Embedded Systems & RTOS(404184)

Teaching Scheme:
Lectures: 3 Hrs/ Week

Examination Scheme:
In Semester Assessment:
Phase I : 30
End Semester
Examination:
Phase II: 70

Course Objectives:

- To understand the Embedded system design issues.
- To learn real time operating system concepts.
- To understand the Embedded Linux environment
- To learn Embedded software development and testing process.

Course Outcomes:

After successfully completing the course students will be able to

Get insight of design metrics of Embedded systems to design real time applications to match recent trends in technology.

Understand Real time systems concepts.

Understand Linux operating system and device drivers.

Get to know the hardware – software co design issues and testing methodology for Embedded system.

Unit I :Introduction to Embedded Systems 6L

Introduction to Embedded Systems, Architecture, Classification and Characteristics of Embedded System, Design Process, Design Metrics and optimization of various parameters of embedded system. Embedded processor technology, IC technology, Design technology. Software development life cycle. Various models like waterfall, spiral, V , Rapid Prototyping models and Comparison

Unit II : Real Time Systems Concepts 6L

Foreground/ Background systems, Critical section of code, Resource, Shared resource, multitasking, Task, Context switch, Kernel, Scheduler, Non-Preemptive Kernel , Preemptive Kernel, Reentrancy, Round robin scheduling, Task Priorities, Static & Dynamic Priority, Priority Inversion, Assigning task priorities, Mutual Exclusion, Deadlock, Clock Tick, Memory requirements, Advantages & disadvantages of real time kernels.

Unit III : µCOS II 6L

Features of µCOS II. Kernel structure. µCOS II RTOS services: Task management, Time management, Intertask Communication and Synchronization.

Unit IV : Embedded Linux Development Environment 6L

Need of Linux, Embedded Linux Today, Open Source and the GPL, BIOS Versus Boot loader, Anatomy of an Embedded System, Storage Considerations, Embedded Linux Distributions. Embedded Development Environment, Cross-Development Environment, Host System Requirements, Hosting Target Boards. Development Tools, GNU Debugger, Tracing and Profiling Tools, Binary Utilities.

Unit V : Linux Kernel Construction 6L

Linux Kernel Background, Linux Kernel Construction, Kernel Build System, Kernel Configuration. Role of a Bootloader, Bootloader Challenges. A Universal Bootloader: Das U-Boot. Porting U-Boot. Device Driver Concepts, Module Utilities, Driver Methods. Linux File System & Concepts

Unit VI : Embedded Software Development, Testing Process and Tools 6L

Embedded Software development process and tools, Host and Target Machines, linking and Locating Software, Getting Embedded Software into the Target System, Issues in Harware-Software Design and Co-design. Testing on Host Machine, Simulators, Laboratory Tools. Case study of Embedded system like Automatic Chocolate Vending Machine, Mobile Phone.

Text Books

1. Jean J. *Labrosse, "*MicroC OS II, The Real-Time Kernel", 2nd edition, CMP Books.
2. Christopher Hallinan, "Embedded Linux Primer -A Practical, Real-World Approach "2nd edition, Prentice Hall.

Reference Books

1. Raj Kamal, "Embedded Systems – Architecture, Programming and Design" 2nd edition, Mc Graw Hill.
2. Frank Vahid and Tony Givargis, " Embedded System Design – A Unified hardware/ Software introduction " 3rd edition, Wiley.

List of Experiments:

Group A: LPC2148 & μCOS - II Based Experiments (any four)

1. Multitasking in μCOS II RTOS using minimum 3 tasks on LPC2148.
2. Semaphore as signaling & Synchronizing on LPC2148.
3. Mailbox implementation for message passing on LPC2148.
4. Queue implementation for message passing on LPC2148.
5 Implementation of MUTEX using minimum 3 tasks on LPC2148.

Group B: ARM9 & LINUX Based Experiments (any four)

6. Download pre-configured Kernel Image, File System, bootloader to target device- ARM9.
7. Writing simple application using embedded linux on ARM9.
8. Writing "Hello World" device Driver. Loading into & removing from Kernel.
9. Write a program for I2C based RTC using embedded linux on ARM9.
10. Using Device driver for GPIO, write a program to blink LED.
11. Write a program for External Interrupt.