Syllabus:

Post Graduate Diploma in Embedded Real Time Systems

NIELIT Chennai Centre,
ISTE Complex, NO. 25, Gandhi Mandapam Road ,
Chennai - 600025
chennai@nielit.gov.in
Phone: 044-24421445/47 Fax: 044-24421441

Module: 1/8

Objectives:

This module is framed to set the required background in embedded system concepts and ‘C’ language for the rest of the modules. This module covers the advanced topics in ‘C’ such as Memory management, Pointers, Data structures and covers the architecture of the popular 32-bit bit Microcontroller such as ARM. The ARM Cortex processor is the industry-leading 32-bit processor for highly deterministic real-time applications.

Outcomes:

After successful completion of the module, the students shall be able to:

- Develop Embedded application using Embedded C Programming
- Use ARM Cortex M with Embedded C Programming for Application Development

Duration:

140 Hours

Module topics:

Embedded Concepts
Introduction to embedded systems, Application Areas, Categories of embedded systems, Overview of embedded system architecture, Specialties of embedded systems, recent trends in embedded systems, Architecture of embedded systems, Hardware architecture, Software architecture, Application Software, Communication Software, Development and debugging Tools.

‘C’ and Embedded C
Introduction to ‘C’ programming, Storage Classes, Data Types, Controlling program flow, Arrays, Functions, Memory Management, Pointers, Arrays and Pointers, Pointer to Functions and advanced topics on Pointers, Structures and Unions, Data Structures, Linked List, Stacks, Queues, Conditional Compilation, Pre-processor directives, File operations, Variable arguments in Functions, Command line arguments, bitwise operations, Typecasting.

Introduction to ARM Cortex
Architecture Introduction to 32-bit Processors, The ARM Architecture, Overview of ARM, Overview of Cortex Architecture, Cortex M3 Register Set and Modes, Cortex M3 Processor Core, Data Path and Instruction Decoding, ARM Cortex M3 Development Environment, Assembler and Compiler, Linkers and Debuggers, ARM, Thumb & Thumb2 instructions, Mixing ARM & Thumb Instructions, Memory hierarchy, Memory Mapping, Cache.

Cortex M3 Microcontrollers & Peripherals
Cortex M3 based controller architecture, Memory mapping, Cortex M3 Peripherals – RCC, GPIO, Timer, System timer, UARTs, LCD, ADC, Cortex M3 interrupt handling – NVIC .Application development with Cortex M3 controllers using standard peripheral libraries.

Module: 2/8

Objectives:

The objective of the course is to provide understanding of the techniques essential to the design and implementation of embedded systems with embedded operating systems.

Outcomes:

After successful completion of the module, the students shall be able to:

- Implement embedded systems with Embedded operating systems
- Develop applications with Embedded Linux

Duration:

70 Hours

Module topics:

Introduction

- Basic Operating System Concepts,
- Linux as Embedded Operating System,
- Comparison of Embedded OS,
- Embedded OS Tools and Development and Discussion on Embedded OS Applications and Products

System architecture of a Basic OS

- Internals of Linux OS System Calls,
- Linux Compiler options, Make Process,
- Multithreading and Synchronization,
- Serial port and Network programming with Embedded Linux,
- Kernel module programming and Device drivers.

Inter Process Communication

- Pipe and FIFOs,
- Shared memory,
- Sockets.

Getting Linux on a device

- Linux boot sequence,
- Building Kernel,
- Building Boot image.

Practical Sessions

- Embedded Linux Applications

Embedded RTOS

Module: 3/8

Objectives:

The objectives of the course is to provide the students with an understanding of the aspects of the Real-time systems and Real-time Operating Systems and to provide an understanding of the techniques essential to the design and implementation of real-time embedded systems. This course covers two popular real time operation systems FreeRTOS /VxWorks.

Outcomes:

After successful completion of the module, the students shall be able to:

- Developan Embedded Real Time software that is required to run embedded systems
- Apply the FreeRTOS RTOS for real-time application development
- Develop real-time applications using FreeRTOS /VxWorks RTOS
- Build real-time embedded systems using FreeRTOS / VxWorks RTOS.

Duration:

140 Hours

Module topics:

Introduction

- Embedded Software – Real-time Vs Non Real-time,
- Introduction to Real-time systems and Embedded Real-time Systems,
- Discussion of popular RTOS like FreeRTOS / VxWorks,
- Comparison of Embedded RTOSs (FreeRTOS /VxWorks),
- Design Goals for Real-time software,
- Discussion on Embedded Real-time applications,
- Considerations for real-time programming

System architecture of FreeRTOS

- Introduction FreeRTOS,
- Thread Creation and Management,
- Thread Synchronization Mechanisms,
- IPC – RTFIFO, Shared Memory, Interrupt Handling

System architecture of FreeRTOS **/VxWorks**

- Introduction to FreeRTOS /VxWorks,
- Task Creation and management,
- Inter Task Communication Mechanisms,
- Semaphores, Message Queues, Pipes,
- Interrupts, Tornado tools.

Practical Sessions

- Application development under FreeRTOS / VxWork.

