[redacted]

Real-time programming applied to the MQX operating system

Objectives

- Discover the concepts of real time multitasking
- Understand Real Time constraints
  - Determinism
  - Preemption
  - Interrupts
- Understand the MQX architecture
- Discover the various MQX services and APIs
- Learn how to develop MQX applications
- Learn how to debug MQX applications
- Learn how to use MQX Library (USB, TCP/IP, File System, Embedded GUI)
- Get an overview on Cortex-M4 architecture

Course Environment

- Theoretical course
  - PDF course material (in English) supplemented by a printed version for face-to-face courses.
  - Online courses are dispensed using the Teams video-conferencing system.
  - The trainer answers trainees' questions during the training and provide technical and pedagogical assistance.
- Practical activities
  - Practical activities represent from 40% to 50% of course duration.
  - Code examples, exercises and solutions
  - For remote trainings:
    - One Online Linux PC per trainee for the practical activities.
    - The trainer has access to trainees' Online PCs for technical and pedagogical assistance.
    - 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.
  - For face-to-face trainings:
    - One PC (Linux ou Windows) for the practical activities with, if appropriate, a target board.
    - One PC for two trainees when there are more than 6 trainees.
  - For onsite trainings:
    - An installation and test manual is provided to allow preinstallation of the needed software.
    - The trainer come with target boards if needed during the practical activities (and bring them back at the end of the course).
- 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

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.

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 day

MQX at a Glance

- MQX overview
- Organization of MQX
- MQX directory structure
  - RTOS directory
  - PSP, BSP, I/0 and others source subdirectories
- Initializing and starting MQX
- Developing with NXP CodeWarrior Development Studio
  - Build projects
  - PSP build-project
  - BSP build-project
  - Post-build processing

Exercise: *Creating a simple MQX (only one task) project using NXP CodeWarrior Development Studio*

Introduction to Real Time

- Base real time concepts
- The Real Time constraints
- Multi-task and real time

Thread safe data structures

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

Exercise: *Build a general purpose thread safe doubly linked list*

Second day

Memory Management

- Memory management algorithms
  - Buddy System
  - Best fit
  - First Fit
  - Pools Management
- Understand how MQX RTOS Manages Memory
  - Memory with variable-size blocks
  - Lightweight memory with variable-size blocks
  - Memory with fixed-size blocks
  - Creating partitions
  - Allocating and freeing partition blocks
- Memory management errors
- Stack monitoring
- Controlling caches and MMU Overview (not present on K60 MCU)

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*
Exercise: *Detect stack overflow*

Element of a real time system

- 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

Managing and Scheduling tasks with MQX

- Managing Tasks
  - Creating tasks
  - Managing task errors
  - Terminating tasks
- Scheduling
  - FIFO scheduling
  - Round Robin scheduling

Exercise: *Creating your first multi-task project*

Third day

Timing with MQX RTOS

- Time components
- Timers
- Lightweight timers
- Watchdogs
- Hardware Timer on Cortex-M4

Exercise: *Turn a LED on and off using software timers*

Interrupt Management in Real Time Systems

- 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
- Handling Interrupts and Exceptions in MQX RTOS
  - Initializing interrupt handling
  - Restrictions on ISRs
  - Handling exceptions
  - Handling ISR exceptions
  - Handling task exceptions
- Handling Interrupts on Cortex-M4

Exercise: *Synchronize Interrupts with tasks*

Synchronization Primitives (1/2)

- Waiting and waking up tasks
- Semaphores
- Events
- Semaphores and Events through MQX RTOS
  - Events
  - Lightweight events
  - Lightweight semaphores
  - Semaphores

Exercise: *Synchronizing a task with another*

Fourth day

Synchronization Primitives (2/2)

- Mutual Exclusion
  - Spinlocks and interrupt masking
  - Mutex or Semaphore
  - Recursive or not recursive mutexes
  - Priority inversion problem
  - Priority inheritance (the automatic answer)
  - Priority ceiling (the design centric answer)
- Mutexes and condition variables
- Mutual Exclusion through MQX RTOS
  - Priority inversion
  - Priority inheritance
  - Priority protection
  - Mutexes
- Mailboxes
- Using Mailboxes through MQX RTOS
  - Messages
  - Queues
  - Task Queues

Exercise: *Implement mutual exclusion between two tasks*
Exercise: *Synchronizing tasks using queues*

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*

Debugging the application

- Instrumentation
  - Logs
  - Lightweight logs
  - Kernel logs
  - Stack usage utilities
- Run-Time Testing
- Embedded Debug

Exercise: *Debug an application with the log component*

Fifth day

Developing a New BSP

- Selecting a Baseline BSP
- Modifying Source Code

USB Library

- USB Host/Device Library Description
- USB Host/Device Stack Directory Structure
- USB Host and USB Device APIs Overview

Exercise: *Run a NXP USB HID Example*

File System (MFS) Library

- MFS at a Glance
- MFS API Overview

Exercise: *Use of MFS accessing the SPI-connect SD Card*

Shell Library

- Shell Library Overview

TCP/IP Stack (RTCS) Library

- RTCS Library Overview
- Setting up the RTCS
- Using Sockets
- RTCS API Overview
- RTCS Applications

Exercise: *Shell Command line providing microprocessor state information Demo*
Exercise: *Simple Web Server Demo*

Embedded GUI Library

- eGUI Library Overview
- Graphic Object Description
- Structure of Project with eGUI
- Driver API Overview

Exercise: *Embedded GUI Demo*

Renseignements pratiques

Inquiry : 5 days

SAS au capital de 138600 € - SIRET 449 597 103 00026 - RCS Nanterre - NAF 6202A
Centre de Formation, Siège social et administration : 19, rue Pierre Curie - 92400 Courbevoie - Tél. 01 41 16 80 10
Last site update: Wednesday 7 May, 2025 at :
https://www.ac6-training.com/en/