Programming applications using the MQX operating system

Objectives

- Becoming familiar with the NXP IDE, Kinetis SDK or CodeWarrior
- Get an overview of Kinetis and Cortex-M4 core architecture
- Revise the concepts of real time multitasking
- 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)

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

Cortex-M4 Overview

- Registers, Mode and Stacks
- Exception Management

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
  - Processor expert

Exercise: *Creating a simple MQX project using SDK or NXP CodeWarrior Development Studio and the Processor Expert Tool*

Second day

Managing and Scheduling tasks with MQX

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

Exercise: *Use the MQX API to manage tasks*

Memory Management

- Memory with variable-size blocks

- Lightweight memory with variable-size blocks
- Memory with fixed-size blocks
  - Creating partitions
  - Allocating and freeing partition blocks

Exercise: *Managing memory*

Synchronizing Tasks

- Synchronizing tasks through MQX RTOS
  - Events
  - Lightweight events
  - Lightweight semaphores
  - Semaphores
  - Lightweight Messages queue
- Mutual Exclusion through MQX RTOS
  - Create critical sections
  - Mutexes
  - Avoiding Priority inversion

Exercise: *Synchronizing tasks using MQX semaphores API*
Exercise: *Create mutual exclusion using MQX semaphores API*

Third day

Interrupt Management

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

Exercise: *Use the MQX API to handle interrupt*

Timing with MQX RTOS

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

Exercise: *Using software timers*

Debugging the application

- Instrumentation
  - Logs
  - Lightweight logs
  - Kernel logs
  - Stack usage utilities

Exercise: *Debug an application with the log component*

IO Drivers at a glance

- Drivers architecture
- Installing Drivers
- Using Drivers

Exercise: *Hands-on: Working with the ADC Driver*

Getting Started with the MQX Libraries

- How to start with the different libraries (MFS, Shell, RTCS, USB) using the providing examples
  - RTCS and Shell Libraries at a glance

Exercise: *Running a simple TCP echo Server / HTTP server application*

Renseignements pratiques

Inquiry : 3 days