Embedded C and Embedded RTOS

48 Hours

Course Description

This course provides a comprehensive introduction to embedded systems programming using the C language and explores the fundamentals of Real-Time Operating Systems (RTOS). Participants will gain practical knowledge of C programming for embedded systems, covering topics such as variable manipulation, program flow control, advanced types, and hardware interaction. The course also delves into RTOS basics, types of operating systems, and their relevance in embedded systems.

Audience

This course is designed for software engineers, developers, and anyone interested in learning embedded systems programming with a focus on C language and RTOS basics. It is suitable for beginners with some programming experience as well as intermediate developers looking to enhance their skills in embedded systems.

Pre-requisite Knowledge/Skills

Participants should have a basic understanding of programming concepts and C language fundamentals. Familiarity with computer architecture and hardware concepts will be beneficial but is not mandatory.

Course Objectives

By the end of this course, participants will:

- Understand the characteristics and best practices of embedded systems.
- Master C programming for embedded systems, including variables, operators, and program flow control.
- Gain insights into advanced topics such as real-time concepts, interrupts, and scheduling techniques.
- Learn about data structures, driver design, and APIs for embedded systems.
- Explore the basics of Real-Time Operating Systems (RTOS) and their applications in embedded systems.

Course Outline

Module 1: Introduction to Embedded C

- Characteristics of Embedded Systems
- C Language Overview
- Structure of a C Program
- Identifiers
- Name Spaces and Scope
- Compilation & Linking
- MCU Boot Process
- C Best Practices for Embedded Systems

Module 2: Variables, Types, and Debugging

- MCU Architecture
- Program Execution
- Variables
- Representing Numbers
- Types
- Casting
- Debugging Embedded Systems

Module 3: Operators and Hardware Manipulation

- Understanding Register Maps
- Operators
- Bit Manipulation
- Modulus and Shifting
- Memory Addressing
- Sizeof
- Ternary Operator
- Precedence Rules
- Best Practices for Embedded Systems

Module 4: Basic Program Flow Control

- Software Design Cycle

- Software Architecture
- UML
- Flowcharts
- Round Robin Scheduling
- Statements
- For and While Loops
- If and Switch statements
- Infinite Loops
- Best Practices for Embedded Systems

Module 5: Advanced Flow Control

- Introduction to Real-time Concepts
- Interrupt Basics
- Interrupt Vector Tables
- Nesting and Priorities
- Software Interrupts
- Volatile keyword
- Shared Data Problems and Solutions
- RMA Analysis
- Interrupts Best Practice

Module 6: Advanced Types, Constants, and Expressions

- Enumerations
- Derived Types
- Literals
- Expressions and Evaluation
- State Machines
- State Charts
- Software Architecture Concepts

Module 7: Arrays and Pointer Basics

- Arrays
- Multidimensional Arrays
- Strings
- String Conversion
- Pointer Types
- Pointers and Arrays

- Pointers Operations
- Best Practices for Embedded Systems

Module 8: More Pointers and Strings

- Pointers to Pointers
- Pointers to Constants
- Constant Pointers
- String Libraries
- Manipulating Memory
- Best Practices for Embedded Systems

Module 9: Functions

- Syntax
- Variable Scope
- Recursion
- Inline Functions
- Software Metrics
- Static Code Analysis
- Testing Techniques
- Best Practices for Embedded Systems

Module 10: Structures and Unions

- Overview of Structures
- Unions
- Driver Design
- Defining APIs
- Driver Models
- GPIO Driver Example

Module 11: Scheduling Techniques

- Arrays of Pointers to Functions
- Function Queue Scheduling
- Cooperative Scheduling
- Scheduler Design
- Energy Profiling
- Low Power Software Design

Module 12: Declarations

- Syntax
- Storage Class Specifiers
- Global Variables
- Type Qualifiers
- Linkage Identifiers
- Best Practice for Embedded Systems

Module 13: Preprocessor

- #define
- Macros
- Precedence
- Conditional Compilation
- Warnings
- #pragma
- Predefined Macros

Module 14: RTOS Basics Tutorial

- System
- Operating System
- The Need for Operating System
- Computer System Components
- Abstract View of System Components
- Functions of Operating Systems
- Four main tasks of OS
- Shell
- Kernel

Module 15: Types of OS

- Batch Processing OS
- Time-Sharing Systems – Interactive Computing
- Real-Time Operating System

Module 16: Types of Embedded RTOS

- Embedded OS

Module 17: Types of System

- What is a Real-Time System?
- Multi-Tasking

Module 18: OS VS RTOS

- Types of RTS
    - Soft real-time
    - Hard real-time
    - Real-Time Spectrum
- What is the need for an RTOS?

Module 19: Polled Loop Systems

- Advantages
- Disadvantages

Module 20: Interrupt Driven Modules

- Overview of Interrupt-driven Programming
- Types of Interrupts
    - Hardware Interrupts
    - Software Interrupts
- Interrupt Service Routines (ISRs)
- Interrupt Vector Table (IVT)
- Enabling and Disabling Interrupts
- Priority Handling in Interrupts