HighIntegritySystems

WITTENSTEIN

FreeRTOS Training

This three-day course is designed for both individuals and organisations, covering the essentials of FreeRTOS features and application. The training can be delivered on-site worldwide or online, subject to availability.

Course Objectives

Completion of this course will provide the necessary practical experience to implement a real-time operating system within an embedded system. The course is made up of hands-on exercises combined with instruction and theory to illustrate the concepts of FreeeRTOS in practice. It is designed to familiarise the user with the concepts and commands necessary to make effective use of an RTOS within an embedded project. See the course outline for the content covered.

Who Should Attend?

Individuals or Organisations with an interest in, or tasked with development of, a resource constrained real-time system such as Software Engineers, Field Engineers or (project) Managers. All that is required is basic C programming knowledge.

Delivery Options

Training can be delivered on-site or online subject to availability. The workbooks, examples and boards stay with the students after course completion.

All training is instructor driven and the course material is provided in English. Delivery of the course can be in English or German as required.

Materials Provided	On-site Training	Online Training
Laptops	x	-
Boards	x	x
Electronic Workbook	x	x
Screen-sharing	-	x
Audio-Bridge	-	x

Booking Information

For more information, pricing or to book a course please contact us at:

sales@highintegritysystems.com

Course Outline

Introduction

- FreeRTOS
  - » An overview including licensing, software architecture, features, and the definition of “real time”
- Creating an example workspace with LPCXpressoIDE and LPC1769
  - » How to download, install, and import an example workspace
- Compile/run example

Task Management

- Tasks
  - » What is a task?
  - » Crude loop periodic delay, task parameter, priorities
  - » vTaskDelay() vs. vTaskDelayUntil(), starvation, idle hook
  - » Creation, states, priorities, idle task, task deletion
  - » Polling vs. event driven

- Scheduling
  - » What is an Operating System?
  - » What is a scheduler?
  - » Ask, determinism, multitasking
  - » Schedulers including Loop, cyclic executives, issues with interrupts, non-/pre-emptive, prioritized pre-emptive, rate Monotonic, deadline, cooperative, hybrid

WITTENSTEIN high integrity systems
Americas: +1 408 625 4712
ROTW: +44 1275 395 600

email: sales@highintegritysystems.com
web: www.highintegritysystems.com

Queue Management

- What is a queue?
- Creation, send, receive
- Blocking on read/write
- With multiple tasks blocking, who will run first?
- Large sets of data and queues
- Indirect/Direct synchronous and asynchronous message passing

Interrupt Management

- What is Interrupt vs. Polling?
- Hardware and Software-interrupts
- What if two interrupts arrive at the same time?
- What is re-entrant code?
- What is an event?
- What kind of events exist?
- What is a Semaphore and what can it be used for?
- What kind of Semaphores exist?
- Signal/Wait pattern, state diagram, event states of binary semaphore
- Counting semaphores
- Queues and ISRs
- Deferred interrupt processing, interrupt handlers, XXXFromISR(), task with interrupt synchronization, efficient queue usage even from within an interrupt, interrupt nesting

Resource Management

- Concurrency, concurrent read/writes
- Mutual exclusion, critical sections, suspending/locking the scheduler, mutexes, priority inversion, priority inheritance, deadlock, gate-keeper tasks, mutex vs. semaphore

Memory Management

- Memory types, fragmentation, memory exhaustion, memory allocation patterns, dynamic memory allocation in kernel
- Memory allocation schemes, xPortGetFreeHeapSize()

Trouble Shooting

- Spelunking, firmware standard, code review, binutils
- Debugging by stopping - FreeRTOS Simulator - State viewer
- Printf-stdarg.c, stack overflow and how to detect it
- Heap overflow and how to detect it
- Task statistics, run time statistics
- Profiling, tracing
- Common sources of error

FreeRTOS-MPU

- Memory management
- User vs. privileged mode, access permission, defining MPU regions, linker, practical usage tips
- Find out why a program failed

Trainer

Robert Berger has over 25 years practical and managerial experience in software design and development for embedded systems with and without hard real-time requirements. He has used GNU/Linux on desktop and server class machines, but mainly for embedded practices (automotive, industrial control, robotics, telecoms, consumer electronics, etc.).

Robert regularly attends international events as an expert and lecturer. His specialty is mainly training, but also consulting (in German or English) worldwide. Robert's expertise ranges from the smallest real-time systems (FreeRTOS) to set-ups with multiple processors/cores and embedded GNU/Linux (user-, kernel-space, device drivers, hardware interfacing, debugging, multi-core, Yocto Project) with a focus on free and open source software.

WITTENSTEIN high **integrity** systems

Worldwide Sales and Support
Americas: +1 408 625 4712
ROTW: +44 1275 395 600
Email: sales@high**integrity**systems.com
Web: www.high**integrity**systems.com