Objectifs

- Découvrir les concepts de base du multi-tâches et du temps réel
- Mettre en pratique la méthode UML dans les différentes phases d'analyse, de conception et de codage d'une application temps réel
- Maîtriser les difficultés de la programmation concurrente
- Connaitre les standards applicables
- Découvrir les contraintes temps réel (déterminisme, interruptions, préemption...)
- Découvrir les bonnes pratiques de la programmation temps réel

De nombreux exercices permettent d'apréhender l'analyse et la conception d'un système temps réel ainsi que sa réalisation.
Pour assurer une bonne compréhension du fonctonnement d'un système temps réel le satagiaire code un noyau minimal et ses différents mécanismes.

Matériel

- Un PC et une carte cible par binôme
- Chaîne de compilation croisée
- Exercices sur un atelier UML
- Manipulations en environnements croisé sur carte cible
- Un support de cours

Pré-requis

- Connaissance de la programmation en C (niveau cours L2)
- Connaissance d'un microprocesseur souhaitée
- Connaissance de la programmation embarquée

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

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

Premier jour

Introduction au temps réel

- Concepts temps réel de base
- Contraintes particulières du temps réel
- Programmation structurée et objet
- Apports des techniques objets

Introduction à UML

- Genèse d'UML
- Modèles UML standards
- Cycle de développement Objet
- Contraintes liées à l'interprétation des diagrammes
- Interprétation des diagrammes
- Définition de nouveaux diagrammes
- Cycle de développement avec RT UML

Le langage UML en contexte temps-réel

- Modélisation statique
  - environnement du système
  - cas d'utilisation
  - contraintes non fonctionnelles
  - modèles de classes
- Modélisation dynamique
  - diagrammes de Séquence
  - diagrammes de Collaboration
  - diagrammes Etats Transitions
  - contraintes temporelles
  - gestion du parallélisme

Analyse avec UML

- L'aspect statique
  - spécification du diagramme de contexte
  - formalisation des contraintes non fonctionnelles

  - description des cas d'utilisation
  - identification des classes de haut niveau
  - ébauche du modèle de classes
- L'aspect dynamique
  - formalisation des cas d'utilisation par les scénarios
  - ajout des aspects temporels dans le diagramme de séquence
  - comportement du système et Diagramme Etat transition
  - ajout des objets d'interface dans les Diagrammes de séquences

Exercise: *analyse d'un système temps-réel simple*

Second jour

Conception générale avec UML

- Affinage des modèles de spécification
  - organisation du système en sous systèmes et packages
  - description dynamique des classes
  - diagramme de comportement de chaque classe
  - intégration des Interfaces d'entrée/sortie
  - prise en considération des objets de stockage
- Gestion multiprocess et multitâches
  - différents types d'architectures logicielles
  - identification des taches
  - allocation des sous-systèmes aux processeurs et aux taches
  - communication inter-process
  - synchronisation
- Environnement d'exécution des tâches
  - problèmes liés au choix de l'exécutif
  - interprétation des mécanismes temps réel en fonction du type d'architecture

Exercise: *conception générale du système analysé précédemment*

Conception détaillée et codage avec UML

- La préparation au codage
  - description détaillée des méthodes et attributs
  - critères d'optimisation
  - affinage de l'héritage
  - classes abstraites, template ...
  - associations et pointeurs ...
- Le codage incrémental
  - définition des incréments
  - implémentation des objets de stockage
- UML et les langages de programmation
  - passage UML -> C
  - passage UML -> C++

Exercise: *conception détaillée et passage au codage du système*

Troisième jour

Particularités de la programmation dans le contexte embarqué

- Les tableaux de pointeurs
- Les structures à champ de bits
- Les unions
- Problèmes spécifiques à l'embarqué
  - les formats big et little endian
  - les attributs "const" et "volatile"
- Utilité des tableaux de pointeurs sur fonctions

Principe de fonctionnement d'un système Temps réel

- Notion de tâche
- Sauvegarde de contexte
- Cadencement des tâches selon leur priorité, préemption
- Déclenchement des commutations de tâches

Structures de données en environnement multi-tâches

- Structures de données:
  - Listes simplement chaînées
  - Listes doublement chaînées
  - Listes circulaires
  - Files d'attentes
  - Piles

Exercise: *réalisation de listes chaînées utilisables en contexte multi-tâches*

Quatrième jour

Gestion de la mémoire

- Algorithmes
  - Best fit
  - First fit
  - Buddy
  - Pool

Exercise: *réalisation d'un allocateur mémoire stable*

- Problèmes de gestion mémoire
- Gestion des fuites mémoire

Exercise: *mise en évidence et détection de fuites mémoire*

Gestion des interruptions

- Nécessité des interruptions dans un système temps réel
  - interruptions périodiques
  - interruptions des périphériques
- Interruptions sur niveau ou sur front
- Acquitement matériel ou logiciel
- Vectorisation des interruptions

Exercise: *écriture d'un gestionnaire d'interruptions*

Eléments d'un système temps réel

- Tâches et descripteurs de tâches
  - contenu du descripteur de tâche
  - listes de tâches
- Changement de contexte

Exercise: *écriture du code de changement de contexte*

- Ordonnancement et préemption
  - ordonnancement avec ou sans tic d'horloge

Exercise: *écriture d'un ordonnanceur à priorité fixe*

- Systèmes d'ordonnancement et preuves d'ordonnançabilité
  - ordonnancement à priorités fixes
  - ordonnancements RMA et EDF (Earliest Deadline First)

Exercise: *étude d'un ordonnanceur EDF*

  - ordonnancement adaptatif

Exercise: *écriture d'un ordonnanceur adaptatif utilisant un tic d'horloge*

Cinquième jour

Communication et synchronisation

- Mise en sommeil et réveil de tâches
- Exclusion mutuelle
  - spinlocks
  - masquage d'interruptions
  - supression de la préemption
  - mutex
- Mutex et variable condition

Exercise: *implementation du mécanisme mutex/variable condition*

- L'inversion de priorité
  - l'éritage de priorité (la solution automagique)
  - Le plafond de priorité (la solution réfléchie)
  - mutex récursifs et non récursifs

Exercise: *mise en place du plafond de priorité*

- Sémaphores
- Boites aux lettres

Exercise: *réalisation de Sémaphores et de Boites aux lettres à partir de mutex*

Pièges et bonnes pratiques du temps-réel

- Le problème de l'étreinte fatale (Dead-lock)
  - origine
  - solutions possibles
  - intérêt du plafond de priorité dans ce contexte

Exercise: *mise en évidence dans le problème du dîner des philosophes*

- Le problème du "live-lock"
- Le problème de la famine

Exercise: *résolution du problème lecteurs/écrivains*

- Le problème des contentions
  - en lecture

Exercise: *réalisation d'un mécanisme read-copy-update*

  - en écriture

Exercise: *réalisation d'un mécanisme de verrou séquentiel*

Les standards du temps réel

- Les standards d'environnements temps réel
  - POSIX
- Les langages à sémantique temps-réel
  - Java
  - Ada

Renseignements pratiques

Inquiry : 5 days