



L3 - Embedded C++

The C++ Language for Embedded Systems

Objectives

- Master the C++ language
- Use C++ Template (generic code) in Embedded Systems
- Master the C++ Advanced aspects such as polymorphism, single and multiple inheritances.
- Learn to redefine the C++ operators for dynamic memory allocation in embedded applications
- Make C++ objects persistent in programmable flash
- Manage C++ exceptions for Secure Embedded applications
- Use C++ objects to handle serial transmission / reception of character strings

Equipment

- Training manuals and software exercises
- One PC for two trainees
- One target board with ARM Cortex M4 microcontroller (STM32F)
- Eclipse environment and GCC compiler

Prerequisite

- C Language knowledge (see for example our [L2 - C language for Embedded MCUs](#) course)

Course Environment

- Theoretical course
 - PDF course material (in English) supplemented by a printed version.
 - The trainer answers trainees' questions during the training and provide technical and pedagogical assistance.
- Practical activities
 - Practical activities represent from 40% to 50% of course duration.
 - Code examples, exercises and solutions
 - One PC (Linux ou Windows) for the practical activities with, if appropriate, a target board.
 - ▶ One PC for two trainees when there are more than 6 trainees.
 - For onsite trainings:
 - ▶ An installation and test manual is provided to allow preinstallation of the needed software.
 - ▶ The trainer come with target boards if needed during the practical activities (and bring them back at the end of the course).
- Downloadable preconfigured virtual machine for post-course practical activities
- At the start of each session the trainer will interact with the trainees to ensure the course fits their expectations and correct if needed

Target Audience

- Any embedded systems engineer or technician with the above prerequisites.

Course Outline

First Day

Introduction to C++ for industrial systems

- Introduction to object oriented programming
- History and definition
- Overview on C++ standards
- Modern C++ objectives
- Switch from C to C++
- Embedded C++ specification (EC++)
- How to write optimized embedded code

Exercise: Understand function mangling

Exercise: Function inlining

Exercise: Volatile variable handling

C++ and embedded systems

- Object Oriented Programming in C++
 - Encapsulation
 - Classes and objects
 - Attributes and member functions
 - Object construction and destruction
 - Construction parameters
 - Copy constructor
 - Object composition and container
 - Scope qualifier operator

Exercise: Declaring classes and methods

Exercise: Working with default, copy and parameterized constructors

Exercise: Understand the differences between composition and aggregation

Second Day

C++ and embedded systems

- Operator Overloading
 - Optimizing parameter object passing
 - Overloading operators by member functions
 - Overloading operators by friend functions
 - Memory management operators overloading

Exercise: The assignment operator

Exercise: Overloading operators

- Simple Inheritance
 - Specialization by addition and substitution
 - Derivation and access rules
 - Construction during inheritance
 - Inheritance polymorphism
 - Virtual methods

Exercise: Understand inheritance

- Persistent and flashable objects
 - Constant and partially constant objects
 - Persistent objects

- Flashable objects

Exercise: Creating constant, mutable, persistent and ROMable objects

- Enhancing security with exceptions
 - Launching, capturing and handling exceptions
 - Retriggering exception
 - Exceptions specifications
 - Handling unexpected exception
 - Exception objects of the C++ standard library

Exercise: Handle errors using exceptions

Exercise: Unexpected exceptions management

Third Day

C++ Advanced Techniques

- I/O streams
 - C++ language standard streams
 - C++ standard libraries streams
 - Standard IO redirection by friend functions

Exercise: Redefine operators ‘ > ’ to read/write objects on an IO stream

- Member pointers
- Generic objects and templates
 - Classes and generic functions
 - Templates overloading
 - Specializing templates
 - STL (Standard Template Library)
 - Templates in embedded systems

Exercise: Generic classes and functions

- Polymorphic objects
- Virtual objects and abstract classes
- Specializing objects by simple inheritance
 - Building derivate objects
 - Access control rules for inherited objects
 - Specializing objects by multiple inheritance
 - Conflicts resolution by scope operator
 - Virtual inheritance

Exercise: Understand virtual methods by subclassing a generic Device class

Exercise: Understand multiple inheritance and virtual bases