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

Sigle : GEN2003 Gr. 01
Titre : Ingénierie de systèmes embarqués
Session : Automne 2026 Horaire et local
Professeur.e : Bennai, Mustapha

1. Description du cours paraissant à l'annuaire :

Objectifs

Au terme de ce cours, la personne étudiante sera en mesure de/d' : - Développer des applications embarquées en mettant l'accent sur la programmation de firmware ; - Concevoir et implémenter des pilotes pour l'intégration avec le système d'exploitation Linux ; - Assurer une interaction efficace entre les composants matériels et logiciels.

Contenu

Systèmes embarqués et domaines d'application. Programmation embarquée : temps de traitement, interruptions, manipulation directe des registres. Alimentation et consommation énergétique. Architecture des microcontrôleurs : architecture RISC et microcontrôleurs ARM et AVR. Interfaces de communication (UART, SPI, I2C, etc.). Utilisation des périphériques embarqués (GPIO, ADC, PWM, etc.) pour interfacer des capteurs et contrôler des actionneurs. Programmation des firmwares : architecture, bootloader, noyau, gestion de la mémoire et optimisation des performances. Création de pilotes sous Linux pour l'intégration de capteurs et de matériel embarqué. Interaction avec le noyau Linux et l'infrastructure de gestion des périphériques. Compilation croisée et configuration du noyau Linux pour les systèmes embarqués. Ce cours comporte des séances obligatoires de travaux pratiques (TP) de 3h par semaine avec une attention particulière à l'interaction matérielle et logicielle.

Descriptif – Annuaire

2. Objectifs spécifiques du cours :

Au terme de ce cours, la personne étudiante sera en mesure de/d' :

- Développer des applications embarquées en mettant l'accent sur la programmation de Micrologiciel ;
- Concevoir et implémenter des pilotes pour l'intégration avec le système d'exploitation Linux ;
- Assurer une interaction efficace entre les composants matériels et logiciels.

Le cours couvre 2 des 12 qualités requises des diplômé(e)s telles que définies dans les normes d'agrément des programmes de génie au Canada (http://www.engineerscanada.ca/fr/ressources-en-matiere-dagrement) :

a. **Qualité 1 : Connaissances en génie**
b. **Qualité 4 : Conception**

Toutes les qualités énumérées ci-dessus sont mesurées dans ce cours pour fins de rétroaction.

Objectifs spécifiques	Qualité	Indicateurs	Introduit	Développé	Appliqué
Connaissance, à un niveau universitaire, des mathématiques, des sciences naturelles et des notions fondamentales de l'ingénierie, ainsi qu'une spécialisation en génie propre au programme.	1	4. Comprendre et appliquer les concepts de l'ingénierie propres au programme.		X	

La conception en ingénierie est un processus consistant à prendre des décisions éclairées pour concevoir de façon créative un produit, un système, un composant ou un procédé devant répondre à des besoins précisés, en tirant parti de l'analyse et du jugement de l'ingénierie. Ce processus est souvent caractérisé comme étant complexe, évolutif, itératif et multidisciplinaire. Les solutions qui en sont issues font appel aux sciences naturelles, aux mathématiques et aux sciences du génie, ainsi qu'à des pratiques systématiques et exemplaires actuelles afin de satisfaire à des objectifs définis, dans le respect des exigences, des normes et des contraintes établies. Parmi les contraintes à prendre en considération, citons la santé et la sécurité, la durabilité, l'environnement, l'éthique, la sûreté, l'économie, les facteurs esthétiques et humains, la faisabilité et la conformité aux aspects réglementaires, de même que des enjeux universels en matière de conception, comme les aspects sociaux, culturels et de diversification.	4	3. Créer des modèles, simulations, prototypes, et faire des tests.	X		
		4. Vérifier la conformité de la conception par rapport au cahier des charges.	X		

3. Stratégies pédagogiques :

Le cours adopte une approche intégrée reliant les fondements de l'ingénierie des systèmes embarqués, la programmation du micrologiciel, l'exploitation des périphériques, l'intégration avec un système Linux embarqué et la validation expérimentale du fonctionnement matériel et logiciel.

1. Mode de prestation

- Le cours sera offert en mode hybride. Les séances de cours prévues les mercredis des semaines 4, 10 et 12 se dérouleront à distance, par vidéoconférence, tandis que les séances de travaux pratiques, tenues les vendredis, demeureront en présentiel. Les liens de connexion aux séances à distance seront diffusés sur Moodle.
- Les travaux pratiques et les examens auront lieu en présentiel.
- Le cours comprend une séance de cours de trois heures et une plage hebdomadaire de laboratoire de trois heures. Des séances de travaux pratiques évaluées et obligatoires sont prévues selon le calendrier détaillé du cours.

