CORE TECHNOLOGY COURSES

For Industry Professionals and Engineering Students

From

ELSARA TECHNOLOGIES (P) LTD
18/4, Vaderhalli, Vidyaranyapura P.O, Bangalore -560097, Karnataka, India
Contact : +919447457762 | sales@elsara.in

PROLOGUE

Technology is ubiquitous.

Over the last few decades, advancements in computer systems have made computational power affordable and within the reach of common man. Technology has thus taken a formidable turn embracing these advancements in evolving and acquiring a software approach in its form and deploy. Advancements in Software technologies have reached an impressive level revolutionising how the technology can be perceived, learned and taught.

Eminent software have emerged that integrate design with analytical and simulation capabilities for each category of technology. The simulation engines from these software abstract the complexities of technology providing conveniences, scope for research and innovation. These tools automate with ease the mundane tasks of iterative computation and big data analysis.

Modern industry now trends with software as the means for research to advance technologies and for innovations on the application side of technology. Sophisticated software is now available for most categories of technologies and reliance on software becomes eminent when time to market matters. These software are well positioned within the industry for design and development of reliable and robust systems.

It is pertinent for students and industry professionals to practice technology on software and acquire application development skills on these tools for better adaptation to Industry needs and trends.

It is in this backdrop of cutting edge software taking over industry that we bring forth the Core Technology Courses(CTC) program that offer training on core technologies acquainting and teaching technology from a software perspective.

Our technology training courses are on leading software used in industry with emphasis on application development. The program combines both proprietary & open source wherever possible to bring forth a flavour that is in industry practice. The CTC training programs are conducted by professionals from industry and prepare students and industry professionals to be adeptly geared up for the industry requirements.

I wish all our budding engineers and professionals a very bright and vibrant career and sincerely desire that they achieve greater distinction with lots of fame and global acclaim …..

Truly,

George Kottackakathu Thomas
Chief Executive Officer
Elsara Technologies (P) Ltd., Bangalore

Training Course Contents

For **Electronics & Communications (ECE)** discipline

(click headings to navigate to course contents)

S/W & H/W Tools available from us

- Boundary Scan kits
- Keil Software for ARM and 8051 development
- Altium Designer for PCB Design
- Arduino Starter kit with Logic Analyzer
- University IoT Lab
- Displays with Graphic development software
- ARM boards (LPC Boards, Beaglebone, Panda, HawkBoard), trainer kits

Training Course Contents

For **Electrical/Instrumentation Engineering discipline**

- **Siemens training programme on basic Automation** – 5 days full time
- **Siemens training programme on basic Process Instrumentation** – 5 days full time
- **Siemens training programme on basic Scada** – 5 days full time
- **Siemens training programme on basics of AC/DC Drives** – 5 days full time
- **Siemens training programme on basics of Induction Motor** – 3 days full time
- **Siemens training programme on low voltage Switchgear** – 5 days full time
- **Siemens training programme ON SMSCP LEVEL - 1** – 30 days full time at Siemens training facility

Tools available from us

- **Low cost Industrial Automation Trainer Packages from the Siemens Co-operates with Education (SCE) program**

  SCE Trainer Packages offer a speci fic combination of original industrial components to cover both the factory and process automation sectors and can be conveniently used in your classes or training laboratory to conduct the complete course on industrial automation at a very low cost.

  Currently more than 90 SCE Trainer Packages are offered including all needed accessories. These cover both the factory and process automation sectors. These price-reduced bundles available exclusively to educational institutions include innovative hardware and software products.

- **SIMATIC IOT2020**

  SIMATIC IOT2020 enables students to get hands-on experience in the first years of their studies or in demanding projects i.e. in

applications with sensor and control technology with open and closed loop control in high-level languages.

The IOT2020 is built around the Intel Quark® x1000 CPU, the open source Yocto Linux operating System and the Intel System Studio IOT Edition, to implement projects in C/C++ Code in Eclipse and other high-level programming languages by educators and students.

Regardless of the major – electrical, automation or process engineering, computer science, building technology, machine building, mechatronics, robotics or embedded systems the IOT2020 is the right open platform for colleges /universities and vocational training schools.

- **MotorSolve - ELECTRIC MACHINE DESIGN SOFTWARE**

MotorSolve version 6 is the complete design and analysis software for induction, synchronous, electronically and brush-commutated machines. Motor and generator designers can use this software for quick virtual prototyping.

MotorSolve simulates machine performance using equivalent circuit calculations and our unique automated finite element analysis engine. Typical FEA operations, such as mesh refinements and post-processing, are not required as MotorSolve handles these for the user. The template-based interface is easy to use and flexible enough to handle practically any motor topology. Custom rotors and stators can be imported via DXF.

Training Course Contents

For **Mechanical** Engineering discipline

1. **Automotive Crash & Impact Analysis using LS-Dyna**
2. **ANSYS Tool Training**

These courses provides an introduction to various products of ANSYS Software

The course focuses on the use of Mechanical / CFD / E-mag modules. In Mechanical / CFD / E-mag module courses you read/create CAD geometry, assign material, propreties, apply loads and boundary conditions, define mesh controls, perform solutions, review analysis results and generate a report with validation where ever possible.

The course devotes time to theory and concepts at a very basic and practical level. These portions of the course emphasize practical theory concepts, which engineers need to understand in order to do finite element analysis

Ansys tool training programs include training on the following Ansys products

- **ANSYS Mechanical**
- **ANSYS Mechanical APDL**
- **ANSYS Fluent**
- **ANSYS CFX**
- **ANSYS Workbench**
- **ANSYS Meshing**
- **ANSOFT (Both HF and LF)**
- **ANSYS + CivilFEM**

Boundary Scan Technology (1149.1) Course Contents
Duration : 1 Day

Introduction

- Need for Debugging, difference between debugging and testing
- Debugging jobs and types, scope for employability (for Engineering Schools and Skill development centres)

Genesis of Boundary Scan technology

- PCB faults and traditional methods of fault diagnosis
- Problem of Limited Access with traditional methods of structural testing
- Nascence of Boundary Scan from addressing problem of limited Access
- The Boundary Scan device as a solution for addressing problem of limited Access

Boundary Scan Device Architecture

- The access solution - A quick peek on the Architecture
- Boundary Scan cell
    - The basic boundary Scan cell (type BC_1)- Architecture with operation explained
    - Signals of a Boundary Scan cell (type BC_1)
    - Operational modes of a Boundary Scan cell (type BC_1)
    - 2 Boundary Scan cells connected to form a Scan chain for real time monitoring explained
    - Scan cell implementation of a Bidirectional I/O pin and tristate output pins
    - Different types of boundary scan cells illustrated
- Test Access port (TAP) explained with design consideration for its signals
    - Details about Test Clock (TCK) and the need for independent clock
    - The Need for TRST signal and design consideration with System Reset
- Registers and Instruction set of Boundary Scan Architecture
    - Instruction Register explained with its function
    - Data register sets of Boundary Scan Architecture
    - Instruction set of Boundary Scan Architecture and purpose of instructions
    - The need for 'bypass' register
- Test Access Port (TAP) Controller
    - Tap Controller operation explained - The finite State Machine and its flow explained
    - The role of TCK in Tap Controller operation
    - Control Signals Generated by the Tap controller
    - How TAP controller multiplexes Instruction Register and Data register between TDI and TDO
    - TAP controller operation for an Instruction 'EXTEST'

Boundary Scan Description Language of IEEE 1149.1

- Introduction to HSDL, BSDL, SVF
- Genesis of Boundary Scan Description Language (BSDL) and why this data format is required
- BSDL Syntax and elements explained in detail with reference to boundary scan device implementation examples

How Boundary Scan technology finds use in niche applications

- Debugging assembled PCB interconnect faults
    - A typical PCB with boundary scan devices and daisy chaining illustrated
    - Tap Signal configurations (Ring & Star)
    - Assembled PCB faults - a quick revist
    - Modelling PCB interconnect faults and types of interconnect faults detected
    - PCB interconnect fault debug - A test strategy in general with tap operations explained for the test strategy
    - The no. of tests required
    - Connection testing test strategy
    - The case of conflicting algorithm
    - Test strategy for interconnect debug with non-jtag devices on board
    - Test Strategy for testing RAM interconnections
    - Boundary scan testable memories
- CPU core access for software on Hardware debug
    - Insights into debugging methods and history
    - How Software debug happens utilising boundary scan technology explained
    - The difference between JTAG boundary scan testing and JTAG software debugging
- JTAG high Speed Flash programming
- Embedded Instrumentation

