oRT3 - Real Time Programming with FreeRTOS

Comprehensive FreeRTOS training: from Theory to Practice

Objectives

- Get an overview on Cortex-M4 architecture
- Discover the concepts of real time multitasking
- Understand Real Time constraints
  - Determinism
  - Preemption
  - Interrupts
- Understand the FreeRTOS architecture
- Discover the various FreeRTOS services and APIs
- Learn how to develop FreeRTOS applications
- Learn how to debug FreeRTOS applications

Prerequisites

- Familiarity with embedded C concepts and programming
- Basic knowledge of embedded processors

Course Environment

- Theoretical course
  - PDF course material (in English).
  - Course dispensed using the Teams video-conferencing system.
  - The trainer answers 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.
  - Code examples, exercises and solutions
  - 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.
  - QEMU Emulated board or physical board connected to the online PC (depending on the course).
  - Some Labs may be completed between sessions and are checked by the trainer on the next session.
- 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

Course duration

- Total: 20 hours
- 4 sessions, 5 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.

Evaluation modalities

- The prerequisites indicated above are assessed before the training by the technical supervision of the traineein his company, or by the trainee himself in the exceptional case of an individual trainee.
- Trainee progress is assessed in two different ways, depending on the course:
  - For courses lending themselves to practical exercises, the results of the exercises are checked by the trainer while, if necessary, helping trainees to carry them out by providing additional details.
  - Quizzes are offered at the end of sections that do not include practical exercises to verifythat the trainees have assimilated the points presented
- At the end of the training, each trainee receives a certificate attesting that they have successfully completed the course.
  - In the event of a problem, discovered during the course, due to a lack of prerequisites by the trainee a different or additional training is offered to them, generally to reinforce their prerequisites,in agreement with their company manager if applicable.

Plan

First Session

Cortex-M Overview

- ARMv7-M Architecture
- Cortex-M4 Architecture
- Registers and Execution States
- Privileges, Mode and Stacks
- Reset Behavior
- Exception and Interrupts
- The System Timer
- Memory Model
- Power Management
- STM32F407x Implementation Example

Exercise: *Create a new project*
Exercise: *Interrupt Management*

Real-Time Concepts

- Embedded system architectures
- Tasks and process
- Real-Time

Exercise: *Context Switch*

Second Session

Introduction to FreeRTOS

- The FreeRTOS Family
- FreeRTOS+Ecosystem
- Why use FreeRTOS
- FreeRTOS Code Structure

Scheduling

- Scheduler

- Schedulability

Task Management

- Creating Tasks
- Task Priorities
- Task States
- The idle task
- Delays
- Changing Task Priority
- Deleting Tasks
- Suspending Tasks
- Kernel Structures
- Thread Local Storage
- Kernel Interrupts on Cortex-M4
- Scheduling Traces
- Visual trace diagnostics using Tracealyzer

Exercise: *Task Management*
Exercise: *Periodic Tasks*
Exercise: *Task Statistics*

Third Session

Memory Management in FreeRTOS

- FreeRTOS Memory Managers
- Out of Memory management
- Stack overflow detection

Exercise: *Context Switch Measurement*

Resource Management

- Mutual Exclusion
- Critical Sections
- Mutexes
- Gatekeeper Tasks
- Lock-Free Data Structures

Exercise: *Resource Management*

Synchronization Primitives

- Queues
- Queues Sets
- Synchronization
- Events and Event Groups
- The Readers/writer problem
- Using Other Primitives within an ISR

Exercise: *Queue Management*
Exercise: *Readers Writer Problem*

Fourth Session

Interrupt Management

- Tasks and Interrupts
- FreeRTOS Binary and Counting Semaphores
- Task Notifications
- Stream Buffers

- Message Buffers
- Interrupt Nesting
- Low Power Support

Exercise: *Interrupt Management*
Exercise: *Tickless Mode*

Software Timer

- Software Timers
- Deferred Interrupt Handling

Exercise: *Implement Soft Timers*
Exercise: *Software Timer Management*

FreeRTOS-MPU

- The Cortex-M MPU
- The FreeRTOS-MPU Port
- Defining MPU Regions
- Creating User and System Tasks
- Practical Usage Tips

Appendix

Data Structures

- FIFO
- Linked list

Memory Management and Real-Time

- Memory Management
- Memory Errors

CMSIS-RTOS

- Overview
- Kernel Information and Control
- Threads Management
- Generic Wait Functions
- Communication and Resource Sharing
  - Semaphores
  - Mutex
  - Message Queue
  - Signal Events
  - Event Flags
  - Memory Pool
  - Mail Queue
- Timer Management
- Interrupt Service Routines

Renseignements pratiques

Inquiry : 24 hours
Prochaines sessions : from 13th to 16th of May, 2025 - Online EurAsia (9h-16h CET)
