oL3 - Embedded C++ Programming

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
- Manage C++ exceptions for Secure Embedded applications
- Use C++ objects to handle serial transmission / reception of character strings

Labs are conducted on a QEMU-emulated ARM-based board

Prerequisite

- C programming skills (see our oL2 - C Language for Embedded MCUscourse)

Course environment

- Theoretical course
  - PDF course material (in English)
  - Course dispensed using the Teams video-conferencing system
  - The trainer to answer trainees' questions during the training and provide technical and pedagogical assistance through the Teams video-conferencing system
- Practical activities
  - Practical activities represent from 40% to 50% of course duration
  - One Online Linux PC per trainee for the practical activities
  - The trainer has access to trainees' Online PCs for technical and pedagogical assistance.
  - Eclipse environment and GCC compiler
  - STM32F4-Discovery, QEMU Emulated, board
  - Downloadable preconfigured virtual machine for post-course practical activities.

Duration

- Total: 18 hours
- 3 sessions, 6 hours each (excluding break time)
- From 40% to 50% of training time is devoted to practical activities
- Some Labs may be completed between sessions and are checked by the trainer on the next session

Target Audience

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

Course Outline

First Session

Introduction to C++ for industrial systems

- Introduction to object oriented programming
- History and definition
- Overview on C++98/C++03/C++11/C++14/C++17/C++20
- Modern C++ objectives
- Switch from C to C++
- Embedded C++
- 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 Session

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 Session

C++ advanced techniques

- Member pointers
- Generic objects and templates
  - Classes and generic functions
  - Templates overloading
  - Specializing templates
  - STL (Standard Template Library)
  - Templates in embedded systems
- 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: Generic classes and functions
Exercise: Understand virtual methods by subclassing a generic Device class
Exercise: Understand multiple inheritance and virtual bases