How Boundary Scan Technology is abstracted by Boundary scan based tools available in market

Introduction to Design for Test (DFT) principles (Hardware board design)

Boundary Scan related IEEE standards overview

Conclusion

ARM Architecture Training (IoT) - I with Hands on projects

Duration: 1 to 5 days (based on college requirements)

Course contents:

1. Processor Architectures

- CISC and RISC Architecture
- Von-neumann and Harvard Architecture

2. Classification of ARM Processors

- Application Processor
- Real-time Processor
- Microcontroller

3. ARM Cortex-M3/M4 Architecture

- Processor Core, Register Set
- System Interfaces
- Clocks, Reset and Power
- Operating Modes
- NVIC
- Memory Model
- Systick Timer
- Exception and Interrupts
- Bus Interfaces
- Memory Protection Unit
- Low Power and System Control Features
- Floating Point Unit
- Introduction to CoreSignt for Debug

4. ARM Instruction Set

- Data Processing Instructions
- Branch Instructions
- Load Store Instructions

5. Keil Microcontroller Development Kit for ARM

- Creating a project
- Writing Code in C and Assembly
- Compiling the and executing the code on development board
- Debugging the Code

6. Hands on lab Assignments with LPC development board

- Accessing OnChip Peripherals
- General Purpose I/O
- Timers, Watchdog, PWM Modulator
- UART, I2C, SPI Interfaces
- Interrupt Service Routines
- Hardware Debugging Tools

Hands on Projects based on Raspberry or Arduino boards

1. Weather monitoring station (IoT Category)
2. Smart home - complete home automation (IoT)
3. Motion Surveillance using RPi (IoT)
4. Flow control sensing using Raspberry Pi.
5. Mobile phone operated lock using RPi
6. Lock which opens based face recognition
7. HDMI Picture frame with RPi for hotels and large rooms for displaying wall hanging pictures
8. Smart phone using RPi
9. Smart home - complete home automation (IoT)
10. Smart garden or agriculture field (Irrigation based on weather and moisture content in soil - to redu ce water consumption) and Data analysis
11. Interfacing Raspberry pi with Twitter (IoT)
12. Measuring the water consumption of every tap (using Infra-red controlled taps) this is a very basic need as water shortage is a big problem all over the world.
13. Power control of home usi ng Motion sensor (switch off and on lights in the whole house)
14. Building an Oscilloscope with RPi
15. Building a solar cell tracker so that solar panels are automatically adjusted to get maximum sun light
16. Embedded Web Server using RPi for monitoring (home and industrial use) IoT use case
17. GNU Radio on Raspberry Pi
18. Simple ECG monitor using RPi and Heart data accusation and upload to cloud for analysis (IoT category)
19. A Robot which can climb vertical pipes (lamp post etc) coconut trees, and carry a surveillance wireless camera with it and provide picture/video from a height and comedown when needed

ARM Training - II (on Blueboard-LPC214x, LPC1114, LPC1343)

Course Duration – 5/7 days

- **ARM Introduction**
- **LPC2148 Block Diagram, Pin Description**
- **BlueBoard LPC214x Schematic Reading**
- **2 Line Character LCD Programming**
- **I2C Programming (EEPROM Interfacing)**
- **RTC Programming (Normal, Alarm Mode)**
- **GPIO Programming (Led, Buzzer, Switches Practical)**
- **SPI Programming (Nokia LCD Interfacing, SPI Ethernet Module Interfacing)**
- **Interrupt Programming (FIQ, IRQ, Vectored Interrupt, Non-Vectored Interrupts)**
- **ADC Programming (Interfacing Sensor to input of ADC and Application programming)**
- **UART programming (Polling Mode, Interrupt Mode) Practical Command Line Inter face using Serial communication. Implementation of debug function / printf function.**
- **ARM Cortex Programming using Industry Standard prototype board LPC Xpresso.**
- **Timers & Counter (Match Mode, Capture Mode, Timer Interrupts, Delays)**
- **GSM, GPS, Blueetooth, Zigbee**
- **PWM Programming (Motor driving program)**
- **RFID, Smart Card, Ethernet Module**
- **Projects: 2**

USB firmware and application development on ARM

Duration : 2 Days

This hands-on training provides complete knowledge required to design your Embedded USB Applications. The training is designed in such a way that it fulfils the industrial development needs.

Who Can Attend ?

Any one with good C programming skills and basic knowledge of micro-controller / microprocessor.

Lab Environment :

The training will be on Blueboard LPC2148 ARM7 board and host development is on Win OS.

Day-1 Morning

Lec: USB Intro, Architecture, Protocol, Bus Topology, USB Transfer Types, USB Descriptors, USB Enumeration process.
Lab: Hands-On Descriptors & Packet flow
Lec: USB Stack components and modifications, USB Classes

Day-1 Afternoon

Lab: USB Demo1 firmware implementation for Target
Lab: USB Demo1 Host Application implementation in VB.NET

Day-2 Morning

Lab: USB Demo2 firmware implementation
Lab: USB Demo2 Host Application

Day-2 Afternoon

Lec: Case study [Designing USB Bootloader]
Lab: Running USB Bootloader, Virtual Com-port example

University IoT Lab with IOT training

Overview

This training covers complete IoT system working starting from data collection from Sensor Node to the Cloud. Below is the brief how IoT System works

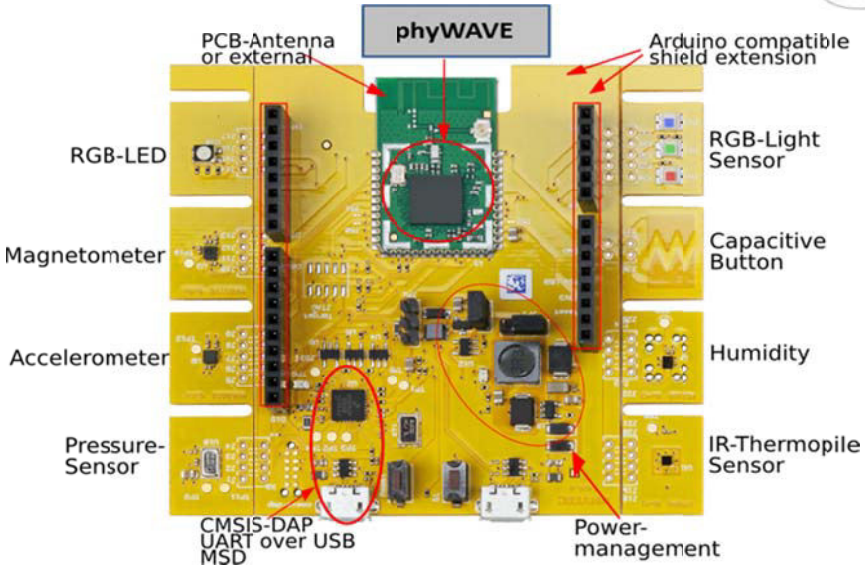
How IoT Works ?

1. phyNode eval board which includes phyWAVE - CC2650 & multiple sensors runs one of the RTOS (TI-RTOS / Contiki) having BLE/6LoWPAN stack can send the data to phyGATE or any BLE enabled Mobile Device.

2. phyGATE can connect to multiple phyNodes and locally display/store the data received, send commands to control & configure the phyNodes. phyGATE can connect to Cloud Services and push the data to cloud using multiple protocols like MQTT, CoAP, RESTful HTTP, XMPP, LWM2M.

3. Multiple Cloud Services (IBM Bluemix, Eclipse MQTT, Amazon, Microsoft Azure) can be easily integrated with phyGATE and store the device data in the cloud.

