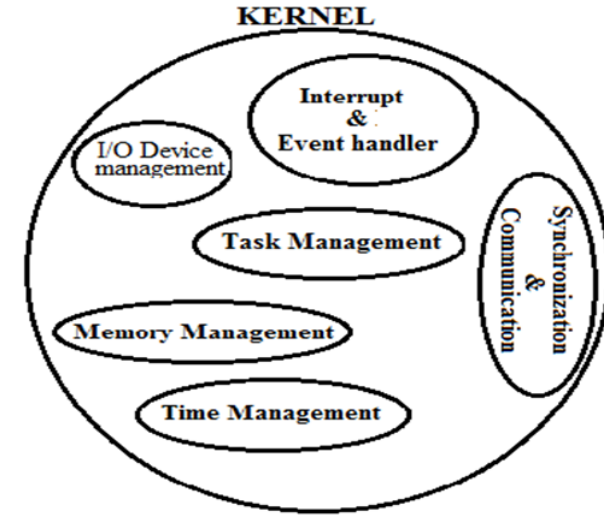


All services provided by the operating system must have bounded as well as reasonable and consistent response times and memory requirements.



# Real Time Operating Systems

- The kernel is the core of an operating system
- It is a piece of software responsible for providing secure access to the system's hardware and to running the programs.
- Kernel is common to every operating system either a real time or non-real time
- The major difference lies in its architecture.
- Kernels has various functions such as file management, data transfer between the file system ,hardware management ,memory management and also the control of CPU time.
- The kernel also handles the Interrupts.



# Kernel Objects

- The various kernel objects are
- Tasks
- Task Scheduler
- Interrupt Service Routines
- Semaphores
- Mutexes
- Mailboxes
- Message Queues
- Pipes
- Event Registers
- Signals and Timers

# Using RTOS

- Each task may be written as if it is the sole user of the system.
- The programmer appears to have complete access to, and control of, system resources.
- If communication with other tasks is required, this can be implemented using clear and simple methods.
- All system functions may be accessed using standard techniques (as developed for that system);
- no knowledge of hardware or low-level programming is needed.
- programmer doesn't have to resort to defensive programming methods to ensure safe system operation.

# Difference in Desktop OS and Embedded OS

- **Desktop OS:**

- OS runs , Boot up time , calls applications
- User code , Application code and OS are separate
- OS protects itself from applications and User codes
- It is general purpose

- **Embedded OS**

- Application runs at boot up time and then calls OS
- Application code and OS are linked together and executed as one binary image
- OS does not protect itself from application
- Dedicated application

# Real Time Concepts

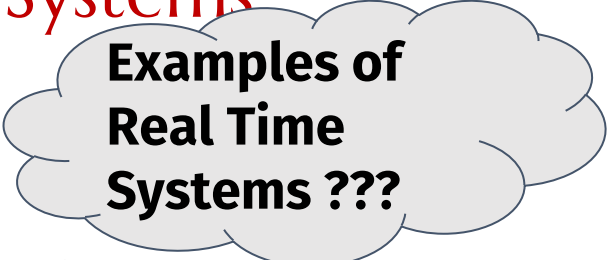
Real-time systems are meant to monitor, interact with, control, or respond to the physical environment.

A system is called a real-time system, when we need quantitative expression of time (i.e. real-time) to describe the behavior of the system.

**A system is said to be Real Time if it is required to complete it's work & deliver it's services on time.**

- The time at which a response is delivered is as important as the correctness of that response, and
- The consequences of a late response are just as hazardous as the consequences of an incorrect response.

# Hard and Soft Real Time Systems



**Examples of  
Real Time  
Systems ???**

- Menti.com use code 84236144
- **Hard Real-Time**
  - Missing a deadline has catastrophic results for the system
- **Soft Real-Time**
  - Reduction in system quality is acceptable
  - Deadlines may be missed and can be recovered from
- **FIRM real time systems**