

## Real Time Programming for embedded systems

### Objective:

- Appreciate the use of multitasking techniques in real-time systems.
- Understand the fundamental concepts of real-time operating systems.
- Understand the features and structures of practical implementations.
- Appreciate how application areas (e.g. safety-critical, desktop, etc.) impact on RTOS facilities.
- Be competent to progress to vendor-specific detailed training.

### Suitable for:

This course is ideal for engineers who are new to the field of real-time. It is also applicable to both managers and engineers who are considering the use of Real-Time Operating Systems on future projects.

### Pre-requisites:

- Knowledge of a high level language (e.g. C, C++, etc.)
- An understanding of the fundamentals of computer-based systems

### Course Details

#### **Introduction to Embedded Systems**

- Basic Components of a System
- Processor
- Memory
- I/O
- Applications

#### **An overview on Embedded Software Life Cycle Development**

##### **Introduction to Real time system**

##### **Types of RTS**

- Hard
- Soft
- Firm

##### **Choosing a right mix**

##### **Types of OS under Embedded Systems**

- Mobile OS
- Embedded OS
- RTOS

##### **Introduction to Real Time Operating Systems (RTOS).**

- Common terms used in Operating systems and also in RTOS.
- Types of kernel: Micro vs. Monolithic
- Commercial RTOS
- An example RTOS: VX-Works
- Other RTOS: A brief on SMX(**H-RTOS**)
- Components of RTOS.

##### **Components of RTOS**

### **Task and Thread**

- What is a process, task, and thread
- The task states
- The control block: Program structure
- The functionalities for a task
- Thread
  - Basic program structure
  - The real need of Thread
    - Parallel execution
    - Parallel Hardware architecture
    - Atomicity support
    - Software architecture
  - Types of thread
    - User/Kernel level threads

### **Introduction to various scheduling policies**

- Some common scheduling mechanism
  - Non-preemptive
  - Cooperative
  - Preemptive
- Static scheduling
- Dynamic scheduling

### **Memory management**

- Address Spaces
- Physical, virtual, linear, and logical memory
- Hierarchy of memory (MTD)
- X86 Memory Model (Typical Example)
  - Segmentation
  - Paging
- A program memory Map [Model dependent]
- Memory allocation and reallocation
  - Fixed partition
  - Dynamic

### **Inter Process Communication**

- Direct Communication
- Indirect Communication
  - Mailbox
  - Pipes
  - Message queue
  - Shared memory
  - Signal
- Inter process Synchronization
  - Semaphores
    - Binary and Counting semaphore.
  - Priority inversion
    - Problems and solution around them.

### **Introduction to Interrupts and Timer**

- ISR fundamental
- Timer Fundamental
- Understanding the Application's need