4. Easy User Interface can be developed as a part of Cloud Services or Stand-alone web application, Mobile Application for Android & iOS. For HMI Devices phyBoard-WEGA / phyBoard-Subra running Linux Qt / Android can be used



University IoT Lab Kit training Schedule

1. Internet of Things

- Introduction - The Internet of Things Vision
- IoT Network architecture

2. Primer on Wireless Network Protocols

2.1 Introduction to Wireless protocols

- BLE
- 6LoWPAN
- Zigbee
- Wifi
- Z-Wave
- Sub-1 GHz
- Thread
- LoRa
- SIGFOX

2.2 Bluetooth Low Energy

- Introduction to BLE
- Protocol Basics
- GAP(Advertising and Connections)
- GATT(Services and Characterstics)
- Topology
- BLE stack analysis

2.3 6LoWPAN

- Introduction to 6LowPAN
- Protocol Basics
- Topology
- Ipv6 architecture
- 6LoWPAN layers
- Routers
- Wireless Sensor Network

3. IoT Node: phyWAVE-CC2650

- Introduction to phyWAVE
- CC2650 SOC architecture
- On chip RADIO analysis

4. IoT Gateway: phyGATE-AM335x

- Introduction to phyGATE
- AM335x SOC architecture

5. Software Development for Node & Gateway

5.1 phyWAVE-CC2650

- Code composer Studio
- TI-RTOS basics
- BLE on SYS/BIOS (TIRTOS)
- Creating custom services & characteristics
- sensor drivers (i2c protocol)

5.2 phyGATE-AM335x

- hcitools (scan, discover services & characteristics)
- MQTT (IoT protcol to push data to the internet)
- pyton scripting (Implimented for phyNODE sensor data and MQTT)
- Cloud Server Connection.

5.3 Android

- phytec-iot app
- phyMQTT.

5.4 Contiki (RTOS)

5.5 Serial Line Internet Protocol (SLIP)

6. Programming IOT kit with sensor modules.

Active Buzzer Module ,
Automatic Flashing Colorful LED Module,
Class magnetic Sensor,
Heartbeat Module,
Flame Sensor Module,
Hall Magnetic Sensor Module,
Hit Sensor Module,
Hunt Sensor Module,
Infrared Emission Sensor Module,
Infrared Sensor Receiver Module,
Key Switch Module, Laser Sensor Module,
Linear Magnetic Hall Sensors,
Magic light Cup Module,
Mercury Open Optical Module,
Metal Touch Sensor Module,
Microphone Sound Sensor Module,
Mini Magnetic Reed Modules,
Obstacle Avoidance Sensor Module,
Optical Broken Module,
Passive buzzer Module,
Photo Resistor Module,
Reed Module,
Rotary Encoder Module,
Sensitive Microphone Sensor Module,
Temperature / Humidity Sensor Module,
Temperature Sensor Module,
Tilt Switch Module,
Vibration Switch Module,
XY-Axis Joystick Module,
2-Color LED Module,
5V relay module

Embedded Linux Training

Duration – 60hrs

Linux Internals (Module-1)

Introduction to Linux

- GNU Project / GPL Licensing
- Evolution of Linux & Development Model
- Device Identities in Linux
- Partitioning Schema

Introduction to Kernel

- History of Linux
- Types of Kernel
- The Linux kernel
- Kernel Architecture

Shell Commands & Shell Scripting

- Basic Shell commands
- Bash Shell Essentials
- Creating Makefiles

Creating Libraries

- Creating Static Library
- Creating Shared Library

The Boot Process

- BIOS Level
- Boot Loader
- Setup, startup_32 functions
- The start_kernel() function

The File System

- Virtual File system & its role
- Files associated with a process - System Calls

Process management

- Process Defined
- Process Descriptor Structures in the kernel
- Process States
- Process Scheduling
- Process Creation
- System calls related to process management

Memory Management

- Defining and Creating secondary memory areas
- Memory allocation & deallocation system calls ma lloc, calloc, alloca, free
- Demand Paging defined
- Process Organization in Memory
- Address Translation and page fault handling
- Virtual Memory Management Multi Thread Programming
- Creating multiple threads
- Parent synchronization with other Threads

Inter Process Communication

- Pipes, Fifo's, signals
- System-V IPC's
- Message queues - Shared memory - Semaphores

Sockets

- An Overview
- System calls related to TCP and UDP sockets

Network Programming

- TCP Server Client Programming
- UDP Server Client Programming
- Netlink socket interface

Programming & Debugging Tools

- strace : Tracing System calls
- ltrace : Tracing Library calls
- Tools used to detect memory access error and Memory leakage in Linux : mtrace
- Using gdb and ddd utilities
- Core Dump Analysis etc

Building Embedded Linux System (Module-2)

Lecture: Introduction to Embedded Linux and Blueboard-AT91RM9200

- Toolchain Components
- Building Toolchain
- Build Systems for compiling toolchain

Lab: Toolchain compilation and usage.

Lecture :

- Bootloader Architecture
- U-Boot Bootloader Porting on New Hardware.
- U-Boot Commands

Lab: Bootloader compilation and downloading on Target board.

- Bootloader commands and usage,
- Bootloader code customization,
- Adding new Ethernet drivers to U-Boot.

Lab: Downloading pre-compiled Linux kernel images on Target board.

- Using SD-Card for rootfs.
- Configuring NFS and using rootfs over NFS.
- Configuring TFTP a nd downloading kernel image over TFTP.

Building Embedded Linux System-II

Lecture:

- Linux Kernel Architecture.
- Linux Source code browsing & code ch anges for a new Target[Porting]

Lab: Configuring and compiling Linux Kernel.

Lecture: Root file system.

Lab:

- Building Root file system w ith Busybox and bo oting the Linux Kernel
- Cross compilation of libraries for target.
- Application development and Cross compilation.

Linux Device Drivers - (Module-3)

An introduction to device drivers

- Role of the Device Drivers
- Splitting the kernel
- Classes of devices and modules
- Kernel Architecture or Model
- kernel modules

Module Basics

- Introduction to Modules & Device Drivers
- Modules Defined
- Types of Modules in the kernel
- Writing Your first kernel module
- Module Related Commands
- Kernel Module vs Applications
- User space vs Kernel space
- Statically linked vs Dynamically linked drivers/modules
- Exporting symbols from modules
- Concurrency in the kernel
- Module Parameters
- Version dependency
- Kernel Module Programming - Lab exercises

The proc file system

- Creating proc file system entries
- Making read & write operations on proc entries
- Lab exercises

Character Device Drivers

- Registering a character device driver
- File operations
- The file structure
- devfs / lseek /ioctl
- Blocking, non bloc king and asynchronous operations - Programming with ioctl( ), mmap()
- Lab exercises

Linux Device Drivers –II

Hardware and Interrupt Handling

- Using IO Ports
- Installing and implementing an interrupt handler
- Tasklets and Bottom halves
- Task queues,
- Work queues
- Lab exercises

Block Device Drivers

- Block drivers structures
- Flash memory Drivers
- Lab exercises

Network Drivers

- The net_device structure in detail
- Making changes in Ethernet drivers in kernel Source
- Lab exercises

Building Embedded Linux Systems

Duration 2 Days

Day-1

Lecture :

- Introduction to Embedded Linux and ARM Cortex-A8 Hardware
- Toolchain Components
- Building Toolchain
- Build Systems for compiling toolchain

Lab: Toolchain compilation and usage.

Lecture:

- Bootloader Architecture
- U-Boot Bootloader Porting on New Hardware
- U-Boot Commands

Lab:

- Bootloader compilation and downloading on Target board
- Bootloader commands and usage,
- Bootloader code customization,
- adding new Ethernet drivers to U-Boot
- Downloading pre-compiled Linux kernel images on Target board.
    - Using SD-Card for rootfs
    - Configuring NFS and using rootfs over NFS
    - Configuring TFTP and downloading kernel image over TFTP

Day-2

Lecture:

- Linux Kernel Architecture
- Linux Source code browsing & code changes for a new Target[Porting]
- Root file system

Lab:

- Configuring and compiling Linux Kernel.
- Building Root file system with Busybox and booting the Linux Kernel
- Cross compilation of libraries for target.
- Application development and Cross compilation

VLSI & Embedded Systems Training Overview

Duration (15 Days)

Aspects Covered

- Verilog & System Verilog languages for design RTL coding & verification
- Introduction to ARM boards, implementing real life applications using ARM boards
- Introduction to FPGA Spartan boards, implementing designs on FPGA boards
- Introduction to Embedded Systems development and testing
- Installation of relevant tool licences in college labs
- Live demonstration of all projects, dedicated student lab sessions
- Support after workshop completion

Detailed Course Structure

Verilog for Design & Verification (14 hours)

- **Verilog Language constructs**
  - All languages constructs with detailed examples
- **Verilog for design implementation**
  - Synchronous & asynchronous FIFO
  - Traffic light controller
  - Pattern detectors
  - Timers
  - Memory controller
- **Verilog for functional verification**
  - Verification of above coded design using Verilog
- **Verilog LAB sessions**
  - Tool installed on college systems, students working on projects live

