**Course name:** Professional Diploma in Embedded Systems

**Description:** This intense course is primarily aimed at Freshers / Engineers / Professionals who wants to have a Career into Embedded Systems domain.

**Benefits of attending:**

Students get the knowledge and experience to be productive in any company with the skills acquired. A comprehensive expert level course covering all aspects of Embedded systems Design.

**Salient Features:**

- Industry Ready Course
- Practicals with real hardware exposure
- Trainers with industry experience
- Interview Preparation workshops and Mock Interviews
- Courseware in sync with industry needs

**Course Duration**

Regular mode – depends on the selected modules, the complete course duration: 5 Months (daily 1 ½ hours)

**Course Delivery**

- Lectures, Classroom Discussions and Lab Exercises

**Course Contents**

This course is divided into three logical modules to enable easy and effective understanding.

## I. Advanced C , Object Oriented Programming

### C language

- *Introduction to C Programming - Structure of a C program, The C compilation process*
- *Types and Operators - C base types, Precedence & Associativity, - Arithmetic operation, Promotion & Typecasting*
- *Control Flow - Logical expressions and operations, Decision Making, Loops*
- *Definitions and declarations, Header files, Scope and lifetime - Storage Classes*
- *Introduction to pointers - Using pointers to access single dim arrays*
- *Bit Manipulation, Bit level manipulation,*
- *Functions - The Function as a logical program unit, Parameter passing by copy and reference*
- *Arrays, Pointers and Strings - Arrays as circular buffers, Relationship between pointers & arrays, Pointer arithmetic, C string handling*
- *Advanced Data types - Structures, Unions and Enums - Structures, Big & Little Endian representations, Unions, Bit-field structures*

- *Dynamic Memory Management - Malloc and free, Issues - leaks, fragmentation, etc.*
- *Data Structures - Linked lists, Stacks, Queues, Binary Search trees, applications*
- *Search and Sort algorithms - Bubble sort, Quick Sort etc, when and why to use*
- *The C Pre-Processor - Macros, Conditional Compilation C Compilation process - pre-processor, compiler, assembler, linker stages*

### C++

- Overview
- Characteristics
- Function Overloading
- Scope Resolution Operator
- Classes in C++
- Access Specifiers
- Constructor, Destructor
- Static members, Functions
- Friend Classes, Friend Functions
- Operator Overloading
- Data Conversions
- Inheritance, Polymorphism
- Exception Handling, Templates
- Input and Output Streams

## II. ARM

**Module 1: Introduction**

- Introduction of ARM Processors
- Evolution of ARM
- 32 - bit Programming

**Module 2: ARM7 Architecture**

- ARM7 Architecture
- LPC21xx Description
- Memories
- Peripherals

**Module 3: ARM Processor Programming**

- ARM Processor Programming in C
- Using ARM Programming Tools

**Module 4: IO Device Interface and practical**

- Study of Input Output Devices
- LED Interfacing
- LCD Interfacing
- Serial Communication Concepts
- I2C
- SPI
- Stepper Motors and DC Motor Interfacing
- Practices on Boards

**Module 5: Advance IO**

- ADC
- GSM Module
- Practices on Board

## III.LINUX

### *Linux System Programming*

***GNU Toolchain & Libraries***

- GCC (GNU Compiler Collection)
- GNU Makefile
- GDB (GNU Debugger)
- Types of Libraries
- Procedure for creation of Static and Dynamic Libraries

**File Management**

- Linux File Structure
- Difference between System call and Standard Libraries.
- Open,read,write,ioctl,close and mmap system calls.
- /Proc and /Sys file Systems

**Process Management**

- Process Concept
- Process Scheduling
- Process Creation

**POSIX Threads**

- Introduction to POSIX thread interface
- Thread creation and management
- Thread attributes
- Detecting Race conditions
- Atomic operations
- Mutual exclusions methods (mutex, semaphores, spinlocks)
- Detecting and handling deadlock events
- Choosing right Mutual exclusion method
- Designing scalable critical sections
- Exploring Thread synchronization methods (signals, condition variables...)

**Inter-process Communication**

- Signals, its importance
- Pipes and FIFO's
- Semaphores
- Shared Memory
- Message Queues
- Sockets

## *Linux Device Drivers*

**Introduction to Linux Kernel & Device drivers**

- Two types of Kernel
- Linux Source tree Overview
- Configuring, Compiling and Booting the Linux Kernel Configuration
- Booting the kernel.
- What is Device Driver?
- Types of Device Drivers
- Classes of Device drivers
- The Role of the Device Driver
- Types of Kernel

**Module Programming**

- What is a Kernel Module?
- User mode vs Kernel mode
- Our First Linux Driver
- Building Our First Linux driver
- Module parameters
- Module dependency
- Kernel Specific GCC Extensions (__init and __exit)

**Character Device Drivers**

- What is CDD?
- The Complete connection.
- Major and Minor numbers.
- Implementation of Character Driver.
- The complete Memory driver
- The complete Character Device Driver.
- Dynamic Character Device Driver
- Multiple Character Device Driver

**Synchronization techniques**

- Concurrency and Its Management
- Semaphores and Mutex
- Spinlocks

**Advanced Character Device Drivers**

- ioctl
- Blocking I/O
- poll and select

**Communication with Hardware**

- I/O Ports and I/O Memory
- Generic Hardware Interfacing
- Using I/O Ports
- Using I/O Memory

**Interrupt Handling**

- Process context vs Interrupt context
- Installing an Interrupt Handler
- Interrup Handler Constraints
- Handler argumnets and Return Values
- Interrupt Control Methods.
- Top and Bottom Halves
- Examples

**Kernel Mechanisums**

- Kernel Threads
- Kernel Timers
- Workqueues

**Memory Management and Allocation**

- Memory management in Linux

**GPIO**

- GPIO framework in Linux
- GPIO Driver customization

**I2C**

- i2c subsystem
- Writing Client Drivers

## *Embedded Linux*

- Embedded Systems Booting process
- Boot-Loader – u-boot customization
- Linux Kernel customization
- User-Space
- Kernel-Space
- C Libraries, Building a Cross compiling tool chain
- Configuring cross compiling toolchain
- kernel bootup flow
- NAND vs NOR
- flash file system
- Boot time optimization
- Linux Porting on ARM9 Board

## *Miscellaneous*

**User space tools**

- GIT
- GDB, gdb server
- strace, valgrind

**Kernel Space tools**

- Kernel OOPS
- Printk. Dmesg
- kprobe and jprobe
- KDB
- KGDB