# RT6 - Real Time Programming with Eclipse ThreadX

## Real-time programming applied to ThreadX (previously Azure RTOS)

## Objectives

- Get an overview on Cortex-M architecture
- Discover the concepts of real time multithreading
  - Understand Real Time constraints
  - Determinism
  - Preemption
  - Interrupts
- Understand the Azure RTOS architecture
- Discover the various Azure RTOS services and APIs
- Learn how to develop Azure RTOS applications
- Learn how to debug Azure RTOS applications

### *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-M Overview

- ARMv7-M Architecture
- Cortex-M Architecture
- Registers and Execution States
- Privileges, Mode and Stacks
- Reset Behavior
- Exception and Interrupts
- The System Timer
- Memory Model
- Power Management
- Cortex-M Advanced Features
  - ARM v7/v8 MPU
  - TrustZone Security Extension

***Exercise:*** *Create a new project*
***Exercise:*** *Interrupt Management*

## Real-Time Concepts

- 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
  - Deterministic preemptive scheduling
  - Scheduling strategies
  - Cooperative scheduling
  - Hybrid scheduling

***Exercise:*** *Context Switch*

## Introduction to Azure RTOS

- The Azure ThreadX
- Azure RTOS Ecosystem
- Azure RTOS Features
- Installation and use of Azure RTOS

## Second Day

## Thread Management

- Thread Control Block
- Thread States
- Thread Creation
- Thread Deletion
- Preemption-Threshold
- Changing Thread Priority
  - Suspending Threads
  - Resume Thread Execution
  - Thread Sleep
- Terminate Thread Execution
- Time-Slice
- Thread States and Thread Design
- Thread Statistics
- Visual trace diagnostics using Tracealyzer

***Exercise:*** *Thread Management*
***Exercise:*** *Periodic Threads*
***Exercise:*** *Time-slice change*
***Exercise:*** *Thread Statistics*

## Memory Management in Azure RTOS

- Azure RTOS Memory Managers
  - Memory Byte Pool
  - Memory Block Pool
- Out of Memory management
- Stack overflow detection

***Exercise:*** *Context Switch Measurement and memory problems detection*

## Resource Management

- Mutual Exclusion
- Critical Sections
- Mutexes
  - Recursive mutexes
- Gatekeeper Threads
- Lock-Free Data Structures

***Exercise:*** *Implement mutual exclusion between tasks*

## Third Day

## Synchronization Primitives

- Queues
- Synchronization
- Semaphores
  - Binary and counting semaphores

- Events and Event Groups
- The Readers/writer problem

***Exercise:*** *Sending messages between tasks*
***Exercise:*** *Synchronizing a task with another one (Producer/Consumer problem)*
***Exercise:*** *Readers/Writer Problem*

## Interrupt Management

- Threads and Interrupts
  - Synchronization between threads and interrupts
- Interrupts on ARM Cortex-M
- Handler thread
- Azure RTOS Primitives Within An ISR
  - Queues Within An ISR
- Low Power Support

***Exercise:*** *Interrupt Management & Deferred interrupt processing*
***Exercise:*** *Tickless and Low Power Mode*

## Application Timers

- Application Timers
- System Timer Thread
- One-shot timers
- Auto-reload timers
- Application Timer Commands

***Exercise:*** *Implement Soft Timers & Synchronize a task with a timer*

# Annexes

## Data Structures

- FIFO
- Linked list

## Memory Management and Real-Time

- Memory Management
- Memory Errors

## CMSIS – RTOS

- Overview
- Kernel Information and Control
- Threads Management
- Generic Wait Functions
- Communication and Resource Sharing
  - Semaphores
  - Mutex
  - Message Queue
  - Signal Events
  - Event Flags
  - Memory Pool
  - Mail Queue
- Timer Management
- Interrupt Service Routines

## Renseignements pratiques

**Duration :** **3 days**
**Cost :** **2510 € HT**