ADVANCE EMBEDDED CONTENTS

Embedded (os) Course Structure

www.semicoretech.com

Contact us : 9700779739, 8688018839

1st floor,sri sai Arcade, Near Aditya Trade Center,
Ameerpet, hyderabad, Telangana, 500016

Advanced Embedded (OS)Training Structure

This training structure constitutes of following classifications with combination of reasonable theory and most attention on related sample programs (+ projects)

1	Linux Basics (Duration - 10Hrs.)
2	Programming in “C” (Duration - 50 Hours)
3	Programming with “Data Structures” in “C” (Duration - 15 Hours)
4	Programming with “C++” under Linux (Duration – 40 Hours)
5	Microcontrollers (Duration – 30 Hours)
6	Shell Scripting (BASH) (Duration – 20 Hours)
7	Advanced Concepts (Duration – 60 Hours) • Linux System Programming (IPC) • Linux Network Programming • Introduction to Linux Kernel • Device Driver Programming • Building and Loading Kernel Image to Hardware Boards (Raspberry PI/Beagle Bone Black)

Daily Presentation/Self Learning Topics

1	Booting Process (including Run Levels)
2	Board bring up activities
3	Static Libraries
4	Dynamic Libraries
5	Makefile
6	Debugging with GDB
7	Numeric Conversions
8	Operations on Bits (Bit Set, Reset, Toggle and Checking Status)
9	Sorting Methods (Merge Sort, Quick Sort, Shell Sort)
10	Difference between Perl and Python
11	TCP/UDP/IP Protocols
12	ARM Architecture
13	Android Architecture and Framework
14	RTOS Concepts
15	Version Control Tools (GIT)
16	SDLC
17	Basic Testing Concepts (Test Plan preparation, setup, test cases development, test report)

NOTE: Most of the topics already covered in class, as these are important topicsincluded in the presentations. The presentation includes sample programs, wherever applicable.

Linux Basics

Introduction to Unix/Linux

- Unix and its history
- Introduction to Linux
- Login session
- Working with the Unix filesystem (Linux Directories)
- Linux Basic Commands (ls, pwd, touch, mkdir, rmdir, cp, mv, cat, rm)
- Handling files and directories (with metacharacters)
- Working with vi (visual editor)
- Linux documentation

File utilities

- Standard I/O, redirection and pipes
- Changing file access rights (users and permissions)
- Soft links and hard links
- Checking file integrity

Linux Utilities

- Disk utilities
- Process utilities
- Text processing utilities
- Miscellaneous commands
- Compressing and archiving (backup and restore) utilities
- User management, time management and shutdown

Programming in "C"

1. **Programming Language overview and essentials**
   - Types of Languages
   - IDE Environment (Editor, Compiler, Execution, Debugger)
   - Compilation steps or Compilation process toolchain (Preprocessor, Compiler, Assembler and Linker)
   - Debugging with GDB (GNU Debugger)
   - Sample C Programs

2. **Basic Syntax**
   - Basic Elements (Character set, Tokens, Semicolons, comments, whitespaces, Keywords, Identifiers)
   - Input and output
   - Constants (Numeric, Character and String)
   - Variables
   - Expression and Statement
   - Data types (char, int, float and double), Sign and Size Qualifiers
   - Numeric Conversions (Binary, Octal, Decimal and Hexadecimal)
   - Operators (Arithmetic, Assignment, Relational, Logical or Boolean, Increment/Decrement, Conditional, Comma, size of(), Bitwise and Implicit/Explicit Operators)
   - Operators Precedence and associativity
   - Type casting

3. **Control structures**
   - Conditional Statements (if, if-else, else-if ladder)
   - Nested if
   - Switch, nested switch
   - Repetitive Statements (For, While, Dowhile Loops)
   - Nested loops
   - Break statement
   - Continue statement
   - Goto statement
   - Infinite loop

4. **Functions**
   - Why do we need function?
   - Defining a function
   - Function declarations
   - Calling a function
   - Function arguments along with input and output types
   - Returning from function
   - Storage classes (viz., auto, register, static and extern) - Scope and Life Time
   - Recursion
   - Enum
   - Operations on Bits (Bit Set, Reset, Toggle and Checking Status)

5. **Arrays**
   - Basic operations on Array
   - Two dimensional arrays
   - Multi-dimensional arrays
   - Passing arrays as functional arguments
   - Strings

6. **Pointers**
   - Why do we need pointers?
   - What are pointers?
   - Using pointers
   - Dereferencing pointer variables
   - NULL pointers
   - Void pointers
   - Call by value and Call by reference
   - Pointer arithmetic
   - Precedence of dereferencing and increment/decrement operators
   - Pointer to pointer
   - Array of pointers
   - Pointer to an Array
   - Function pointers
   - Array of function pointers
   - Passing Pointers to functions
   - Return array/pointer from function
   - Passing pointers as parameters
   - Command line arguments

