

RTOS Training

μCOS-II • VxWorks • RT Linux



What is RTOS?

A real-time operating system (RTOS) is an operating system that guarantees certain capability within a specified time constraint. Real-time operating systems are often required in small embedded operating systems that are packaged as part of micro devices.

What makes RTOS so important?

Let's say there is a nuclear power plant and let's say that the temperature is rising & that there is 10 seconds to cool it down before it is too late. But..... Since the plant was running its anti-virus program at the time it missed its window & "Meltdown" happened...!!

That means using RTOS we are meeting deadlines of task & being able to break out of lower priority processes so that more important tasks can be accomplished.

Types of RTOS

Most embedded RTOS's stores an entire binary code image in ROM & it's into just KB's, from which it also executes. The reasons for this are that the code is never expected to change, as embedded devices usually are dedicated to a specific purpose. There are very popular RTOS's like μ COS-II, RT Linux, VxWorks, etc. are available in industries.

Who can join the course?

One can join with prerequisite of having knowledge of 8/16/32 bit Microcontrollers & C programming.

Laboratory:

- To get familiarize with IDE for RTOS - VxWorks/ μ C OS- II. Programming to demonstrate multitasking using Task management.
- Programming for demonstrating Inter-process communication (Pipes, Semaphores, Mailbox, and Message Queues)
- Programming to demonstrate Signals, timers and Interrupts in VxWorks.
- Porting VxWorks Applications on Pentium target Board.
- Porting UCOS-II based applications on Embedded targets (PIC/ARM)

Course Contents:

μCOS-II

Real-time Operating Systems Concepts : Definition • Type's of RTOS • Soft Real Time and Hard Real Time Systems • Foreground/Background Systems • Critical Sections of code • Resources and shared resources • Multitasking, Tasks • Context Switches, Kernels • Schedulers

Kernel Structure : Tasks, task states • task control Blocks • Ready List • Task Scheduling • idle Task • statistics task • Interrupts under μC/OS-II • How to initialize and start μC/OS-II • locking and unlocking the scheduler • clock tick • Task priorities

Task Management & Time Management : How to create and delete a task • check the size of a task's stack • how to change a task's priority • how to suspend a task, resume a task • get information about a task • How to delay a task's execution • resuming a delayed task • How to get and set the system time.

Intertask Synchronization: Semaphore Management • Mutual Exclusion Semaphore

Event Management: Event Flag Internals • creating and deleting event flag group • waiting for event(s) • setting and clearing event(s)

Intertask Communication: Message Mail box management • Message Queue Management

Memory Management: Memory Control Blocks Creating a partition • obtaining a memory block • returning a memory block • obtaining status of memory partition.

RT-Linux

Introduction to Linux OS : operating system services • why Linux • Different types operating systems • Monolithic • Micro etc • Basic Linux user commands • Linux root file system structure.

Introduction to GNU Tool chain: GCC compiler • Make file • GDB

Operating system basics: Operating system structure, kernel Architecture, system calls • Process Management (PCB, Process creation etc...)(LAB-4) • Programs on process management • Memory Management (Paging, Swapping, Segmentation Virtual Management etc...)• Scheduler (scheduling policies) • File management, Open, read Write system Calls

RTOS basics -Linux as Real Time : RTOS Introduction (Hard Real Time, Soft Real Time) • Latency in Linux, Priority Inheritance • Linux 2.6 features for real-time • 2.6 Kernel Compilation

RT LINUX patching : Linux RTPREEMPT Patches • Configuring the Kernel with RT-PATCH • debugging the Real time Kernel • High Resolution Timers, The Latency Tracer

Implementation of Real Time application

Linux real-time API.

Testing the Real-time preempt patch: Measuring and comparing scheduling latency in standard Linux and in Linux with the latest RT patches.

VxWorks

Overview of VxWorks: Definition of Real-Time operating system • Type's of RTOS • Soft Real Time and Hard Real Time Systems • Foreground/Background Systems • Critical Sections of code

Project Management: Resources and shared resources • Multitasking • Tasks • Context Switches • Kernels, Schedulers • Non-preemptive Kernels • Preemptive kernels • Reentrant functions

Host Shell: Task priorities • static and dynamic priorities • priority inversion • mutual exclusion • Semaphores • Intertask • communications • Interrupts

Real-time Multitasking: Tasks, task states • task control Blocks

Real-time Processes: Locking and unlocking the scheduler • clock tick

Semaphores: How to create and delete a task • check the size of a task's stack • how to change a task's priority

Inter-task Communication: How to suspend a task • resume a task, and get information about a task • How to delay a task's execution • Resuming a delayed task • How to get and set the system time.

Message Channels: Semaphore Management

Memory Management: Mutual Exclusion Semaphore

Error Detection and Reporting: Event Flag Internals • creating and deleting event flag group • waiting for event(s) • setting and clearing event(s)

Debugging and Analyzing: Message Mail box management • Message Queue Management

Exceptions, Interrupts, and Timers: Memory Control Blocks • creating a partition • obtaining a memory block • returning a memory block • obtaining status of memory partition

Indus Institute of Embedded Technologies (IIET), Pune.

2nd Floor, Above Bank of Maharashtra,
Near Modern Cafe
Shivaji Nagar, Pune

Website- www.embeddedssystemspune.com

Email: info@bicard.org

Office No: **020-30209696**

Mob No: **9372202549**