SystemVerilog for Functional Verification (30 hours)

- **SystemVerilog Language constructs**
  - All languages constructs with detailed examples
- **AXI & AHB Protocol**
  - Detailed understanding of AXI & AHB protocols
  - Verification IP development for AXI Protocol
- **Memory Controller Functional Verification**

- Reading specification, listing down features
- Testplan development
- Testbench Architecture definition & coding
- Sanity testcase coding & debug
- Complete testsuite development
- Setting up regression
- Verification closure using functional & code coverage, regression

Introduction to ARM Boards (6 hours)

- ARM Panda board overview
- Demonstration of real life projects using Panda board

Introduction to Spartan FPGA Boards (6 hours)

- Spartan FPGA Board overview
- Implementing designs on FPGA Boards

Introduction to Embedded Systems (12 hours)

- Embedded Systems development concepts
- Embedded Systems verification concepts

Android System Development Workshop

Duration : 2 Days

To be abreast with growing technology, learning the new technology is th e primary step. To help a t wo days workshop is offered to pro vide th e overview of Android and explore its building blocks. The training acquaints primary elements to start working on Android with a hands-on experience . The targeted audience should have prior knowledge in Linux Basics and good C, C++ or Java Programming skills.

What all things I know after this workshop ?

Big picture of Android. Layers & Modules of Android Framework. Setup the work environment for Android. Executing Sample Java Ap plications on real device. Android Runtime flow. Writing & Executing Native C/ C++ Programs on real device. Host and Device communication using Android ADB. Compiling Android for a Hardware and Bring-up different targets.

What next to this workshop ?

After this workshop you are ready to experiment on dif ferent layers o f Android as per your domain.

Further to this workshop we also offer advance workshops on

1. **" Android Application Development " (Java Programmers)**
2. **" Android Advance Programming" (Java Programmers)**
3. **" And roid Porting, BSP, Device Drivers & Native Framework Development" (C \ C++ & Linux programmers)**

Day-1

Android Anatomy

- Introduction & history of Android
- Native Libraries
- Android Runtime
- Application Framework

Android Kernel

- Introduction
- Binder
- Power
- Ashmem
- Low Memory Killer
- Logger, ADB
- Miscellaneous Patches

Getting Started

- Setting up development Machine
- Packages required on development machine
- Hands On (Demonstration)

Repo & Git - Overview

- Manifest file
- Working with repo and git
- How to Download Build & Compiling

Android Emulator on Windows and Linux Host

- Introduction
- Goldfish & QEMU
- Working with emulator
- Connecting to emulator over ADB
- Hands on with Eclipse SDK emulator (Demonstration)

Android Runtime

- Introduction to Dalvik/Zygote
- How Android framework starts
- app_runtime -> zygote
- System server, Android services
- Instances of Dalvik
- Hardware abstraction layer

Day 2

Android Toolchain

- Introduction
- Features of Android Toolchain
- Dependency of libc with android build

Android Native Layer Development

- Overview of Android C/C++ Libraries
- Modifying C/C++ Libraries
- Porting New C/C++ Libraries
- Cross Compiling a C Program and executing on Target.

Preparing and Porting Android for ARM Board

- Download Android kernel
- Compile Android kernel
- Prepare Android filesystems
- Porting applications on ARM Board

Android Development Program – Android for Beginners

This training aims at providing in depth knowledge on android programming practices being used in the Industry where engineers are working currently. The training focuses on Android in particular.

Pre-requisites: Understanding of OOPS, Core java for beginners

Software Tools required: Eclipse, ADT plugin for eclipse, Android SDK fo r windows/mac

COURSE CONTENT :

Introduction to Android

- What is Android?
- Setting up development environment
- Dalvik Virtual Machine & .apk file extension
- Fundamentals:
  - Basic Building blocks – Activities, Services, Broadcast Receivers & Content providers
  - UI Components – Views & notifications
  - Components for communication -Intents & Intent Filters
  - Android API levels (versions & version names)
  - Application Structure (in detail)
  - AndroidManifest.xml
  - uses-permission & uses-sdk
  - Resources & R.java
  - Assets
  - Layouts & Drawable Resources
  - Activities and Activity lifecycle
  - First sample Application

Emulator-Android Virtual Device

- Launching emulator
- Editing emulator settings
- Emulator shortcuts
- Logcat usage
- Introduction tDDMS
- Second App:- (switching between activities)
  - Develop an app for demonstrating the communication between Intents

Basic UI design

- Form widgets
- Text Fields
- Layouts
- [dip, dp, sip, sp] versus px

Preferences

- SharedPreferences
- Preferences from xml
- Examples

Menu

- Option menu
- Context menu
- Sub menu
- menu from xml
- menu via code

MATLAB Courses

Matlab® fundamentals and Programming Techniques

This course mainly deals with MATLAB ® programming techniques. MATLAB® is a programming environment for algorithm development, data analysis, visualization, and numerical computation. Using MATLAB(R), you can solve technical computing problems faster than with traditional programming languages, such as C, C++, and FORTRAN.

MATLAB Product Description

- Key features
- Architecture

MATLAB Software

- Introduction to MATLAB(R) Software
- MATLAB(R) windows
- Command Window
- Editor Window
- Workspace
- Command History
- Current directory

MATLAB(R) Data Types

- Data types
  - Numeric
  - String
  - Data type conversion
    - Numeric to String
    - String to Numeric

Operators & Special characters

- Arithmetic operators
- Bit-Wise Operators
- Relational Operators
- Logical Operators

Complex Numbers & Trigonometric functions

- To work with complex numbers and trigonometric functions in MATLAB(R)

Matrices and Arrays

- Array Initializations
- About Matrices
- Generating Matrices
- Matrix Sum, transpose, diagonal, inverse
- Matrix Multiplication, division
- The magic Function
- Matrix and Array Operations
- Matrices and Magic Squares

Types of Arrays

- Multidimensional Arrays
- Structures
- Cell Arrays

Loops and Conditional Statements

- Control Flow
- Conditional Control – if, else, switch
- Loop Control – for, while, continue, break
- Program Termination – return

Functions

- Writing user defined functions
- Function calling
- Return Value
- Types of Functions
- Global Variables

Plots

- Plotting vector and matrix data
- Plot labelling, curve labelling, legend and colour bar editing
- Plot types

- ***2-D Plots***

- Basic Plotting Functions
- Creating a Plot
- Plotting Multiple Data Sets in One Graph
- Specifying Line Styles and Colors
- Graphing Imaginary and Complex Data
- Figure Windows
- Displaying Multiple Plots in One Figure
- Controlling the Axes

- ***3-D Plots***

- Creating Mesh and Surface
- About Mesh and Surface Visualizing
- Subplots

M-files

- The MATLAB(R) Editor
- Script M-files
- The MATLAB(R) path
- Function M-files
- Sub-functions and nested functions
- Debugging
- Best script file writing tactics

Visualizing the different applications in MATLAB(R)

- Statistical parameter estimations
- DSP applications
- Image Processing applications
- Control System applications

MATLAB Courses

Image Processing with Matlab

This course deals with using MATLAB® Image Processing toolbox for image processing, analysis, visualization, and algorithm development. The training covers various topics such as importing and exporting images, pre- and post-processing of image s, analysis and vi sualization of images, and spatial transformations and image registration.

