



RT3 - FreeRTOS Real Time Programming

Real-time programming applied to the FreeRTOS operating system

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

Course environment

- Convenient course material with space for taking notes
- Example code, labs and solutions
- A PC under Windows 7 for two trainees
- A ST STM32F4 (Cortex/M4) with System Workbench IDE

Prerequisites

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

Target Audience

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

Course Outline

First Day

Cortex-M resources used by RTOS

- Cortex-M Architecture Overview
 - Two stacks pointers
 - Different Running-modes and Privileged Levels
 - MPU Overview
 - SysTick Timer Description
- Exception / Interrupt Mechanism Overview
 - Interrupt entry and return Overview
 - SVC / PendSV / SysTick Interrupt Presentation
- Developing with the IDE

Exercise: Interrupt Management on Cortex-M4

Element of a real time system

- Base real time concepts
- The Real Time constraints
- Multi-task and real time
- Tasks and Task Descriptors
 - Content of the task descriptor
 - List of task descriptors
- Context Switch
- Task Scheduling and Preemption
 - Tick based or tickless scheduling
- Scheduling systems and schedulability proof
 - Fixed priorities scheduling
 - RMA and EDF scheduling
- Scheduling through FreeRTOS
 - Deterministic preemptive scheduling
 - Scheduling strategies
 - Cooperative scheduling
 - Hybrid scheduling

Exercise: Analyse a Context Switch

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

Second Day

FreeRTOS Memory Management

- FreeRTOS Memory Managers
- Out of Memory management
- Stack Overflow Management

Exercise: Check stack usage in existing programs

Resource Management

- Mutual exclusion through FreeRTOS
 - Critical sections (interrupt masking)
 - Suspending (locking) the scheduler
 - Mutexes
- Mutexes concepts

- Mutex or Semaphore
- Recursive or not recursive mutexes
- Priority inversion problem
- Priority inheritance (the automatic answer)
- Priority ceiling (the design centric answer)
- Gatekeeper tasks

Exercise: Implement mutual exclusion between tasks

Synchronization Primitives

- Introduction
 - Waiting and waking up tasks
 - Semaphores
 - Events
 - Mailboxes
- Binary Semaphores through FreeRTOS
 - Give a Binary Semaphore
 - Take a binary Semaphore
- Queue Management through FreeRTOS
 - Creation
 - Sending on a queue
 - Receiving from a queue
 - Data management
 - Sending compound types
 - Transferring large data
- Event groups
- Task Notifications
- Stream Buffers and Message Buffers

Exercise: Synchronizing a task with another one through binary semaphores

Exercise: Synchronizing a task with another one through queues

Exercise: Task Notifications

Exercise: Properly use stream Buffers

Exercise: Message Buffers

Parallelism Problems Solution

- Parallel programming problems
 - Uncontrolled parallel access
 - Deadlocks
 - Livelocks
 - Starvation

Exercise: The producer-consumer problem, illustrating (and avoiding) concurrent access problems

Exercise: The philosophers dinner problem, illustrating (and avoiding) deadlock, livelock and starvation

Third Day

Interrupt Management

- Need for interrupts in a real time system
 - Software Interrupt
 - Time Interrupts
 - Device Interrupts
- Level or Edge interrupts
- Hardware and Software acknowledge
- Interrupt vectoring
- Interrupts and scheduling
- Deferred interrupt processing through FreeRTOS

- Tasks with interrupt synchronization
- Using semaphores within an ISR
- Counting semaphores
- Using queues within an ISR
- FreeRTOS interrupt processing
 - Writing ISRs in C
 - Interrupt safe functions
 - Interrupt nesting

Exercise: Synchronize Interrupts with tasks

Software Timer

- The Timer Daemon Task
- Timer Configuration
- One-shot / Auto-reload Timer
- Software Timer API
- Deferred interrupt handling

Exercise: Implement Soft Timers

FreeRTOS-MPU

- The Cortex/M MPU
 - User and privileged modes
 - Access permissions
- Defining MPU regions
 - Overlapping regions
 - Predefined regions
 - Programmer-defined regions
- Needed linker configuration
- Practical usage tips

Exercise: Implement protected memory regions

appendixes

Data structures

- Need for specific data structures
- Data structures
 - Linked lists
 - Circular lists
 - FIFOs
 - Stacks
- Data structures integrity proofs
 - Assertions
 - Pre and post-conditions
- Thread safety

Exercise: Build a general purpose linked list

Memory Management

- Memory management algorithms
 - Buddy System
 - Best fit / First Fit
 - Pools Management
- FreeRTOS-provided memory allocation schemes
 - Allocate-only scheme
 - Best-fit without coalescing

- Thread-safe default malloc
- Checking remaining free memory
- Adding an application-specific memory allocator
- Memory management errors
- Stack monitoring

Exercise: Write a simple, thread safe, buddy system memory manager

Exercise: Write a generic, multi-level, memory manager

Exercise: Enhance the memory manager for memory error detection