Systèmes Embarqués 1 & 2

Objectifs du travail écrit n° 2

Les étudiant-e-s devront être capable :

Langage assembleur GNU/ARM

- de décrire succinctement le processus de développement
- de décrire la syntaxe et les directives de l'assembleur
- de coder dans les règles de l'art un traitement simple en assembleur
  - directives pour la déclaration de symboles publiques (fonctions, variables, …)
  - déclaration des différentes sections (.text, .bss, .data, …)
  - directive d'alignement des sections
- de reconnaitre et utiliser les différents modes d'adressage du processeur ARM
- de concevoir des programmes en langage assembleur
  - utilisation des directives de l'assembleur
  - utilisation du jeu d'instructions
  - utilisation des différents modes d'adressages
  - sauvegarde et restauration du contexte d'une fonction (les registres du µP utilisés par la fonction)
  - utilisation des instructions conditionnelles
  - mise en œuvre de branchements et de boucles
  - appel et retour de sous-routines (fonctions/méthodes)
- de traduire en langage assembleur ARM du pseudo-code C

Traitement numérique des nombres

- d'interpréter correctement des valeurs entières dans une représentation en nombre signé et dans une représentation en nombre non-signé
- de prévoir correctement l'état des flags (Z, C, N, V) lors d'une opération arithmétique en assembleur
- d'utiliser correctement les branchements conditionnels en fonction des interprétations signées ou non signées
  - utilisation des mnémoniques pour les nombres non-signés (lo, ls, eq, ne, hs, hi)
  - utilisation des mnémoniques pour les nombres signés (lt, le, eq, ne, ge, gt)
- de représenter des valeurs réelles dans un format à virgule fixe et flottante et vice versa

Systèmes Embarqués 1 & 2

Objectifs du travail écrit n° 2

Interfaçage assembleur - C

- d'utiliser correctement la pile (stack) dans un programme codé en assembleur
- de programmer une sous-routine en assembleur ARM en utilisant la pile pour le passage de paramètres et pour les variables locales
- 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
- de représenter l'image de la pile lors de l'appel d'une sous-routine en assembleur
- de coder dans les règles de l'art l'appel de routines développées en C depuis un code en assembleur ARM
- de spécifier correctement l'appel de routines en assembleur depuis un code C

Programmation en C

- de décrire les concepts de programmation orientée objet en C
- de décrire l'héritage simple et multiple en C
- de concevoir des programmes orientés objet en C
- et tous les objectifs du premier travail écrit

Travail pratique TP.03

- de citer les caractéristiques principales (signaux et protocole) de l'interface de communication UART et de donner un exemple d'utilisation / d'application
- de citer les caractéristiques principales (signaux et protocole) de l'interface de communication I2C et de donner un exemple d'utilisation / d'application
- de citer les caractéristiques principales (signaux et protocole) de l'interface de communication SPI et de donner un exemple d'utilisation / d'application
- de décrire le principe de fonctionnement de l'écran LCD OLED
- de décrire le système de codage des couleurs RGB et plus particulièrement RGB565
- de décrire et concevoir les algorithmes permettant de dessiner des figures géométriques simples (carré, rectangle)
- de décrire et concevoir l'algorithme permettant de dessiner des caractères ASCII
- de décrire et concevoir l'algorithme permettant d'effectuer une rotation d'image

Systèmes Embarqués 1 & 2

Objectifs du travail écrit n° 2

Travail pratique TP.04

- de citer les composants principaux des horloges DMTimer du µP AM3358 de TI
- de décrire le principe de fonctionnement des horloges DMTimer
- de décrire et concevoir l'algorithme à mettre en place si l'on souhaite disposer d'une horloge permettant de compter sur plusieurs années avec une granularité de 1/24MHz, ceci sachant que le compteur des DMTimer ne permet de compter des intervalles que de 3 minutes environ.
- de décrire et concevoir un algorithme permettant de mesurer le temps d'exécution de parties de code ou de fonctions

Travail pratique TP.05

- de décrire et concevoir un algorithme récursif en assembleur

Le seul document à disposition est le « 05_ARM_Instruction_Set_Summary.pdf »
(jeu d'instructions du processeur ARM)