

Embedded C Course Outline

Course Outline

COURSE OBJECTIVES :

Participants completing the Embedded C Course will be equipped with the technical skills required to develop a full Industrial strength Embedded System solution using 'C' as programming language. The course will cover all three major stages of the product development cycle:.

- Developing
- Implementing
- Integrating

SUITABLE FOR

Programmers and engineers who already have some understanding of programming and who now wish to gain a solid understanding of the use of C for embedded systems software development.

PREREQUISITES

Attendees should have a basic knowledge of C programming.

Days

3 days

Course Contents

Module 1: 'C' programming language

Brief history of 'C' programming language
Why C in Embedded
ANSI Standard

Fundamentals of C

Data types and Constants
Simple & Formatted I/O
Memory Usage
Operators & Expressions
Flow Control
Loops

Storage Classes

Scope and Life
Automatic, Static, External, Register, Volatile
Memory(CPU / RAM)

Functions

Role of Functions
Pass by value / reference
Returning values from Functions
Recursive Functions
Reentrant functions
Call Back Functions
Implications on Stack
Library Vs User defined function
Passing variable number of arguments
Functions and implication on Stack.

Arrays

Defining, initializing and using arrays

Multi Dimensional Arrays

Arrays of Characters and Strings

Arrays and Pointers

Passing arrays to functions

String handling with and without library functions

Structures & Unions

What structures are for

Declaration, initialization

Accessing like objects

Nested Structures

Array of Structures

Passing structures through functions

Allocation of memory and holes

Structure Comparison Structure bit operation

Typedef for portability

Unions

Overlapping members

Enumerated data types

Enum, Indexing,

enum Vs #define

Bit Operations

AND (&), OR (|), XOR (^)

Compliment (~)

Left-Shift (<<), Right Shift (>>)

Masking, Setting, Clearing and Testing of Bit / Bits

Pointers

The purpose of pointers

Defining pointers

The & and * operators

Pointer Assignment

Pointer Arithmetic

Multiple indirections

Advanced pointer types

Generic and Null Pointer

Function Pointers

Pointers to Arrays and Strings

Array of Pointers

Pointers to Structure and Union

Pointers to Dynamic memory

Far, Near and Huge Pointers

Pointer Type Casting

Complicated pointer declarations

Creating look up tables and Vector tables using pointers

Accessing peripherals using pointers

Dynamic Memory Allocation

Malloc(), Calloc(), Realloc(), Free()

Farmalloc(), Farcalloc()

File Handling Concepts

Concept of a FILE data type

File pointer

Character handling routines

Formatted Data Routines

Raw data Routines

Random Access to FILE

Advanced Data Structures

Linear & non-linear

Homogeneous & non-homogeneous

Static & Dynamic

Single, Double & Circular Linked Lists

Stacks & Queues

Binary Trees

Hash Tables

Sorting and Searching Techniques

Insertion sort

Selection sort

Bubble sort

Merge sort

Quick sort

Heap sort

Device Drivers: Introduction and development Techniques

What is a Device Driver

Types of Device Drivers

Building a device driver

Integrating a Device driver

Module 2: Real-time Operating Systems: Concepts

What is Real Time Computing

Why Real Time Operating System?

Multi-Tasking Programs

Theory of Operation

Overview of typical Real Time Operating System

RTOS Building Blocks

Task Management & Scheduler Events

Task Scheduling policies

Porting on to H/W

RTOS Implementation Examples

Module 3: Introduction to Development Tool chain

Overview

Specifications

Setting up Development and debugging environment

Compiler

Linker

Make utility

Object files

Multiple C file Program Development

Project management

Build process

Dynamically Loading Libraries (DLL)

Optimization Techniques

Programming Target