OS2 - MQX Programming on Kinetis Microcontroller

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

- Convenient course material with space for taking notes
- Documentation, labs and solutions
- A PC under Windows 7 for two trainees
- A NXP Kinetis K70 or K60 (Cortex/M4) with SDK or CodeWarrior 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-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