2. Approche pédagogique

- Exposés structurés portant sur les architectures embarquées, la programmation des microcontrôleurs, les systèmes d'exploitation embarqués et les pilotes Linux.
- Démonstrations techniques réalisées sur la plateforme Arduino UNO Q.
- Exercices dirigés portant sur la programmation bas niveau, les interruptions, les interfaces de communication, la gestion des périphériques et l'intégration matérielle-logicielle.
- Analyse de fiches techniques, de schémas électriques, de traces d'exécution et de résultats de mesure.
- Études de cas provenant notamment des domaines industriel, automobile, médical, domotique et de l'Internet des objets.
- Apprentissage progressif allant de la programmation du microcontrôleur et des périphériques vers Zephyr, Debian Linux et le développement d'un pilote simple.
- Validation des implantations au moyen de mesures, de traces, de scénarios de test et de critères d'acceptation définis.

3. Plateforme technologique

- La plateforme principale du cours est l'Arduino UNO Q.
- Le microcontrôleur STM32U585 est utilisé pour la programmation du micrologiciel, la gestion des entrées-sorties, les interruptions, les interfaces de communication et les applications sous Zephyr.
- Le processeur QRB2210 fonctionnant sous Debian Linux est utilisé pour l'étude du démarrage, des processus, des services, des modules noyau et des pilotes de périphériques.

- Le capteur numérique MCP9808 est utilisé pour l'acquisition et l'interprétation de données transmises par l'interface I²C.
- Les outils logiciels, bibliothèques, paquets et versions utilisés au laboratoire seront figés et validés avant le début du trimestre.

4. Travaux pratiques

- Le trimestre comprend cinq travaux pratiques évalués et obligatoires.
- Chaque séance de TP est précédée d'un travail préparatoire comprenant notamment la lecture de documents techniques, l'analyse du problème et la préparation de certaines parties du programme.
- Les TP mettent l'accent sur la mesure, le diagnostic, la validation expérimentale et la justification des choix techniques. Chaque TP explicite les exigences, un modèle, un prototype ou un choix de conception à justifier, les essais et les critères d'acceptation utilisés pour vérifier la conformité au cahier des charges.
- La progression des travaux pratiques est la suivante :

TP	Titre	Compétences dominantes	Production attendue
TP1	GPIO, chronométrage et programmation non bloquante	C1, C2, C4 et C7	Sketch fonctionnel, traces série, chronogramme et validation.
TP2	Interruptions et mesure de pertes d'événements	C2, C7	Deux versions comparées, mesures et analyse quantitative.
TP3	Capteur MCP9808, I2C, UART et PWM	C3, C7	Pilote I2C minimal, mesure convertie, sortie PWM et diagnostic.
TP4	Architecture multitâche avec Zephyr	C4, C7	Application Zephyr, diagramme des threads, états et traces de défaut.
TP5	Module noyau et pilote caractère	C5, C6, C7	Module, Kbuild, application utilisateur, traces et rapport de validation.

- Les TP sont réalisés en binôme fixe. Un trinôme peut exceptionnellement être autorisé lorsque l'effectif du groupe l'exige.
- Les rapports de TP doivent être remis, individuellement, dans les sept jours suivant la séance. Une pénalité de 20 % par jour de retard s'applique.
- Une absence non justifiée à une séance obligatoire entraîne la note zéro pour l'activité concernée.
- Chaque membre d'une équipe doit être en mesure d'expliquer individuellement le fonctionnement du matériel, du programme et des méthodes de validation présentés dans le rapport.

5. Compétences visées :

Au terme des activités d'apprentissage et des travaux pratiques, la personne étudiante devra être en mesure de :

Code	Compétence mesurable
C1	Décomposer une architecture embarquée en composants matériels, micrologiciels, système et application.
C2	Programmer des entrées-sorties, des interruptions et des traitements temporels sur le STM32U585.
C3	Acquérir et interpréter une donnée I2C, puis commander une sortie PWM.
C4	Concevoir un micrologiciel non bloquant et une architecture multitâche sous Zephyr.
C5	Expliquer la chaîne de démarrage et l'organisation d'un Linux embarqué.
C6	Compiler, charger et tester un pilote caractère simple et son application utilisateur.
C7	Appliquer une démarche de validation fondée sur des mesures, des traces et des critères d'acceptation.

6. Travail personnel

- La personne étudiante doit prévoir du temps chaque semaine pour la révision des notions théoriques, la réalisation des exercices, la lecture des documents techniques et la préparation des séances de laboratoire.
- Les travaux préparatoires doivent être réalisés avant la séance de TP correspondante.
- Les programmes, mesures, résultats et analyses doivent être conservés de manière structurée afin de faciliter la rédaction des rapports et la validation des travaux.

7. Intégrité académique et usage des outils numériques

- Toute source externe utilisée dans un travail, notamment du code, des données, des schémas, des bibliothèques ou de la documentation technique, doit être déclarée et citée.
- Les contenus générés ou modifiés à l'aide d'outils d'intelligence artificielle doivent être déclarés conformément aux directives du cours et aux politiques institutionnelles.
- La personne étudiante demeure responsable de l'exactitude, de la sécurité et du fonctionnement de tout code ou contenu soumis.
- Chaque personne étudiante doit être en mesure d'expliquer et de défendre individuellement le contenu de ses travaux.

