Embedded C with STM32 (Basic + Fundamentals + RTOS with STM32)

Course Description

This course is designed to provide learners with a strong foundation in embedded systems programming using **C** and **STM32 microcontrollers**, progressing from basic programming concepts to peripheral interfacing and introductory **FreeRTOS** concepts. Participants will begin with the fundamentals of C, embedded systems, ARM Cortex architecture, and the STM32 family. The course then moves into STM32 architecture, memory organization, MPU and security, development environment setup using **STM32CubeIDE**, and practical firmware development.

Learners will gain hands-on exposure to **GPIO**, **interrupts**, **timers**, **PWM**, **ADC**, **DAC**, and communication protocols such as **UART, SPI, and I2C**. The course also introduces debugging practices using **J-Link Debugger**, along with STM32 debugging workflows. In the later part of the course, participants will explore **FreeRTOS basics**, including task management, scheduling, timing functions, and queues, and apply their learning in project-based implementation with **DHT11 sensor interfacing** and **servo motor control**.

Audience Profile

- Beginners in embedded systems development
- Students and engineers learning embedded C and STM32
- Electronics and instrumentation learners
- Professionals transitioning into embedded and IoT development
- Developers interested in firmware, RTOS, and hardware interfacing

---

Prerequisite

- Basic understanding of C programming
- Basic digital electronics knowledge

- Familiarity with microcontrollers (preferred)

Course Objectives

- Understand embedded systems programming using C
- Understand STM32 architecture and features
- Interface peripherals and external devices
- Develop reusable embedded firmware
- Work with memory protection and STM32 security concepts
- Implement GPIO, timers, PWM, ADC, and communication protocols
- Use **J-Link Debugger** for debugging
- Understand FreeRTOS fundamentals and task scheduling

TOC

Module 1: Introduction to C and Embedded Systems

- Why C in Embedded Systems
- Fundamentals of C
- ANSI C Standards
- Embedded Systems Overview
- ARM Cortex Overview
- STM32 Family Overview
- Applications

Module 2: STM32 Architecture and Development Setup

- STM32 Architecture
- Memory Organization
- MPU and Security

- STM32CubeIDE Setup
- Compiler and Toolchain
- Debugger Setup
- **J-Link Debugger**
- ST-Link Overview
- First Program (LED Blink)

Module 3: GPIO, Interrupts, and Debugging

- GPIO Programming
- HAL Library
- Digital Input/Output
- Interrupts and NVIC
- External Interrupts
- Debugging using IDE and J-Link

Module 4: Timers and PWM

- Timer Architecture
- PWM Generation
- Applications

Module 5: ADC, DAC, and UART Basics

- ADC Configuration
- Sensor Interfacing
- DAC
- UART Basics

Module 6: Communication Protocols

- UART
- SPI
- I2C
- Device Interfacing
- Debugging Communication

Module 7: FreeRTOS Fundamentals

- RTOS Concepts
- Task Management
- Scheduling
- Timing Functions
- Queues (Intro)
- Task prioritization

Module 8: Project Development with STM32

- DHT11 Sensor Interface
- Servo Motor Control
- Integration
- Testing and Debugging

Client-Side Requirements (To Be Arranged by Client)

Hardware Requirements

- STM32 Development Board (Recommended: **STM32F401 Nucleo-64 / NUCLEO-F401RE**)
- USB Cable for programming and power

- **J-Link Debugger** (External, if required beyond onboard ST-Link)
- Breadboard and jumper wires
- Sensors:
    - DHT11 Temperature & Humidity Sensor
- Actuators:
    - Servo Motor
- IoT Components (for extended use cases):
    - Wi-Fi Module (ESP8266 / ESP32)
    - Optional: Bluetooth Module (HC-05/HC-06)
- Power supply (if external peripherals used)

Software Requirements (Client Machine Setup)

- STM32CubeIDE
- STM32CubeMX (if required)
- J-Link Software & Drivers
- ST-Link USB Drivers
- Serial Terminal Tool (Tera Term / PuTTY)
- USB Drivers for STM32 boards
- Basic C toolchain (bundled with IDE)

IoT Extension

- Interfacing STM32 with Wi-Fi modules (ESP8266/ESP32)
- Sending sensor data to cloud/server
- Basic IoT architecture understanding (device → gateway → cloud)