COURSE CONTENT :

Introduction

- A quick overview of MATLAB(R) computing environment
- Overview of MATLAB(R) Image Processing toolbox
- Course content and material discussion

Acquiring and handling images in MATLAB

- Connecting the hardware
- Retrieving hardware information
- Acquiring and viewing the image data
- Image file I/O
- Exploring image types (RGB, binary, intensity, and
- indexed images)
- Image type conversions
- The concept of color space and image color space conversions
- Finding pixel value information
- Computing mean and standard deviation of images
- Measuring properties of image regions

Image enhancement techniques

- Adjusting image intensity
- Image histogram operations: adjustment, equalization,and stretching
- Multidimensional arrays
- Image arithmetic operations
- Cropping and resizing images
- Image alignment correction: rotating images

Image filtering

- Neighborhood and block processing of images
- Distinct block operations
- Sliding neighborhood operations
- Performing image convolution and correlation
- Averaging filters
- Region of interest processing
- Introduction to spatial and frequency domain filtering

Image restoration techniques

- Reducing noise from images
- De-blurring images
- Correcting background illumination

Edge detection related operations

- Edges in an image
- Detecting edges with various methods: Sobel, Prewitt,
- Roberts, Laplacian of Gaussian, zero cross and Canny.
- Computing edge directive histogram

Image morphological operations

- Bridging unconnected pixels, cleaning, closing, and opening
- Dilation and erosion
- Identifying and labeling connected components

Image transforms

- Forward and inverse Discrete cosine transform
- Forward and inverse Fast Fourier transform
- Forward and inverse Radon transform
- Applying wavelet transform to images

MATLAB Courses

Modelling & Simulation for Automotive and Aerospace Applications

Mathematical Modeling or Model Based Design (MBD) – The course deals with how to develop mathematical model from a Physical system. This involves, learning control system concepts, mathematical background to understand applications. For Ex: How to optimize the performance of suspension system. In this case, we need to develop a mathematical model of a Suspension system by considering the d ifferent forces acting on t he system when it hits the road humps or path holes. Using the kn owledge of control system, kinematics & mathematics we develop a model. Similarly, we take different cases from electrical, mechanical, automotive, aerospace domain.

Once the model i s developed, we n eed to t ransfer them into software program. We make use of the Matlab®, Simulink®, Stateflow®, RTW platform, LabView to realize them in the software. Based on the response, we fine tune in the software. It involves, programming on the above said platform.

This course is intended to provide training on Matlab® Simulink; and extension of MATLAB® computing environment for modeling, simulating, and analysing dynamic and linear / nonlinear systems. (Customer specific applications will be discussed).

COURSE CONTENTS

Introduction

- A quick overview of MATLAB® computing environment
- Overview of Simulink: A tool for simulation and Model-based design
- Course content and material discussion
- Understanding the architecture of the Software
- Programming on Matlab®
- Using the control system toolbox
- Translating the physical applications into Matlab® environment
- Optimizing the performance of the Models

Building your first Simulink model

- Modeling example
- Creating an empty model
- Browsing the Simulink block library
- Adding blocks
- Connecting the blocks
- Configuring the model
- Setting simulation preferences
- Running the model
- Visualizing and retrieving simulation results
- Generating a model report
- Annotating diagrams
- Interactive demonstrations

Developing hierarchical Models

- Creating subsystems
- Navigating the subsystem models
- Controlling access to subsystems
- Creating conditionally executed subsystems
- Enabled subsystem
- Triggered subsystem
- Function call subsystem

- Referencing model

Simulink debugger

- Introduction
- Using the debugger GUI
- Using the debugger command
- Starting the debugger
- Starting a simulation in debug mode
- Running a model step-by-step
- Setting break points

Control System Tool Box

- Exploring the functions in control system toolbox
- Developing open loop control systems
- Developing closed loop control systems
- Optimizing the performance with controller

State flow

- Concept of State flow
- State flow and Simulink

MATLAB Courses

Advanced Digital Signal Processing with Matlab

This course mainly deals with using MATLABSignal Processing toolbox for Digital signal processing, analysis, visualization, and algorithm development. The training covers various topics such as filter design, windowing techniques, transforms, multi-rate signal processing, statistical signal processing, parametric modelling etc.

COURSE CONTENT :

Introduction to DSP

- Introduction to DSP
- Sampled data systems
- Aliasing and antialiasing
- Reconstruction
- Practical limitations
- Frequency & amplitude resolution
- Quantization and timing errors
- Correlation and convolution
- Frequency analysis
- Fourier transforms
- Frequency 'leakage'
- Windowing
- Multi-rate signal processing

Transforms

- Fourier Transform• Z – Transform• DCT Transform
- Hilbert Transform
- Wavelet Transform

Filters

- FIR Filter – FIR digital Filters
  - FIR filter basics
  - Analysis of FIR filters
  - Frequency & impulse responses
  - The window design method
  - Optimization design methods
  - Practical limitations of FIR filters
- IIR Filter –
  - IIR filter basics
  - Analysis of FIR filters
  - Frequency & impulse responses
  - IIR filter design
  - Poles, zeroes and filter response

Cepstral analysis

- Complex Cepsturm
- Inverse complex cepstrum
- Real cepstrum and minimum phase reconstruction

Statistical signal processing

- Introduction to statistical parameters
- Autocorrelation matrix
- Power spectral density (PSD)
- Cross power spectral density
- Finding PSD using various Methods (periodogram, modified periodogram, covariance, Eigen vector, burg, yule walker,

	Welch, MUSIC Algorithm, Root MUSIC Algorithm) ▪ Spectrogram ▪ Transfer function estimation
Parametric modelling	▪ Introduction to signal modelling ▪ Study of Auto Regressive Moving Average Models (ARMA), ARModels and MA models ▪ Estimation of model parameters using various methods like Yule-Walker, prony etc)
DSP with MATLAB(R)	▪ Introduction to DSP Toolbox ▪ Signal processing functions in MATLAB(R) (conv, conv2, corrcoef,cov, cplxpair, deconv, fft, fft2, fftshift, filter2, freqspace, ifft, ifft2,unwrap) ▪ Time domain analysis of a signal ▪ Frequency domain analysis of a signal
Digital Filter Design in MATLAB(R)	▪ Discrete-Time Filters (Direct form I, Direct form II, lattice filters) ▪ 1_D Median filtering ▪ Butterworth filter design ▪ Chebyshev Type I filter design (pass band ripple) ▪ Chebyshev Type II filter design (stop band ripple) ▪ Raised cosine FIR filter design ▪ Recursive digital filter design
Window Design	▪ Rectangular window ▪ Hamming window ▪ Hanning window ▪ Bartlett window ▪ Kaiser window etc
Transforms	▪ Discrete fourier transform ▪ Discrete cosine transform ▪ Hilbert transform ▪ Discrete wavelet transform ▪ inverse transforms
Multi-rate Signal Processing	▪ Decimation ▪ Interpolation ▪ Up-Sampling ▪ Down-Sampling ▪ Re-Sampling
Linear Systems	▪ Stabilize polynomial ▪ *z*-transform partial-fraction expansion ▪ conversion of digital filter parameters to transfer function form/ pole-zero form etc
Cepstral analysis	▪ Complex cepstral analysis ▪ Inverse complex cepstrum ▪ Real cepstrum and minimum phase reconstruction
Statistical signal processing	▪ Cross Correlation ▪ Covariance ▪ Data matrix for autocorrelation matrix estimation ▪ Power spectral density (PSD)

- Cross power spectral density
- Finding PSD using various Methods (periodogram, modified periodogram, covariance, Eigen vector, burg, yule walker, Welch, MUSIC Algorithm, Root MUSIC Algorithm)
- Spectrogram
- Transfer function estimation

Parametric Modelling

- Autoregressive (AR) all-pole model parameters estimated usingBurg method
- Estimate AR model parameters using covariance method
- Estimate AR model parameters using modified covariance method
- Estimate autoregressive (AR) all-pole model using Yule-Walker method
- Cross power spectral density
- Prony method for filter design

Waveform Generation

- Swept-frequency cosine
- periodic sinc function
- Pulse train
- Saw-tooth or triangle wave

GUI's

- Filter Design and Analysis Tool
- GUI-based filter design
- Open interactive digital signal processing tool
- Open Filter Visualization Tool

Bi-level Waveform Measurements

- Duty cycle of pulse waveform
- Fall time of negative going bi-level waveform transitions
- Period of bi-level pulse
- Separation between bilevel waveform pulses
- Bilevel waveform pulse width
- Slew rate of bilevel waveform

MATLAB Courses

Biomedical Signal Processing with Matlab

This training is about using MATLAB Signal Processing toolbox for Bio-Medical signal processing, analysis, visualization, and algorithm development. The training covers various topics such as filter design, windowing techniques, transforms, multi-rate signal processing with respect to Bio-Medical Signals etc.

