UNIVERSITÉ DE SHERBROOKE

Département d'informatique

IFT209 — Programmation système

Plan de cours

Enseignant :

Courriel :
Site du cours :
Disponibilité :

Professeur responsable :

Horaire :

Description officielle de l'activité pédagogique[1]

Objectif(s)	Comprendre l'architecture d'un ordinateur, les systèmes de numération, les types élémentaires de données, les structures de contrôle, les entrées-sorties; savoir effectuer une programmation-système.
Contenu	Introduction à l'architecture des ordinateurs. Système de numération. Modes d'adressage. Format des instructions machine. Représentation des données. Technique de mise au point de programmes. Arithmétique entière. Arithmétique à virgule flottante. Manipulation de bits. Sous-programmes. Application à une architecture contemporaine. Entrées-sorties. Traitement des interruptions.
Crédits	3
Organisation	3 heures d'exposé magistral par semaine 1 heure d'exercices par semaine 5 heures de travail personnel semaine
Préalable	IFT159

1 http://www.usherbrooke.ca/fiches-cours/ift209

1 Présentation

Cette section présente les objectifs spécifiques et le contenu détaillé de l'activité pédagogique. Cette section, non modifiable sans l'approbation d'un comité de programme du Département d'informatique, constitue la version officielle.

1.1 Mise en contexte

À propos des langages d'assemblage...

Lorsque les premiers ordinateurs sont apparus, vers la fin des années 1940, la programmation s'effectuait en langage machine, car à cette époque, les outils de programmation étaient rudimentaires ou inexistants. La programmation constituait une tâche très fastidieuse puisque toutes les instructions et les données devaient être codées manuellement à l'aide d'une suite de chiffres 0 et 1. La moindre erreur pouvait entraîner plusieurs jours de travail additionnels.

Dans le but de réduire le travail des programmeurs, les informaticiens de l'époque inventèrent des langages de programmation. Les premiers furent les langages d'assemblage qui existent encore de nos jours. Ils facilitent l'écriture de programmes en utilisant une notation symbolique dans laquelle les instructions machine, les données, ainsi que les adresses de la mémoire principale sont représentées par des symboles. Des modifications peuvent désormais être apportées aux programmes sans devoir les réécrire complètement, les suites de chiffres 0 et 1 étant remplacées par des symboles. Peu à peu, les langages d'assemblage ont fait place à des langages de plus en plus évolués qui ont contribué à une amélioration sensible de la production de programmes.

Même si aujourd'hui les programmeurs écrivent rarement des programmes en langage d'assemblage, il n'en demeure pas moins que ces derniers sont encore utiles. D'une part, ils servent principalement dans la construction de compilateurs ainsi que dans la mise en œuvre de systèmes d'exploitation et de systèmes embarqués. D'autre part, ils sont un outil précieux dans l'introduction et l'expérimentation des notions de base en architecture des ordinateurs. Enfin, ils permettent de mieux saisir les concepts fondamentaux des langages de programmation.

À propos des entrées/sorties et des interruptions...

De tout temps, la lenteur des entrées/sorties par rapport à la vitesse du processeur a posé le problème de l'interaction entre le processeur et les périphériques d'entrées/sorties. Une solution adéquate à ce problème nécessite la compréhension des mécanismes matériels et logiciels de synchronisation, en particulier le mécanisme d'interruption qui permet la gestion d'événements internes aux activités du processeur ou d'événements externes au processeur. La compréhension du mécanisme d'une interruption est indispensable à l'étude des principes sous-jacents aux systèmes d'exploitation.

À propos de la place de cette activité pédagogique dans votre programme...

Le cours de *Programmation système* est un cours de base. Il est préalable aux cours obligatoires suivants des programmes d'informatique et d'informatique de gestion :

IFT320 Systèmes d'exploitation
IFT585 Télématique

Il est également préalable, dans le programme d'informatique, pour les concentrations *sans concentration* et *systèmes et réseaux* aux cours à option suivants :

GEI201 Circuits logiques
GEI301 Architecture et organisation des ordinateurs

Sans en être un préalable, il permet d'aborder dans le programme d'informatique, pour toutes les concentrations, le cours à option suivant :

IFT580 Compilation et interprétation des langages

1.2 Objectifs spécifiques

À la fin du cours, l'étudiante ou l'étudiant sera capable de:

1. maîtriser l'arithmétique dans plusieurs systèmes de numération
2. connaître et expliquer des notions de base en architecture des ordinateurs;
3. connaître et expliquer des types élémentaires de données comme les entiers, les nombres en virgule flottante, les tableaux, les chaînes de bits, les chaînes de caractères;
4. connaître et expliquer des structures de contrôle comme la sélection, l'itération et les sous-programmes;
5. utiliser des méthodes élémentaires dans le développement de programmes;
6. comprendre les concepts élémentaires des langages procéduraux à travers ceux des langages d'assemblage;
7. écrire de petits programmes (100 à 350 instructions) dans un langage d'assemblage;
8. connaître et expliquer les opérations de lecture et d'écriture sur un périphérique d'entrées/sorties;
9. comprendre le mécanisme d'interruption et les différentes étapes de sa gestion.

1.3 Contenu détaillé

Le tableau suivant contient la matière présentée dans ce cours. L'étude de cette matière est accompagnée d'au moins six devoirs couvrant tous les sujets, dont au moins quatre travaux de programmation.