7. Memory Management

- Dynamic memory allocation
- Resizing and releasing memory

8. Structures and Unions

- Defining a structure
- Accessing structure members
- Typedef
- Structure padding
- Array of structures
- Arrays within structures
- Nested structures
- Pointer to structure
- Pointer within structure
- Structures and functions
- Defining a union
- Accessing union members
- Bit fields

9. File I/O

- What is File I/O?
- Types of file handling - Low Level and High Level
- Types of files
- File operations (Opening files, closing a file, Writing and Reading files)
- File I/O types (Character I/O, String I/O, formatted I/O and block I/O)
- Traversing within the file
- Error handling

10. Preprocessors

- Header files
- Macro Substitution
- Nested Macros
- Issues with Macros
- Parameterized Macros
- Macros vs functions
- Overloading functions in “C” way
- Stringizing and token pasting operators
- Conditional compilation, Debugging purposes
- Predefined macros

Programming with Data Structures in "C"

1. Introduction to Data Structures
- Why data structures?
- Efficient memory utilization and faster access
- Searching Techniques (Linear and Binary Search)
- Sorting Techniques (Bubble Sort)

2. Stacks and Queues
- Stacks using Arrays
- Queues using Arrays
- Circular Queue
- Stacks using Linked Lists
- Queues using Linked ListsInfix to postfix conversion of expression
- Solving Arithmetic expression using stacks

3. Linked Lists
- Operations on Linked Lists – Creation, insertion, deletion, search, display, count number of nodes, reversing listetc
- Singly Linked List
- Doubly Linked List
- Circular Linked List

4. Trees
- Introduction to trees
- Tree traversals (In-Order, Preorder and Post-Order)
- Binary Search Tree (Creation, Display and Deletion)

Microcontrollers

1. Microcontroller Basics
- 8051 Architecture.
- ARM Cortex- M4 Architecture.
- Systick timer.
- NVIC
- GPIO

2. Serial Communication Protocols
- UART
- ADC
- 12C
- SPI
- Timers

3. Hands On
- Understanding database of controller
- Initializing and accessing GPIO's
- Toggling LED.
- Interfacing button with both polling & interrupting method
- Interfacing 7-sigment display
- UART & use of Printf().
- Reading analog voltage using ADC of Microcontroller.
- 12C with MPU 6050 6-axis motion sensor
- 1.8 inches TFT LCD interfacing with SPI.
- DC Motor Control using PWM.

4. Projects
- Temperature Monitoring using STM32 & ESP8266 as a webserver.
- Audio Player Using STM32 Board.

5. Tools & Software's
- STM32 Cube IDE & STM32 CubeF4 Package.
- Keil MDK-ARM.

Complementary Benefits: Porting RTOS(Free RTOS) to STM32 Board Based on ARM Cortex-M4 architecture.

Object Oriented Programming using C++ under Linux

1. Introduction to C++
- Procedural vs Object-oriented programming
- Concepts of Object Oriented Programming
- Concepts of Class & Object
- Introduction to Encapsulation, Data Hiding/Data Abstraction
- Introduction to Polymorphism and Inheritance
- Message passing and Dynamic Binding
- C versus C++
- Static variables and member functions

2. Console I/O Operation, Character & String Handling
- Stream class – istream, ostream
- istream.h, ostream.h, iostream.h
- Cout, cin, insertion / extraction operators
- getc(), putc(), getline(), write()

3. Tokens
- Variables, Constants
- Keywords, Operators
- Reference variables

4. Class & Object
- Structure versus Class
- Defining Class
- Access specifier – private, public, protected
- Object, scope resolution operator
- Arguments to public functions
- Objects to public functions

5. Functions
- Return type in main, prototype
- Call by Reference, Return by Reference
- Inline functions and advantages over macros
- Friend function
- Functions with default arguments
- Functions overloading
- Constant arguments

6. Constructors & Destructors
- Uses and Usage
- Types of Constructors – default, parameterized, copy, multiple
- Constructor overloading
- Default arguments
- Dynamic Constructors
- Destructors

7. Operator Overloading
- Concept of Operator Overloading
- Overloading Unary & Binary Operators
- Using public & Friend functions

8. Inheritance
- Base class
- Derived class
- Public & private inheritance
- Inheritance mode - Types of inheritance (Single, Multiple, Multi-level, Hierarchical, Hybrid)
- Virtual classes

9. Polymorphism
- Introduction
- 'this' pointer, pointer to objects
- Static & Dynamic polymorphism
- Virtual functions and Abstract Classes (Pure Virtual Functions)

10. Introduction to Exception Handling
- What are Exceptions
- Handling Exceptions (try, catch, throw)

11. File Handling and I/O Streams
- Importance of Data management
- Introduction to streams
- Classes for file stream operation
- I/O operations using iostreams

12. Introduction to Templates
- Generic Programming
- Function Templates
- Class Templates

13. Introduction to STL
- Introduction to Containers, Algorithms and Iterators
- Sequence Containers (Vector, List, Deque, Arrays)
- Container Adaptors (Queue, Stack)
- Associative Containers (Set, Map)

Linux System Programming

01 GETTING STARTED

- The GNU project and Free Software Foundation
- UNIX history recap and Linux recap
- Linux/UNIX architecture overview
- Linux/UNIX filesystem overview
- Layout of C program in memory
- Introduction to Main memory