COURSE CONTENT :

Introduction to DSP

- Introduction to DSP• Sampled data systems
- Aliasing and antialiasing
- Reconstruction
- Practical limitations
- Frequency & amplitude resolution
- Quantization and timing errors
- Correlation and convolution
- Frequency analysis
- Fourier transforms
- Frequency 'leakage'
- Windowing
- Multi-rate signal processing

Transforms

- Fourier Transform• Z – Transform
- DCT Transform
- Wavelet Transform

Filters

- FIR Filter – FIR digital Filters• FIR filter basics
  - Analysis of FIR filters
  - Frequency & impulse responses
  - The window design method
  - Optimization design methods
  - Practical limitations of FIR filters
- IIR Filter –
  - IIR filter basics
  - Analysis of FIR filters
  - Frequency & impulse responses
  - IIR filter design
- Poles, zeroes and filter response

DSP with MATLAB(R)

- Introduction to DSP Toolbox
- Signal processing functions in MATLAB(R) (conv, conv2, corrcoef, cov, cplxpair, deconv, fft, fft2, fftshift, filter2, freqspace, ifft, ifft2,unwrap)
- Time domain analysis of a signal
- Frequency domain analysis of a signal

Digital Filter Design in MATLAB(R)

- Discrete-Time Filters (Direct form I, Direct form II, latticefilters)•
- 1_D Median filtering
- Butterworth filter design
- Chebyshev Type I filter design (pass band ripple)
- Chebyshev Type II filter design (stop band ripple)
- Raised cosine FIR filter design
- Recursive digital filter design

Window Design	▪ Rectangular window• Hamming window ▪ Hanning window ▪ Bartlett window ▪ Kaiser window etc
Biomedical signal processing	▪ Introduction to bio-medical signals (ECG, EMG, EEG)• Signal analysis ▪ Biomedical signal processing ▪ Signal enhancement ▪ Artifact removal
Case Studies	▪ ECG Analysis• EMG Analysis ▪ EEG Analysis ▪ Brain Computer Interface Demo

MATLAB Courses

Biomedical Image Processing with Matlab

This training is all about how MATLAB Image Processing toolbox can be used for Bio-Medical image processing, analysis, visualization, and algorithm development. The training covers various topics such as importing and exporting images, pre and post-processing of images, analysis and visualization of images, and spatial transformations and image registration.

Introduction	▪ A quick overview of MATLAB(R) computing environment ▪ Overview of MATLAB(R) Image Processing toolbox ▪ Course content and material discussion
Acquiring and handling images in MATLAB(R)	▪ Image file I/O ▪ Exploring image types (RGB, binary, intensity, andindexed images) ▪ Image type conversions ▪ The concept of color space and image color space conversions ▪ Finding pixel value information ▪ Computing mean and standard deviation of images ▪ Measuring properties of image regions
Image enhancement techniques	▪ Adjusting image intensity ▪ Image histogram operations: adjustment, equalization,and stretching ▪ Multidimensional arrays ▪ Image arithmetic operations ▪ Cropping and resizing images ▪ Image alignment correction: rotating images
Image filtering	▪ Neighborhood and block processing of images ▪ Distinct block operations ▪ Sliding neighborhood operations ▪ Performing image convolution and correlation ▪ Averaging filters ▪ Region of interest processing ▪ Introduction to spatial and frequency domain filtering
Image restoration techniques	▪ Reducing noise from images ▪ De-blurring images ▪ Correcting background illumination
Edge detection related operations	▪ Edges in an image ▪ Detecting edges with various methods: Sobel, Prewitt,Roberts, Laplacian of Gaussian, zero cross and Canny. ▪ Computing edge directive histogram
Image morphological	▪ Bridging unconnected pixels, cleaning, closing, andopening

operations	▪ Dilation and erosion ▪ Identifying and labeling connected components
Image transforms	▪ Forward and inverse Discrete cosine transform ▪ Forward and inverse Fast Fourier transform ▪ Forward and inverse Radon transform ▪ Applying wavelet transform to images
Bio-Medical Image Processing	▪ Introduction to Bio-Medical Image Processing ▪ Overview of different imaging modalities ▪ Medical Image Enhancement ▪ Medical Image filtering ▪ Medical Image segmentation

Digital Signal processing & Image processing using SCI Lab

Scilab is an open source, cross-platform numerical computational package and a high-level, numerically oriented programming language. As the syntax of Scilab is similar to MATLAB, Scilab includes a source code translator for assisting the conversion of code from MAT LAB to Scilab. Scilab is available free of cost under an open source license and is one of several open source alternatives to MATLAB(R). Scilab has been widely exploited for different applications in signal processing, statistical analysis, image processing, fluid dynamics simulations, numerical optimization, and modeling, simulation of explicit and implicit dynamical systems and symbolic manipulations. This training workshop is intended to provide basic understanding about the Scilab platform and to exploit its integration in the field of signal and image processing.

COURSE CONTENT :

Introduction to Scilab	• Why Scilab? • Pros & Cons • Software Architecture • Variables and datatypes
Handling Arrays & Matrix	• Basic Matrix operations • Indexing • Using Built in functions
Flow Control	• Loops- for, while, do • Conditions- If else, select case
Handling files	• Basic input-output functions
Functions	• How to create user defined functions • Passing and returning multiple arguments
Plotting options	• Simple plots – 1D,2D, 3D
Basic Math operation	• Solving Equations • Simple Examples
Introduction to Matlab®	• Architecture • Matrix operation, Flow control, Functions, • Structures, cells, Plots
Interfacing Matlab® and Scilab	• M to S conversion
Basics of Signal	• Basic DSP principles

Processing	• Signals, system, convolution, correlation, digital filters, transforms, modulation
Scilab for 1D Signal Processing	• Simple Scilab examples
Basics of Image Processing	• Digital Image, image enhancement, mage segmentation, filtering • IP in scilab examples
Scilab for control applications	• Simple control/ network programs in Scilab

Image processing using openCV and Phyton

This training program / course mainly deals with Implementation of Image Processing algorithms using Python Scripting on OpenCV platform.

Introduction

OpenCV is a library of cross platform programming functions aimed at real time Computer Vision. IT was designed for computational efficiency and with a strong focus on real-time applications, video and image processing. Python is a widely used gen eral-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to exp ress concepts in fewer lines of code than would be possible in languages such as C. The language provides constructs intended to enable clear programs on both a small and large scale. The growing demand of integrating OpenCV with python promises clear cut solutions to image processing problems. Since the tools are open source, researchers can exploit the freedom and possibilities of expansion. Wide spread applications in the field of robotics underlines the scope of OpenCV for image processing.

Course content :

Python Training

- History & Overview
- Installation & Getting Started
- Basic Syntax and Operators
- Decision Making
- Strings, Lists, Tuples, Dictionary
- Mutability
- Loops
- User defined Functions
- File IO

Getting Started with Python-Opencv

1. **Introduction**

- About OpenCV
- Installation
- Opencv & Python Integration
- About Images

2. **Basic operations on Images**

- Read & Writing an Image
- Access pixel properties, values & modifying
- Splitting & Merging of image channels
- Arithmetic Operation
- Bitwise Operation

3. **GUI Features**

Image Processing Module 1

- Display images in window
- Getting started with video capturing
- Drawing Functions like circle, line, rectangle, polyline
- Plotting functions

4. Changing Color Spaces

5. Geometric Transforms

- Scaling
- Translation
- Rotation of Image

6. Histograms

- About Histogram
- Histogram Calculation
- Histogram Equalization

Image Processing Module 2

7. Filters

- About Convolution
- Different types of filters like Averaging, Blurring, Gaussian and Median

8. Thresholding

- About Thresholding
- Adaptive Thresholding

9. Edge Detection

- Different type of edge detection like Canny, Sobel and Laplacian edge detectors.

Image Processing Module 3

10. Morphological Operation

- Erosion, Dilation, Opening and Closing

11. Image Transformation

- Discrete Fourier Transform and Inverse Fourier Transform

12. Python-OpenCV sample projects

Texas Instruments DSP Processors 6713/ 6416 CCS

This course mainly deals with programming on TMS320C6713/TMS320C6416 DSP Starter Kit (DSK) using CCS, which is a low-cost development platform designed to speed the development of high precision applications based on TI´s TMS320C6XXX floating point DSP generation.

