Systèmes Embarqués 2 pour les classes T-2/I-2

Objectifs de l'examen oral 2018

Les étudiant-e-s devront être capables :

Prérequis : programmation en assembleur et en C

- *de coder dans les règles de l'art un traitement simple en assembleur ARM*
- *d'utiliser et de reconnaître les différents modes d'adressage présent sur le ARM*
- *de manipuler correctement les types de base de C*
- *de coder un algorithme de traitement de données en C*
- *de coder dans les règles de l'art des appels de fonctions en C*
- *de décrire le passage d'arguments par valeur et référence lors d'appel de fonction en C*
- *de différencier les fonctions globales des fonctions locales dans une application en C*
- *de différencier les variables globales des variables locales et des variables rémanentes dans une application en C*
- *d'expliquer les conditions pour qu'une fonction C soit réentrante*
- *de manipuler correctement les types complexes (énumérations, tableaux, structures, unions …) de C*
- *de concevoir une interface C permettant d'accéder aux registres d'un périphérique*
- *de manipuler corrections les pointeurs en C*
- *de décrire les conversions des types en C*
- *de manipuler correctement les pointeurs de fonction en C*

***Remarque** : un exemplaire du jeu d'instructions du processeur ARM sera mis à disposition, document « 05_ARM_Instruction_Set_Summary.pdf »*

100. Interfaçage assembleur – C

100.01 d'utiliser correctement la pile (stack) dans un programme codé en assembleur

100.02 de programmer une sous-routine en assembleur ARM en utilisant la pile pour le passage de paramètres et pour les variables locales

100.03 de représenter le passage de paramètres à des sous-routines et d'expliquer le retour de valeurs et du résultat pour interfacer avec des programmes écrits en C

100.04 de représenter l'image de la pile lors de l'appel d'une sous-routine en assembleur

100.05 de coder dans les règles de l'art l'appel de routines développées en C depuis un code en assembleur ARM

100.06 de spécifier correctement l'appel de routines en assembleur depuis un code C

100.07 de décrire l'utilisation des registres des µP ARM en assembleur

Systèmes Embarqués 2 pour les classes T-2/I-2

Objectifs de l'examen oral 2018

120. Programmation orientée-objet en C

120.01 de décrire les concepts de programmation orientée objet en C

120.02 de décrire l'héritage simple et multiple en C

120.03 de concevoir des programmes orientés objet en C

140. Chaîne d'outils

140.01 de décrire l'organisation des fichiers sources

140.02 de décrire la gestion de versions et révisions ainsi que des outils utilisés

140.03 de décrire les méthodes de documentation d'applications en C et des outils utilisés

140.04 de décrire les méthodes de vérification et de tests en C

140.05 de concevoir un programme de tests unitaires en C

140.06 de décrire sommairement la fonction des différents utilitaires utilisés pour le développement de logiciel en C

140.07 d'expliquer un Makefile utilisé pour la génération d'applications codées en langage C (plusieurs cibles, règles suffixes, dépendances, conditions, sous-make,…)

200. Interruptions

200.01 d'expliquer les différentes phases de traitement d'une interruption (séquence)

200.02 de classer les types d'interruptions et d'exceptions

200.03 d'expliquer la fonctionnalité de la table des vecteurs d'interruptions sur les processeurs ARM, ainsi que l'implémentation spécifique du µP TI AM335x

200.04 de reconnaître les modes d'opération du processeur

200.05 d'expliquer comment l'on peut passer du mode superviseur au mode utilisateur et vice versa

200.06 d'utiliser correctement les pointeurs de pile dans les différents modes d'opération des processeurs ARM

200.07 de décrire le concept de commutation de contexte, de latence et de gigue

200.08 de décrire le système d'interruption des processeurs ARM et du µP TI AM335x

200.09 d'expliquer les mécanismes d'activation et de désactivation des interruptions matérielles (µP ARM – core –, µP TI AM335x – intc, gpio, … –)

