



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

- Example code, labs and solutions
- STM32L5 Board on STM32CubeIDE

Prerequisites

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

Environnement du cours

- Cours théorique
 - Support de cours au format PDF (en anglais) et une version imprimée lors des sessions en présentiel
 - Cours dispensé via le système de visioconférence Teams (si à distance)
 - Le formateur répond aux questions des stagiaires en direct pendant la formation et fournit une assistance technique et pédagogique
- Activités pratiques
 - Les activités pratiques représentent de 40% à 50% de la durée du cours
 - Elles permettent de valider ou compléter les connaissances acquises pendant le cours théorique.
 - Exemples de code, exercices et solutions
 - Pour les formations à distance:
 - ▶ Un PC Linux en ligne par stagiaire pour les activités pratiques, avec tous les logiciels nécessaires préinstallés.
 - ▶ Le formateur a accès aux PC en ligne des stagiaires pour l'assistance technique et pédagogique
 - ▶ Certains travaux pratiques peuvent être réalisés entre les sessions et sont vérifiés par le formateur lors de la session suivante.
 - Pour les formations en présentiel:
 - ▶ Un PC (Linux ou Windows) pour les activités pratiques avec, si approprié, une carte cible embarquée.
 - ▶ Un PC par binôme de stagiaires s'il y a plus de 6 stagiaires.
 - Pour les formations sur site:
 - ▶ Un manuel d'installation est fourni pour permettre de préinstaller les logiciels nécessaires.
 - ▶ Le formateur vient avec les cartes cible nécessaires (et les remporte à la fin de la formation).
- Une machine virtuelle préconfigurée téléchargeable pour refaire les activités pratiques après le cours

- Au début de chaque session (demi-journée en présentiel) une période est réservée à une interaction avec les stagiaires pour s'assurer que le cours répond à leurs attentes et l'adapter si nécessaire

Audience visée

- Tout ingénieur ou technicien en systèmes embarqués possédant les prérequis ci-dessus.

Modalités d'évaluation

- Les prérequis indiqués ci-dessus sont évalués avant la formation par l'encadrement technique du stagiaire dans son entreprise, ou par le stagiaire lui-même dans le cas exceptionnel d'un stagiaire individuel.
- Les progrès des stagiaires sont évalués de deux façons différentes, suivant le cours:
 - Pour les cours se prêtant à des exercices pratiques, les résultats des exercices sont vérifiés par le formateur, qui aide si nécessaire les stagiaires à les réaliser en apportant des précisions supplémentaires.
 - Des quizz sont proposés en fin des sections ne comportant pas d'exercices pratiques pour vérifier que les stagiaires ont assimilé les points présentés
- En fin de formation, chaque stagiaire reçoit une attestation et un certificat attestant qu'il a suivi le cours avec succès.
 - En cas de problème dû à un manque de prérequis de la part du stagiaire, constaté lors de la formation, une formation différente ou complémentaire lui est proposée, en général pour conforter ses prérequis, en accord avec son responsable en entreprise le cas échéant.

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

Renseignements : 3 jours