Thème	Contenu	Heures	Objectifs	Référence	Travaux
1	Introduction : présentation du plan de cours; présentation de la programmation en langage d'assemblage	2			
2	Systèmes de numération : écriture des nombres dans un système de numération; conversion de nombres d'un système de numération en un autre.	2	1	chap. 1 de [3]	T
3	Architecture des ordinateurs: architecture von Neumann; mémoire principale, processeur, registres; jeux d'instructions; aspects particuliers à l'organisation d'un ordinateur; processeur *UltraSPARC*.	4	2	chap. 2 de [3]	
4	Accès aux données: données, adresses, modes d'adressage; étapes de la vie d'un programme.	4	2	chap. 3 de [3]	
5	Programmation en langage d'assemblage: survol à partir d'un petit programme; différence entre une instruction et une pseudo-instruction; programmation de haut niveau des entrées/sorties.	4	3,5,6,7	chap 4 de [3]	T
6	Les nombres entiers: représentations des entiers signés et non signés; report et débordement; instructions arithmétiques.	4	3,6,7	chap 5 de [3]	T
7	Tableaux: tableaux à une dimension et à deux dimensions.	2	3,6,7	chap 6 de [3]	T
8	Structures de contrôle : condition, branchements, séquence, sélection, itération; appel et retour de sous-programmes; notion d'environnement; récursivité.	4	4,6,7	chap 7 de [3]	T
9	Chaînes de bits:	2	3,6,7	chap 8 de [3]	T

	opérations logiques, décalages.				
10	Chaînes de caractères: codes de représentation de caractères; opérations sur les caractères et les chaînes de caractères.	2	3,6,7	chap 9 de [3]	T
11	Sous-programmes : appel et retour, passage de paramètres, sauvegarde/récupération, récursivité.	4	4,6,7	chap 10 de [3]	T
12	Les nombres en virgule flottante: représentations des nombres en virgule flottante; erreur d'arrondi, erreur de troncation, dépassement de capacité; norme IEEE 754; instructions arithmétiques.	3	3,6,7	chap 11 de [3]	T
12	La programmation des entrées/sorties: interrogation de bits d'états, interruption, niveaux de priorités des interruptions; illustration à l'aide de dispositifs simples.	10	8,9		

Le cours doit comprendre au moins cinq travaux pratiques couvrant tous les sujets marqués d'un T dans le tableau.

2 Organisation

Cette section propre à l'approche pédagogique de chaque enseignante ou enseignant présente la méthode pédagogique, le calendrier, le barème et la procédure d'évaluation ainsi que l'échéancier des travaux. Cette section doit être cohérente avec le contenu de la section précédente.

2.1 Méthode pédagogique

Une semaine comporte quatre heures de présence en classe réparties dans une proportion de trois heures de cours magistral et d'une heure d'exercices. Les exercices sont faits par les étudiantes ou les étudiants, autant que possible à la fin d'un thème. Ainsi, dans une semaine, il est possible qu'il n'y ait aucune séance d'exercices, mais que dans la semaine suivante il y ait une séance d'exercices de deux heures. Le manuel de cours [1] couvre la totalité de la matière, à l'exception des deux derniers thèmes. On s'attend à ce que les étudiantes et les étudiants aient lu chaque chapitre avant que la matière correspondante soit abordée en classe. On s'attend aussi à une participation active des étudiantes ou des étudiants lors de la présentation de la matière. Toutes les notions théoriques relatives aux différents thèmes sont illustrées à l'aide d'exemples construits à partir de l'architecture SPARC version 8 et de programmes en langage d'assemblage SPARC version 8. De plus, l'architecture NES est utilisée pour illustrer la programmation des entrées/sorties et des interruptions.

2.2 Calendrier du cours

Semaine	Date	Thème	Lecture	Exercices
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				

2.3 Évaluation

Devoirs (6 x 5%) :
Examen périodique:
Examen final:

Les appareils électroniques (calculatrice, portable, téléphone cellulaire) sont interdits durant les examens.

Conformément aux articles 36, 37 et 38 du règlement facultaire d'évaluation des apprentissages[2] l'enseignant peut retourner à l'étudiante ou à l'étudiant tout travail non conforme aux exigences quant `a la qualité de la langue et aux normes de présentation.

[2] http://www.usherbrooke.ca/accueil/documents/politiques/pol 2500-008/pol evaluation/sciences.html

Toute situation de plagiat sera traitée en conformité, entre autres, avec l'article 8.1.2 du Règlement des études[3] de l'Université de Sherbrooke.

2.4 Échéancier des travaux

TP	Réception	Thème	Remise
1			
2			
3			
4			
5			
6			

3 Matériel nécessaire pour le cours

Le manuel de M. St-Denis [3] est obligatoire. L'architecture SPARC version 8 est entièrement décrite dans la référence [2]. Tous les programmes et les diapositives présentés dans le cours sont disponibles dans le répertoire /home/public/cours/ift209.
Les programmes du manuel de cours [1], traduits dans la version 8 de l'architecture SPARC sont également disponibles dans le répertoire /home/public/cours/ift209.

4 Références

[1] Outils de développement pour modules ARM, http://www.siwawi.arubi.uni-kl.de/avr_projects/arm_projects/index.html

[2[SPARC International, The SPARC Architecture Manual, version 9, Prentice-Hall, 1994 (1 exemplaire à la bibliothèque)

[3] R.St-Denis, *La programmation en langage d'assemblage SPARC,* éditions GGC, 1998 (2 exemplaires à la bibliothèque, mis en réserve).

[4] SPARC International, *The SPARC Architecture Manual, version 8,* Prentice-Hall, 1992 (2 exemplaires à la bibliothèque).

[5] R.P. Paul, *Sparc Architecture, Assembly Language Programming, & C*, Prentice-Hall, seconde édition, 2000 (1exemplaire plus 1 exemplaire de la première édition (1994) à la bibliothèque).

[3] http://www.usherbrooke.ca/programmes/etude