200.10 de décrire le principe de niveaux de priorité

200.11 de décrire la procédure de reconnaissance d'interruption multiple

200.12 de décrire et de concevoir un gestionnaire d'interruption

200.13 de programmer en assembleur une application utilisant une interruption

200.14 de décrire et de concevoir des opérations atomiques

Systèmes Embarqués 2 pour les classes T-2/I-2

Objectifs de l'examen oral 2018

220. Systèmes d'exploitation

220.01 de citer quelques techniques et méthodes de développement pour des systèmes embarqués

220.02 de décrire les différents types de systèmes multitâches (systèmes d'exploitation)

220.03 de décrire les différentes composantes d'un noyau

220.04 de décrire les ressources globales, partagées et privées d'un thread

220.05 de décrire les états principaux d'un thread

220.06 de décrire la commutation de contexte entre deux threads

220.07 de concevoir un algorithme de transfert de contexte entre deux threads

220.08 de concevoir l'initialisation du contexte d'un thread

220.09 de concevoir un ordonnanceur (scheduler) élémentaire

220.10 de concevoir un mécanisme de synchronisation simple (sémaphore)

220.11 de décrire les éléments et structures nécessaires à la gestion d'un thread

220.12 de décrire la fonction d'un ordonnanceur (scheduler) élémentaire

240. Entrées/Sorties

240.01 de décrire le concept et la structure générale des entrées/sorties

240.02 de décrire les différents modes et techniques pour piloter des périphériques d'entrées/sorties

240.03 de différencier une programmation interruptive d'une programmation par scrutation d'une entrée (interrupt vs. polling)

240.04 de coder une entrée/sortie en mode interruptif

240.05 de coder une entrée/sortie en mode par scrutation

240.06 de calculer la taille des tampons d'émission et de réception sous des conditions données

300. Memory Management Unit (MMU)

300.01 de décrire la fonctionnalité (les rôles) de la MMU

300.02 de décrire l'architecture de la MMU

300.03 de décrire les mécanismes de translation d'adresses

300.04 de décrire le rôle de la TLB

300.04 de décrire l'implémentation de la MMU sur le µP TI AM335x

Systèmes Embarqués 2 pour les classes T-2/I-2

Objectifs de l'examen oral 2018

310. Mémoire cache

310.01 de décrire la hiérarchie de la mémoire sur des systèmes à µP

310.02 de décrire la fonctionnalité (les rôles) d'une mémoire cache

310.03 de décrire les 2 principes de localité spatiale et temporelle

310.04 de décrire les mécanismes de la mémoire cache (identification d'une ligne, placement d'une ligne, recherche d'une ligne, etc.)

310.05 de décrire la différence entre une mémoire cache virtuelle et physique

310.06 de décrire l'implémentation de la mémoire cache sur le µP TI AM335x

310.07 d'expliquer pourquoi certains algorithmes ont de meilleures ou moins bonnes performances au niveau du µP et de sa mémoire cache

400. Direct Memory Access (DMA)

400.01 d'expliquer la fonctionnalité d'un DMA

400.02 de décrire le principe et l'architecture DMA

400.03 de décrire la fonctionnalité des DMA sur le processeur TI AM335x

500. Travail pratique et mini-projet

501.01 de décrire l'architecture du système de fichiers virtuels (VFS) et d'expliquer son utilité

501.02 de décrire la structure de données de base du système de fichiers virtuels (VFS)

502.03 de concevoir un pilote pouvant être attaché au système de fichiers virtuels (VFS)

502.01 de décrire le principe de fonctionnement d'une shell (ligne de commande / CLI)

502.02 de décrire la structure de données de base d'une shell (ligne de commande / CLI)

502.03 de concevoir une nouvelle commande pour la shell (ligne de commande / CLI)

503.01 de décrire l'architecture et le fonctionnement de l' « Interrupt handler » conçu pour le traitement des interruptions matérielles (périphériques internes du µP et des GPIO)

504.01 de décrire l'architecture et le fonctionnement du noyau élémentaire offrant des services pour les threads et les sémaphores

504.02 de décrire et concevoir un service noyau pour la mise en pause d'un thread pour un certain laps de temps

505.01 de décrire le principe de fonctionnement d'un timer du processeur TI AM335x

505.02 de décrire et concevoir un algorithme permettant de générer une horloge avec une granularité de 1/24MHz sur une période de plus de 100 ans à partir d'un timer avec une période de moins de 3 minutes