Université du Québec en Outaouais **Département d'informatique et d'ingénierie**

Sigle : INF1633 Gr. 01
Titre : Programmation de systèmes embarqués en C/C++
Session : Automne 2026 Horaire et local
Professeur : Bennai, Mustapha

1. Description du cours paraissant à l'annuaire :

Objectifs

Au terme de ce cours l'étudiant.e aura maîtrisé, par la pratique, la programmation en langage C/C++ des systèmes informatiques embarqués.

Contenu

Introduction au domaine des systèmes embarqués. Aspects matériels : architecture RISC et microcontrôleurs ARM et ATMEGA. Langage C/C++. Chaîne de compilation GNU. Environnements de programmation. Développement et intégration d'applications dans des environnements embarqués. Techniques de débogage. Introduction aux systèmes d'exploitation temps réel (RTOS). Réalisation d'un projet de système embarqué. Ce cours comporte des séances obligatoires de travaux dirigés (TD).

Descriptif – Annuaire

2. Objectifs spécifiques du cours :

Au terme du cours, l'étudiant.e devra être capable de concevoir, programmer, déboguer et valider une application embarquée en C/C++ sur la plateforme FRDM-KL28Z, en reliant systématiquement le code au comportement observable du matériel.
• Utiliser MCUXpresso IDE, le SDK NXP, OpenSDA, le terminal série et le débogueur pour construire, programmer et observer une application sur le MKL28Z512VLL7.
• Écrire un code C lisible, documenté et modulaire, en choisissant des types adaptés et en organisant les interfaces et implémentations dans des fichiers .h/.c.
• Comprendre et utiliser la mémoire, l'adressage, les pointeurs, les structures, const et volatile dans le contexte des API et pilotes embarqués.
• Configurer et programmer des périphériques et interfaces : GPIO, RTC/timers, interruptions, I²C, UART et liaison XBee.
• Utiliser les mécanismes C++ utiles à l'embarqué — classes, encapsulation, héritage et polymorphisme — en comprenant leur coût et leur pertinence.
• Structurer une application simple avec FreeRTOS à l'aide de tâches, priorités, états et délais, sans ajouter de mécanismes de synchronisation inutiles.
• Diagnostiquer les erreurs par couches et valider chaque sous-système par des observations reproductibles sur la cible.
• Intégrer progressivement des modules déjà validés afin de réaliser et démontrer un petit système embarqué complet.

3. Stratégies pédagogiques :

Le cours se donne dans une salle de laboratoire. Chaque séance de 3 heures alterne des explications théoriques, des démonstrations en direct sur la FRDM-KL28Z et de courtes manipulations guidées. Chaque notion C/C++ est reliée à un problème embarqué concret, à une action sur la cible et à une méthode d'observation ou de validation.
La plateforme de référence est la carte NXP FRDM-KL28Z équipée du microcontrôleur MKL28Z512VLL7. Les activités utilisent MCUXpresso IDE, le SDK NXP, OpenSDA, le terminal série et le débogueur. Le matériel pédagogique, les cahiers de notes, les consignes et les ressources sont accessibles sur Moodle; un forum de discussion y est aussi disponible.
Les séances de cours prévues les semaines 4, 10 et 12 se dérouleront à distance, par vidéoconférence Zoom. Le lien sera disponible sur le site du cours sur Moodle.
Cinq travaux dirigés (TD), non obligatoires et réalisés sous forme de travaux pratiques guidés en laboratoire, consolident la matière sur la même plateforme : prise en main et débogage, RTC, I²C/BMP180, UART/XBee et introduction à FreeRTOS. Le projet intégrateur réutilise ensuite ces sous-systèmes dans une application unique et les valide de manière incrémentale.

4. Heures de disponibilité ou modalités pour rendez-vous :

Communication par courriel et via le forum de discussion sur Moodle.
- Assistant à l'enseignement : Bekobo Nguedi, Ariel Toussaint - beka07@uqo.ca

- Responsable des laboratoires : Abdallah Guire Ali - alixab01@uqo.ca
- Technicien de laboratoire : Abdelkrim Chebihi - abdelkrim.chebihi@uqo.ca.
- Enseignant : Mustapha Bennai – mustapha.bennai@uqo.ca

5. Plan détaillé du cours sur 15 semaines :

Semaine	Thèmes	Dates
1	**Chapitre 1 — Comprendre et programmer un premier système embarqué sur FRDM-KL28Z** • Présentation du cours et du contrat pédagogique. • Système embarqué, MKL28Z512VLL7, mémoire et périphériques. • MCUXpresso, SDK, OpenSDA : Build → Flash → Run → Debug.	10 sept. 2026
2	**Chapitre 2 — Programmer le comportement d'un système embarqué en C** • Types, variables, constantes et opérateurs utiles. • Décisions, boucles et fonctions. • Organisation modulaire .c/.h et préprocesseur.	17 sept. 2026
3	**Chapitre 3 — Mémoire, adressage, pointeurs et structures en C** • Flash/SRAM, pile et notions de tas. • Passage par adresse, structures, const et volatile.	24 sept. 2026
4	**Cours à distance** **Chapitre 4 — GPIO et programmation du matériel** • Broches, multiplexage, entrées/sorties et logique active basse. • Registres, masques de bits et API du SDK.	01 oct. 2026
5	**Chapitre 5 — Temps, timers et horloge temps réel (RTC)** • Fréquence, période, tick, prescaler et temporisation. • Configuration et lecture de la RTC avec le SDK NXP.	08 oct. 2026
6	**Semaine d'études**	12 - 16 oct. 2026
7	**Examen de mi-session**	22 oct. 2026
8	**Chapitre 6 — Interruptions, NVIC et programmation événementielle** • Polling versus interruption; événement, IRQ, NVIC et ISR. • Priorités, acquittement et données partagées simples avec volatile.	29 oct. 2026
9	**Chapitre 7 — Communication avec un périphérique externe : bus I²C** • SDA/SCL, adresses, registres, START/STOP et ACK/NACK. • Lecture et validation du capteur BMP180 par LPI2C. **Publication de l'énoncé du projet intégrateur.**	05 nov. 2026
10	**Cours à distance** **Chapitre 8 — Communication série et sans fil : UART, XBee et IEEE 802.15.4** • UART2, trame série 8-N-1 et diagnostic par couches. • XBee, mode transparent, configuration et validation avec XCTU.	12 nov. 2026
11	**Chapitre 9 — Conception embarquée en C++ : classes, objets et encapsulation** • Du module C à la classe C++. • public/private, constructeur, organisation .hpp/.cpp et appel d'API C.	19 nov. 2026