Porting on ARM Cortex Microcontrollers

Module: 4/8

Objectives:

The ARM architecture (32-bit) is the most widely used architecture in mobile devices, and most popular 32-bit one in embedded systems The ARM Cortex processor is the industry-leading 32-bit processor for highly deterministic real-time applications, specifically developed to enable partners to develop high-performance low-cost platforms for a broad range of devices including microcontrollers, automotive body systems, industrial control systems and wireless networking and sensors.

Outcomes:

After successful completion of the module, the students shall be able to:

- Comprehend the Kernel Compilation Procedure for ARM and use it for Application Development
- Apply Porting of Open Source Operating Systems on ARM Cortex Microcontrollers

Duration:

70 Hours

Module topics:

Porting RTOS to ARM Cortex Microcontrollers

- Building root file system, Kernel Compilation for ARM, Porting of OS to ARM.
- Overview of open source RTOS (Chibi-OS / FreeRTOS / MicroC-OS etc.),
- Porting open source - Embedded OS (Linux) & other RTOS (Chibi-OS / FreeRTOS / MicroC-OS etc.) on ARM Cortex Microcontrollers.
- RTOS based applications development on Cortex Microcontrollers.

Internet of Things (IoT)

Module: 5/8

Objectives:

The Internet of Things (IoT) is a scenario in which objects, animals or people are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction. The Internet of Things (IoT, sometimes Internet of Everything) is the network of physical objects or "things" embedded with electronics, software, sensors and connectivity to enable it to achieve greater value and service by exchanging data with the manufacturer, operator and/or other connected devices. Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing Internet infrastructure. Experts estimate that the IoT will consist of almost 50 billion objects by 2020.

Outcomes:

After successful completion of the module, the students shall be able to:

- Apply the concepts of IoT Architecture and Layering
- Implement IoT applications using proper hardware and software platforms
- Develop IoT Applications with Aurdino and other platforms

Duration:

70 Hours

Module topics:

- Introduction to IoT, WoT and M2M
- Basics of Internet & Review of Internet protocols
- Data logging /IoT Layering concepts
- Wireless PAN (Bluetooth & Zigbee), GSM, Wifi
- Introduction to Wireless Sensor Networks
- Routing Protocols in WSN
- Database Management

Embedded Protocols & Device Drivers

Module: 6/8

Objectives:

Today, at the low end of the communication protocols, we find I^2C and SPI Both protocols are well-suited for communications between integrated circuits, for slow communication with on-board peripherals. This module will brief the basics of protocols and programming aspects.

Device drivers take on a special role in the Linux kernel. They are distinct “blackboxes” that make a particular piece of hardware respond to a well-defined internal programming interface; they hide completely the details of how the device works. This module will brief the programming aspects of Linux device driver The course is structured to include the learning of SPI, I2C, USB, CAN protocols and the Linux device driver implementation . Hands own experiments and a mini-project are included in the module.

Outcomes:

After successful completion of the module, the students shall be able to:

- Understand SPI, I2C, USB, CAN enabled devices in embedded application
- Develop device drivers for Linux

Duration:

105 Hours

Module topics:

Embedded Concepts:

- Embedded Protocols
- Overview of Embedded TTY, I2C protocols, SPI, CAN Processor Bus, USB
- Overview of Linux Device drivers
- Linux Drivers overview, Review of Kernel 'Embedded C' Programming, Device driver developing Environment ,The First driver.
- The Character driver: Name vs Number, Registration & the Cleanups, Kernel Data Structures & File Operations, Linux Device Model & Bus Architectures, Analog & Digital I/Os
- Low-level Accesses: Memory Access, Hardware Access.
- USB Drivers: Device & Driver Layout, USB Core, Driver & Device Registration,
- USB & its Functionalities.
- Interrupts: Interrupts & IRQs, Soft IRQs, and Exceptions.
- Block Drivers
- File System Modules: Virtual File System, The Five Operation Sets, Interaction with the Block Device
- Network Drivers

Seminar and case study

Module: 7/8

Objectives:

Each Student is required to choose a topic of their interest for seminar from any Embedded domain which is not covered under the course curriculum. The duration of the seminar is about 20 minutes with the support of power point presentations.

A committee consisting of at least two faculty members (preferably specialized in Embedded Systems) shall assess the presentation of the seminar and award marks to the students.

Each student shall submit two copies of a write up of his/her seminar topic. One copy shall be returned to the student after duly certifying it by the course coordinator. Internal continuous assessment marks are awarded based on the relevance of the topic, presentation skill, quality of the report and participation.

Outcomes:

After successful completion of the module, the students shall be able to:

- learn new / latest trends in embedded real time systems
- Design and develop a various case studies related to real-time embedded systems.

Duration:

35 Hours

Project Work

Module: 8/8

Course Description:

The students can select hardware, software or system level projects. The project can be implemented using Microcontroller, Device driver or RTOS tools which students have studied and used during the course. A total product or project can be selected.

The objective of this module is to help fresh graduates and practicing engineers to enhance their knowledge and skills of embedded product design covering the various aspects of product development process and design of a stand- alone embedded system.

Duration:

210 Hours