COURSE CONTENT :

Introduction to MATLAB

- Quick overview on MATLABâ architecture and computing environment
- Data types and operators in MATLABâ
- Array and matrix operations
- Functions
- Structures
- Plots

Introduction to Digital Signal Processing

- Introduction to signals and systems
- Sampling and Quantization
- Overview of Digital Signal Processing
- Windowing Techniques
- Filtering

Introduction to DSK

- Why Special Purpose processor for DSP
- History of TMS Series
- What is Code Composure Studio
- Difference Between Floating and Fixed Point Processors
- An Introduction to TMS320C6713
- An Introduction to TMS320C6416

DSK(TMS320C6X) Architecture

- Von Neumann Architecture and Harvard architecture
- Concerns on Fixed Point Processors (Quantization Error)
- Functional Units
- Pipelining
- Registers
- Interrupts
- McBSP’s
- DMA

Hands on DSK

- Introduction to CCS
- Quick Test of DSK
- Difference between Compiler, Linker, Assembler
- Detailed Explanation of Support Files
- Building a small Project (hello world)
- Generation of Sinusoid using DIP switch and explanation of the program
- Illustration of Watch Window, GEL file
- Few Experiments on Sine generation program using DIP Switch

Plotting with CCS	▪ Generation of Sine and Plotting with CCS ▪ Usage of Circular Buffer ▪ Usage of Hardware Interrupt int_11
Profiling with CCS	▪ Dot Product of Two Arrays ▪ Implementing Variable watch ▪ Setting up Break Points ▪ Profiling Printf function
Real Time Implementation	▪ Input with Onboard AIC23 Stereo Codec ▪ TLV320AIC23 Onboard Stereo Codec ▪ Example Program to Illustrate onboard Stereo Codec using Hardware Interrupt and explanation of the program using McBSP's. ▪ Example Program to Illustrate onboard Stereo Codec using polling and explanation of the program using McBSP's ▪ Example program to illustrate Multi Channelled McBSP's. ▪ Examples Illustrating Echo and Delay ▪ Example illustrating sine generation using table created by MATLAB(R) ▪ Few assignments (Square generation and ramp generation using table created by MATLAB and CCS plotting
Real time Implementation	▪ Generation of amplitude Modulated signal using C6713 DSK ▪ Generation of Pseudorandom Noise using C6713 DSK ▪ Recording Voice using external Memory(SDRAM)
FIR filters	▪ Implementation using Pseudorandom Noise sequence as input to filter and output stored in memory ▪ Two Notch filters recovering the Corrupted Input ▪ Voice Scrambler using Filtering and Modulation ▪ Real Time Convolution ▪ FIR Implementation of LP HP BP BS Using DSK 6713 DSK ▪ FIR
IIR and Adaptive Filters	▪ IIR filtering using cascaded direct form -II ▪ Adaptive Filters ▪ What is Adaptive Filter ▪ Application of adaptive Filters ▪ Least Mean Square Algorithm and RMS ▪ Noise Cancellation , System Identification

Implementation of Adaptive Filters

- Implementing Adaptive Filter for sinusoidal Noise cancellation on DSK 6713
- Adaptive FIR Filter for System ID of a Fixed FIR as an Unknown System
- Adaptive FIR for System ID of a Fixed FIR as an Unknown System with Weights of an Adaptive Filter Initialized as an FIR Bandpass- Plotting with CCS
- Adaptive FIR for System ID of Fixed IIR as an Unknown System

DSP/BIOS

- What is DSP BIOS?
- Uses of DSP BIOS
- Sine Generation with DIP Switch Control Through DSP/BIOS
- Blinking of LEDs at Different Rates Using DSP/BIOS
- Sine Generation Using BIOS to Set Up Interrupt INT11

Arduino Uno Microcontroller Training

This course mainly deals with microntrollers and programming basics. Arduino is an open-source physical computing platform based on a simple I/O board and a development environment that implements the Processing / Wiring language. Arduino can be used to develop stand-alone interactive objects or can be connected to software on your computer. This is a hands-on training.

Course content:

Getting started with Arduino Uno	▪ Microntroller ▪ Development board ▪ Arduino uno product description
Arduino software	▪ Installing software ▪ The integrated development environment
Project 1 : Working with LED	▪ Blinking of led ▪ Multiple led circuit ▪ Led cube
Project 2 : Working with Potentiometers	▪ Input to arduino ▪ Pushbuttons ▪ Potentiometers
Project no 3: Generating different Sounds	▪ Sound ▪ Simple note ▪ Music
Project no 4: Distance Measurement	▪ Measuring distance and temperature
Project no 5: Interfacing with Displays	▪ LCD
Project no 6: Sensing X,Y & Z direction	▪ Sensors ▪ Reed switch(magnetic field detector)
Project no 7: RoboCar	▪ Controlling robot car motion using arduino.
Interfacing arduino uno to different software's	▪ Matlab ▪ Python ▪ OpenCV
Reading data from Matlab(R) using arduino and performing different robotic action	▪ Brain computer interface performing different Robot Actions. ▪ Reading Different inputs generated from Hand Gesture application into Arduino

Appcelerator – Hybrid Mobile Development training

Pre-requisites: Javascript

Software Tools required: Appcelerator studio, Android / iOS SDK, Xcode (only for mac users + iphone appps)

COURSE CONTENT :

JavaScript	▪ Understand JavaScript syntax and fundamentals ▪ Identify CommonJS coding patterns ▪ Instantiate Titanium objects ▪ Describe execution context and the means by which you create one or more ▪ Identify the benefits of a single-context app design ▪ Compare and contrast include() and require()
Cross-platform development	▪ Implement branching techniques using appropriate Titanium properties ▪ Include platform-specific resources at build time ▪ Include density and aspect-ratio specific images at build time ▪ Compare and contrast mobile platform features
Titanium basics	▪ Create a Titanium project ▪ Run a Titanium project in the simulator or emulator ▪ Configure app properties such as the SDK version, target platforms, etc. ▪ Describe the architecture of Titanium
User interface	▪ Select the appropriate UI measurement units ▪ Position elements on screen accounting for the UI coordinates system ▪ Select and implement layout modes ▪ Send and react to user and non-user events
App properties	▪ Store data in an application property ▪ Retrieve data from an application property ▪ Identify the data types and access methods supported by app properties ▪ Determine when app properties are the most suitable storage location for your app's data
Web content	▪ Include HTML/CSS content in your Titanium app ▪ Identify the ramifications and pitfalls of the WebView ▪ Communicate between the WebView and native Titanium environments
Networking	▪ Implement the HTTP Client object ▪ Retrieve data in various formats from network

	services ▪ Upload and download files across the network ▪ Post JSON-formatted data ▪ Determine when to use JSON, XML, and SOAP to transmit rich data across the network
Appcelerator Cloud Services	▪ Cloud-enable an app ▪ Manage app keys ▪ Implement ACS APIs ▪ Identify ACS APIs and features ▪ Describe ACS security features
Mapping	▪ Add a map to your app ▪ Set map options and properties ▪ Add annotations to your map ▪ Set annotation options and properties ▪ Enable event handling for maps and annotation

Agile / Scrum Testing Program

This training aims at provi ding in d epth knowledge, practical experience and roles plays on Agile and Scrum.

Agile software development is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery, a time- boxed iterative approach, and encourages rapid and flexible response to change. It is a conceptual framework that promotes foreseen tight iterations throughout the development cycle.

Five Reasons Why Agile Best Fits Today's Fast-Paced Organizations

- Project Priorities change: Agile organizations can more easily adapt their project portfolios due to chan ging business priorities. Agile projects deliver working software on a regular schedule, typically weekly or biweekly.

- Learning and responding to customer needs is critical: The most important lessons you will get about your software will come from customers. Agile teams are able to get software into their hands quickly by only building just enough software to get valuable features delivered.

- Product owners need to focus on the customer and market: The details of delivery bog down many product owners: spending time assigning work, checking progress and writing status reports. In contrast, agile teams strive to be "self-organized."

- Your Star player may be gone tomorrow: Teams change. Employees quit . New hires come on, and employees move between projects within and organization. Agile teams can handle changes to the team much more easily than non-agile teams.

- Stay competitive by discovering trouble spots: Every organization and process has problems. Finding these problems and fixing them is critical to staying competitive. Detecting and eliminating issues in the development process is a central feature of agile development.

- The training focuses on providing an inside view into Scrum a very popular agile software development methodology. During the course, the participants get opportunity to learn and experience Agile and Scrum from industry leaders in the agile community. The trainers are alumnus of IITs and prestigious institutes in United Kingdom and have worked for different fortune 500 companies in the past.