12	**Cours à distance** **Chapitre 10 — Multitâche avec FreeRTOS : tâches, ordonnanceur, états et délais** • Tâches, piles et états Running/Ready/Blocked. • xTaskCreate, vTaskDelay, priorités et observation dans la Task List.	26 nov. 2026
13	**Chapitre 11 — Abstraction et réutilisation en C++ embarqué** • Surcharge, héritage, virtual/override et classes abstraites. • Polymorphisme : utilité, limites et coût mesurable.	03 déc. 2026
14	**Chapitre 12 — Architecture logicielle, intégration et validation d'un système embarqué** • Modules, interfaces, ressources et budgets Flash/RAM/CPU. • Intégration incrémentale, baseline et diagnostic par couches. • Présentation et démonstration du projet : mini station environnementale connectée.	10 déc. 2026
15	**Examen final**	17 déc. 2026

5.1 Calendrier des travaux dirigés

TD	Contenu et dépendance pédagogique	Dates selon le groupe
TD1	**FRDM-KL28Z, MCUXpresso, OpenSDA, console série et débogage.** le TD approfondit la chaîne Build → Flash → Run → Debug déjà expliquée en cours.	Lundi 14 sept. 2026 Mercredi 16 sept. 2026
TD2	**Horloge temps réel RTC.** Prépare la transition vers les interruptions.	Lundi 26 oct. 2026 Mercredi 28 oct. 2026
TD3	**I²C et capteur BMP180.** Réutilisation de l'adressage, les buffers et le pilote LPI2C sur un périphérique externe réel.	Lundi 9 nov. 2026 Mercredi 11 nov. 2026
TD4	**UART2 et XBee.** Configuration série, mode transparent, XCTU et diagnostic par couches.	Lundi 16 nov. 2026 Mercredi 18 nov. 2026
TD5	**Introduction à FreeRTOS.** Réinvestir les tâches et vTaskDelay() dans la station environnementale.	Lundi 30 nov. 2026 Mercredi 2 déc. 2026

6. Évaluation du cours :

Dans le cas spécifique du cours Programmation de systèmes embarqués en C/C++, l'attribution des notes se fera selon la répartition suivante :

- **Examen de mi-session : 30 %**
- **Examen final : 40 %**
- **Projet intégrateur — mini station environnementale connectée : 30 %**

7. Politiques départementales et institutionnelles :

- Politique du département d'informatique et d'ingénierie relative à la tenue des examens
- Note sur le plagiat et sur la fraude
- Politique relative à la qualité de l'expression française écrite chez les étudiants et les étudiantes de premier cycle à l'UQO

- Absence aux examens : cadre de gestion, demande de reprise d'examen (formulaire)

Tolérance ZÉRO en matière de violence à caractère sexuel.

Le Bureau d'intervention et de prévention en matière de harcèlement (BIPH) a pour mission d'accueillir, soutenir et guider toute personne vivant une situation de harcèlement, de discrimination ou de violence à caractère sexuel. Le BIPH oriente ses actions afin de prévenir les violences à caractère sexuel pour que nous puissions étudier, travailler et s'épanouir dans un milieu sain et sécuritaire.

Vous vivez ou êtes une personne témoin d'une situation de violence à caractère sexuel ? Vous êtes une personne membre de la communauté étudiante ou une personne membre du personnel, autant à Gatineau qu'à Ripon et St-Jérôme, l'équipe du BIPH est là pour vous, sans jugement et en toute confidentialité.

Ensemble, participons à une culture de respect.

Pour de plus amples renseignements consultez UQO.ca/biph ou écrivez-nous au Biph@uqo.ca

8. Principales références :

- Documents, cahiers de notes, travaux dirigés et guides disponibles sur le site Moodle du cours.
- NXP Semiconductors. FRDM-KL28Z — FRDM Development Platform for Kinetis KL28 MCUs.
- NXP Semiconductors. MKL28Z512VXX7 — Kinetis KL28 Data Sheet; MKL28ZRM — Kinetis KL28 Reference Manual.
- NXP Semiconductors. MCUXpresso IDE User Guide et documentation du MCUXpresso SDK pour FRDM-KL28Z.
- FreeRTOS. FreeRTOS Kernel Documentation et Mastering the FreeRTOS Real Time Kernel.
- Michael Barr et Anthony Massa. Programming Embedded Systems, 2e éd., O'Reilly Media.
- Documentation Digi International : modules XBee et logiciel XCTU, utilisée pour la liaison série et radio du cours.

9. Page Web du cours :

https://moodle.uqo.ca