02 DEVELOPMENT TOOLS AND UTILITIES

- Brief on compilation process toolchain (Preprocessor, Compiler, Assembler and Linker)
- GDB debugger
- Make Utility
- Archive utility
- Object file format
- Executable or binary file formats
- Decoding ELF object files using binary tools
- Converting executable to different formats

03 THE GNU C LIBRARY AND SYSTEM CALLS

- Library Standards
- GNU C Library – glibc
- Library Functions (or API) vs System Calls
- Using System Calls
- Handling errors with errno
- Using strace

04 LINUX ENVIRONMENT

- Program Arguments (using argc, argv)
- Handling options with getopt()
- Environment variables
- Time functions
- User information
- Host information
- Temporary files
- Makefile program

05 BUILDING LIBRARIES

- Why use Libraries?
- Static versus shared
- Creating a static library
- Using a static library
- Creating a shared library
- Using a shared library
- Library locations

06 PROCESS CONTROL

- What is a process?
- Process relationships
- Creating a child process
- Related exec functions
- Alarms and timers
- Zombie, Daemon and Orphan processes

07 WORKING WITH FILES

- Overview
- Linux file structure
- Opening/Closing files
- Input/Output functions
- Repositioning file descriptors

08 INTER PROCESS COMMUNICATION

- Introduction to pipes
- Standard I/O: popen()/pclose()
- Using pipe()
- Named pipes
- Using named pipes
- IPC Overview (System V)
- Shared Memory (System V)
- Semaphores (System V)
- Message Queues (System V)

09 MANAGING SIGNALS

- What Signals are?
- Blocking/Checking Signals
- Working with Signals Sets
- Handling signals (with signal() and sigaction())
- Sending Signals

10 PROGRAMMING WITH THREADS (POSIX)

- Introducing Multi-Threaded Programming
- Creating Threads
- Synchronizing by Joining
- Detaching Threads
- Stopping Threads
- Synchronizing with Mutexes
- Using Mutexes
- Read/Write Locks
- Conditional Variables

Linux Network Programming

01 Introduction to Networking

- Need/Uses of Networking
- Use of Layered Architecture
- OSI Protocol Layers and its functionalities
- TCP/IP Protocol Layers
- Networking and internetworking devices
- LAN, MAN, WAN

02 TCP/IP Stack Internals

- Internet Addresses concepts
- IP Address vs H/W address
- Unicast, Broadcast and Multicast addresses
- Subnetting/Supernetting
- Internet Protocol (IP)
    - IP concepts
    - Routing concepts
    - User Datagram Protocol (UDP)
    - Transmission Control Protocol (TCP)

03 Socket Concepts and Programming

- Socket concepts
- Socket API Interface
- Client vs Server
- Connectionless and Connection oriented client-server communication
- Socket calls for UDP/TCP Server/Client
- Iterative and Concurrent servers
- Iterative Connection-less servers (UDP)
- Iterative Connection-Oriented servers (TCP)
- Concurrent server implementation (TCP) using multiple processes

Introduction to Linux Kernel

01 OPERATING SYSTEM / KERNEL CONCEPTS

- Memory Management (Virtual Memory, Paging etc)
- File System Management (VFS objects, file system types etc)
- Process Management (Process address space, creation, states, demand paging, copy on write etc)
- Device Management (Role of Device driver, classes of devices and modules etc)

02 INTRODUCTION TO LINUX KERNEL PROGRAMMING

- Kernel Classifications (Monolithic and Micro Kernels)
- User space and Kernel space
- Tool Chains, Libraries, Makefile
- The Linux Kernel
    - Getting the sources
    - Configuring the kernel
    - Diff and Patching utilities
    - Compiling the kernel
    - Installing and Booting the kernel
- Module Programming (The HelloWorld Module)
- System Calls
    - Registering a System Call
    - System Call Handler

03 DEBUGGING THE KERNEL

- Printk, traces and watches
- Information on kgdb, kdb
- User mode Linux
- Proc and Sys file systems

Linux Device Drivers Programming

01 CHARACTER DRIVERS

- Device Numbers
  - Major and Minor Numbers
  - Registering and Unregistering
  - Static and Dynamic Allocations
- Important Structures
  - File Operations
  - File
  - Inode
- Character Devices
  - Structure cdev
  - Adding, Allocating, Initializing and Deleting
  - User Space Applications and Device Driver mappings
  - Access methods within the driver, open, read, write and close
  - IOCTL

02 INTERRUPTS

- Handling I/O
  - I/O Architecture
  - I/O Mapped I/O
  - Memory Mapped I/O
- Interrupts and Registering Interrupt Handlers
- Interrupt Context vs Process Context

03 BLOCK I/O LAYER

- Block Device Structure
- Block Driver
- I/O Scheduling

Building Kernel Image

01 BOARD BRING UP ACTIVITIES

- Flashing Boot-loader Image
- Flashing Kernel Image
- Flashing File System Image
- Loading the kernel image into the Hardware Board.