MICRO CONSULT

As of 23.10.2019

Embedded C++ Advanced: Object-Oriented Programming for Microcontrollers with C++/EC++

The growing complexity of embedded software applications and the ever increasing performance of hardware has resulted in C++ being more and more used in embedded systems. Depending on the application, quality features like safety and security, performance, resource consumption etc. have to be taken into account.

Objectives

You know the application and efficiency of advanced C++ constructs (namespaces, templates, exception handling, runtime type identification, new style casts, multiple inheritance, memory management).

You are able to make informed decisions on using these constructs in your application.

You have an overview of the elements and mechanisms of the STL (Standard Template Library) and can already use it.

You can adapt patterns (state patterns, Singleton patterns, observer patterns, smart pointer patterns and layer patterns) to your applications and implement them.

Participants

The EC++ advanced training addresses programmers, software developers, software designers and software architects who use advanced C++ constructs in embedded software development.

Requirements

Basic knowledge of object oriented C++ is a must; basic knowledge of UML of advantage.

Embedded C++ Advanced: Object-Oriented Programming for Microcontrollers with C++/EC++

Content

C++ for Embedded Applications

- History
- Recommendations, rules and standards
- C++ compiler basics
- Practical tip with important references

Summary - Basic C++ Constructs and Efficiency Considerations

- Class and object
- Class elements
- Modifier for data, functions and objects
- Class relations (association, aggregation, composition and inheritance)
- Virtual functions and interfaces

Namespaces and Efficiency Considerations

- Using namespaces in program code
- Namespace alias, anonymous namespace, Koenig lookup
- Assembler, memory and runtime analysis and optimization
- Application examples in embedded software
- Exercise: Integration of namespaces in existing program code, based on the architecture

Single Inheritance, Multiple Inheritance and Efficiency Considerations

- Programming single inheritance and multiple inheritance (with interfaces)
- Issues related to multiple inheritance, solutions

- Virtual inheritance
- Assembler, memory and runtime analysis and optimization
- Application examples in embedded software
- Exercise: Using and programming multiple inheritance, virtually as an option

Exception Handling and Efficiency Considerations

- Exception handling - definition and programming
- Exception classes and hierarchies
- User exceptions
- C++ standard exceptions
- Assembler, memory and runtime analysis and optimization
- Application examples in embedded software
- Exercise: Integrating exception handling in the existing application

Memory Management and Efficiency Considerations

- Memory segments (BSS, stack, heap) for objects, comparison
- Dynamic memory management with new and delete
- Operator overload with new and delete
- Identifying risks and avoiding pitfalls
- Placement new
- Assembler, memory and runtime analysis and optimization
- Application examples in embedded software
- Exercise: Creating and deleting an object dynamically on the heap

Runtime Type Identification (RTTI)

- RTTI - definition and programming
- type_info class
- Consequences in use
- Relation to exception handling
- Assembler, memory and runtime analysis and optimization
- Exercise: Using RTTI for class identification at runtime

Type Conversion with New Style Casts and Efficiency Considerations

- Static, dynamic, const and reinterpret cast
- The right choice for use
- Relation to RTTI and exception handling
- Assembler, memory and runtime analysis and optimization
- Application examples in embedded software

Templates

- Template functions
- Template class and object
- Template parameters
- Inheritance and interfaces with template classes
- Assembler, memory and runtime analysis and optimization
- Practical tips: Static versus dynamic polymorphism
- Application examples in embedded software

C++ Libraries and Efficiency Considerations

- C++ string class
- Input/output stream library (I/O stream)
- String streams
- Standard template library (STL)
- Containers, iterators, adapters, algorithms
- Function objects
- Smart pointers
- Assembler, memory and runtime analysis and optimization
- Application examples in embedded software
- Exercise: Using the string class in the application and implementing the observer pattern (container based)

State Machine Implementation and Efficiency Considerations

- Basic possibilities of object-oriented implementation and modeling
- Implementation based on switch-case / if-else construct
- Implementation based on table execution
- Implementation based on state pattern and singleton pattern

- Comparison and assessment of different implementation types
- Inheritance of state machines
- Practical tip: Framework for state machine implementation

Programming (Quasi) Parallel Software - Specifics

- Race conditions
- Resource management, semaphore, mutex
- Issues and solutions
- Resource granularity
- Thread-safe programming
- Practical tips: Identifying risks and avoiding pitfalls

Operating System Abstraction (OSAL Operating System Abstraction Layer) in C++

- Benefit, advantages, disadvantages
- Using C++
- Practical example with FreeRTOS™

Hardware Abstraction, Hardware Drivers and Interrupts in C++

- Software quality features
- Layer patterns and application examples in embedded software
- Layer communication and callback mechanisms
- Object-oriented concepts and programming of standard peripheral drivers
- Object-oriented concepts and programming of interrupt handlers
- Communication and protocols
- Exercise: Integrating a driver and an interrupt service in the application

Practical Exercises in the Workshop

- You will use the Arm Keil MDK (microcontroller development kit) along with real hardware, based on an Arm Cortex®-M3 microcontroller throughout the exercise (watch application).

MicroConsult Plus:

- All participants have the following options to further use their exercises and the solutions developed by MicroConsult from this workshop:
- You take the files with you on a free USB stick provided by MicroConsult, or …
- You e-mail the files to your account, or …
- You get access to file download on request.
- You get all examples for C++ in electronic format and can easily adjust them to your development environment.
- You moreover get a tool and software component overview for developing embedded software.
- You get helpful notation overviews for UML (Unified Modeling Language) and SysML (Systems Modeling Language) in DIN-A3 format.

Trainings

Price *	Duration
2.430,00 €	4.5 days

Training code: E-EC++/FOR

* Price per attendee, in Euro plus VAT

Coaching

Our coaching services offer a major advantage: our specialists introduce their expertise and experience directly in your solution process, thus contributing to the success of your projects.

We will be happy to provide you with further information or submit a quotation tailored to your requirements.