Agile / Scrum Testing Program course content :

Introduction to Agile	▪ Introduction to Agile ▪ The Traditional Approach ▪ Agile Software Development ▪ The Agile Manifesto
Foundations of Scrum	▪ Essence & Foundations ▪ Role Plays
Essential Scrum Activities	▪ Writing Product backlog items ▪ Sprint Planning Meeting ▪ Daily Scrum Meeting ▪ Definition of Done ▪ Backlog refinement meeting ▪ Sprint Review meeting ▪ Sprint Retrospective
Scrum Simulation	

SIEMENS TRAINING
TRAINING PROGRAMME ON BASIC AUTOMATION

Course Duration: 5 days
Course Contents :

1. Basic constituents of PLC:
Signal modules, CPU, Power Supply, mounting rail and MMC.

2. How PLC works?

3. Installation guidelines, powering and wiring of modules with information on addressing

4. Programming:

- Programming language and representation in STL, FBD and LAD
- Hardware Configuration and setting object Properties of Modules in STEP
- c. Programming instruction: AND, OR, AND-before-OR, OR-before - AND, NO / NC contacts, Edge detection instructions. Set / Reset, Elementary data type

5. Overview of SIMATIC S7 – PLC:

- Programming Units and using PC as Programming Unit
- Hardware Configuration and setting object Properties of Modules in STEP
- c. Step 7 Instructions and programming: Set / Reset, Elementary data type, Load / Tra nsfer, Comparison, basic math instructions.
- Timers / Counters List etc

6. Using Symbol Table and VAT.

7. STEP 7 blocks and structured programming

8. Using Data Blocks.

9. Use of Organisation Blocks.

10. Analog signal processing.

11. Introduction to HMI.

Prerequisite :

Diploma / Degree students in Electrical / Electronics / Instrumentation / ENTC / Biomedical / Mechanical Engineering.

SIEMENS TRAINING
TRAINING PROGRAMME ON BASIC PROCESS INSTRUMENTATION

Course Duration: 5 days full time

Course Contents:

1. **FLOW measurement**

   - Magnetic Flow meter technology
   - Mass Flow meter technology
   - Ultrasonic Flow meter technology (Inline & Clamp On)

2. **LEVEL measurement**

   - Ultarsonic Level Technology
   - Radar Level Technology
   - Capacitance Level Technology

3. **PRESSURE Measurement**

4. **TEMPERATURE Measurement**

5. **ELECTROPENUMATIC POSITONER**

Topics for all above Measurements.

- Various Technology
- Working Principles
- Product over view
- Live demo & Hands on tria l for Pr essure, Temperature & Positioners, etc.

Prerequisite:

Diploma / Degree students in Electrical / Electronics / Instrumentation / ENTC / Biomedical / Mechanical Engineering.

SIEMENS TRAINING
TRAINING PROGRAMME ON BASIC SCADA

Course Duration: 5 days full time

Course Contents:

- System overview of SIMATIC WinCC.
- Creating a project
- Configuring connections to the SIMATIC S7
- Graphics Designer and graphics displays for human machine interfacing
- Alarm logging for message representation, messag e archiving
- Tag logging for curve representation, measured value archiving
- User Administration
- Data archiving with the User Archives optio n (introduction)
- Report Designer for logging (introduction)
- Background processing (introduction of Global Scripts)
- Practical exercises

Prerequisite:

Diploma / Degree students in Electrical / Electronics / Instrumentation / ENTC /Biomedical/ Mechanical Engineering.

SIEMENS TRAINING
TRAINING PROGRAMME ON BASICS OF AC/DC DRIVES

Course Duration: 5 days full time

Course Contents:

- Brief Basic Power Electronics (including Thyristors, Power-Transistors & IGBTs)
- DC Motor Basics (construction, principle of operation, T-N Characteristics etc).
- DC Drives Basics (Block diagram, 1Q-4Q principle of operation, T-N Curves etc)
- Selections, Calculations & applications of typical DC drives.
- Siemens DC Drives (6RA80) - Ratings, Specs, features, options & applications.
- AC Motor Basics (construction, principle of ope ration, T-N Characteristic etc).
- AC Drives Basics (Block diagram, 1Q-4Q principle of operation, T-N Curves etc)
- Selections, Calculations & applications of typical AC drives.
- AC Drives (Sinamics S & G)-Ratings, Specs, features, options & applications.
- MEDIUM VOLTAGE (MV Drives & Motors):
- MV Motor types & Fundamentals (including starting methods, options/features)
- MV Motor offers from Germany (separately for Induction & Synchronous Motor)
- MV Converter Basics & types (Voltage, Current Source & Cyclo - converters)
- Siemens MV Converters (Sinamics GM, Simovert-S and Perfect Harmony)
- Selection, configuration & Applications of MV Drive systems
- Short briefing on MV Transformers along with their options & protections.

Prerequisite:

Diploma / Degree students in Electrical / Electronics / Instrumentation / ENTC / Biomedical / Mechanical Engineering

SIEMENS TRAINING

TRAINING PROGRAMME ON BASICS OF INDUCTION MOTOR

Course Duration: 3 days full time

Course Contents:

- Basics of Electricity.
- Motor- Definition, meaning, History regarding invention,
- Construction: Description of various parts & their significance in motor operation.
- Operation, working principle & basic equations.
- Speed Torque Characteristics, Effects of supply variations over the motor performance.
- Transformer equivalent circuit of induction motor
- Efficiency of induc tion motor-Various losses i n the ind uction motor.
- Types of insulating materials used & their temperature ranges,
- Product spectrum of Siemens motor.
- Comparison of normal & inverter driven motor.

Comparison of normal and energy efficient motor.

- Various reasons of high starting current of an inductio n motor & their effects on supply system
- Starters- DOL & star delta etc.
- Soft starter – brief overview.

- VFD – brief overview
- Advance control of induction motor-SIMOCODE overview.
- Installation & commissioning guidelines.
- Maintenance guidelines.
- General faults in the induction motor & their countermeasures i.e. Leads Overheating,Vibration,Abnormal sound etc

Prerequisite:

Diploma / Degree students in Electrical / Electronics / Instrumentation / ENTC / Biomedical / Mechanical Engineering

SIEMENS TRAINING

TRAINING PROGRAMME ON ON LOW VOLTAGE SWITCHGEAR

Course Duration: 5 days full time

Course Contents:

- Basics of power distribution – fa ult level calculation, definitions & terms used in Industry.
- Power products Range overview with latest Indian & International Standards an overview
- Low Voltage offerings in Power Distribution in Industry today · Air Circuit Breakers.
- Moulded case circuit breakers
- Switch disconnector fuse & Load break switches · DIN Fuse – Importance of fuses
- Control products with latest Indian & International Standards an overview · Basic Control Products used in Industry today.
- Contactor- New tech nology, Compactness DOL, RDO L & S-D a ssy – Hands on of Star-Delta assembly.
- Overload Relay, Microprocessor Relay- Why new versions of relays?
- Motor protection circuit breaker- why MPCB needs to be used?
- Soft starter- overview on use of soft starter

Prerequisite:

Diploma / Degree students in Electrical / Electronics / Instrumentation / ENTC / Biomedical / Mechanical Engineering

Automotive Crash & Impact Analysis using LS-Dyna

The objective of this course is to introduce the capabilities of explicit non-linear solver LS-DYNA to analyze complex impact problems. Detailed descriptions are given of the data required to run LS-DYNA analysis. Examples are used to illustrate the points made in the lectures.

COURSE CONTENT :

Introduction	▪ A quick overview of LS-DYNA computing environment ▪ Overview of LS-PrePost: Pre and Post processor for LS-DYNA ▪ Course content and material discussion ▪ Understanding the architecture of the solver ▪ Code structure in LS-DYNA
Building first LS-DYNA model & Theoretical Foundations	▪ Brief History of finite element simulation ▪ Sample ls-dyna conference presentations & sample simulations ▪ FE analysis (preprocessors, solver, postprocessors) ▪ Details of an example (tube collapse) ▪ Ls-dyna deck & using ls-prepost ▪ Details of postprocessing, output files, ascii, binary ▪ Element formulation & selection ▪ Brief description of available material models ▪ Boundary, and initial conditions, symmetry ▪ Modeling for physical phenomenon ▪ Contact modeling ▪ How to tell if your results are correct ▪ Error, debugging, and other useful information (d3hsp)
Practical Aspects	▪ Running LS-DYNA ▪ All the nuances of the code are demonstrated with live examples ▪ Use of different material models to solve engineering problems ▪ Understanding the significance of boundary conditions ▪ Contact problems and finding a cure for contact problems ▪ Choice of element type
Software Packages	▪ LS-DYNA ▪ LS-PrePost

> End of Document <

For Training Contact
George K. Thomas
Cell : +919447457762 | email : ctc@elsara.in