8. Ressources et communication

- Moodle est utilisé pour diffuser les notes de cours, les énoncés, les fichiers de départ, les travaux préparatoires, les consignes de laboratoire, les avis et les résultats.
- Les travaux et rapports doivent être remis au moyen des espaces de dépôt prévus sur Moodle.
- La plateforme Moodle doit être consultée régulièrement. Cette responsabilité incombe à la personne étudiante.

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

- Les disponibilités sont offertes sur rendez-vous à distance, selon une plage convenue avec l'enseignant.
- Les demandes de rencontre doivent être transmises par courriel institutionnel ou par la messagerie Moodle. Les rencontres auront normalement lieu par Microsoft Teams.
- Les questions générales relatives au contenu du cours, aux travaux préparatoires ou aux travaux pratiques doivent d'abord être publiées dans le forum Moodle afin que les réponses puissent bénéficier à l'ensemble du groupe.
- Assistante à l'enseignement : Sabrina Mefo Fotso - mefs02@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.** Introduction aux systèmes embarqués : domaines d'application, contraintes de temps, mémoire, énergie, coût, fiabilité et sécurité; architecture hybride MCU/MPU de l'Arduino UNO Q; cycle exigences–architecture–implantation–validation.	8 au 11 sept. 2026
2	• **Chapitre 2.** Microcontrôleur et programmation bas niveau : Cortex-M33, Flash/SRAM, registres mappés en mémoire, pile, démarrage, compilation Arduino sur Zephyr, GPIO et chronométrage.	14 au 18 sept. 2026
3	• **Chapitre 2 (suite).** Interruptions et comportement temporel : latence, variables partagées, volatile, traitement différé, programmation non bloquante et mesure des pertes d'événements. • Préparation à la séance de TP1.	21 au 25 sept. 2026
4	• **Chapitre 3.** Périphériques embarqués : GPIO, ADC et PWM; contraintes électriques 3,3 V; échantillonnage, quantification, conversion et commande d'actionneurs. • **Séance de TP1 - Vendredi 02 oct. 2026**: GPIO, chronométrage et programmation non bloquante. • **SÉANCE DE COURS EN MODE NON PRÉSENTIEL.**	28 sept. au 2 oct. 2026

5	• **Chapitre 3 (suite).** Interfaces UART, I2C et SPI : structure des échanges, adressage, registres, START/STOP, ACK/NACK et diagnostic des erreurs. Intégration du capteur MCP9808. • Préparation à la séance de TP2.	5 au 9 oct. 2026
6	• **Semaine d'études : aucune activité d'enseignement.**	12 au 16 oct. 2026
7	• **Examen intra portant sur les contenus des chapitres 1 à 3.**	21 oct. 2026
8	• **Chapitre 4.** Architecture du micrologiciel : couches logicielles, pilotes, services, logique applicative, Super boucle non bloquante, événements, machines à états et stratégie de diagnostic. • **Séance de TP2 – Vendredi 30 octobre 2026** : Interruptions et mesure de pertes d'événements.	26 au 30 oct. 2026
9	• **Chapitre 4 (suite).** Zephyr RTOS : threads, priorités, temporisations, sémaphores, files de messages, mutex et surveillance logicielle. • Préparation à la séance de TP3.	2 au 6 nov. 2026
10	• **Chapitre 5.** Linux embarqué sur le QRB2210 : Debian, processus, fichiers, services, distinction hôte/cible, compilation native et croisée, ABI, bibliothèques et sysroot. • **SÉANCE DE COURS EN MODE NON PRÉSENTIEL.** • **Séance de TP3 - Vendredi 13 nov. 2026** : Capteur MCP9808, I2C, UART et PWM.	9 au 13 nov. 2026
11	• **Chapitre 5 (suite).** Démarrage et organisation d'un Linux embarqué : Boot ROM, chargeur, noyau, Device Tree, système de fichiers, services, /dev, /proc et /sys; comparaison des rôles de Buildroot et Yocto. • Préparation à la séance de TP4.	16 au 20 nov. 2026
12	• **Chapitre 6.** Espace utilisateur et espace noyau : Kbuild, cycle de vie d'un module, printk, dmesg, règles de sécurité et préparation de l'environnement du pilote. • **SÉANCE DE COURS EN MODE NON PRÉSENTIEL.** • **Séance de TP4 – Vendredi 27 novembre 2026** : Architecture multitâche avec Zephyr.	23 au 27 nov. 2026
13	• **Chapitre 6 (suite).** Pilote caractère Linux : numéros majeur et mineur, file_operations, open, read, write, release, copy_to_user et copy_from_user; application utilisateur. • Préparation à la séance de TP5.	30 nov. au 4 déc. 2026
14	• **Chapitre 6 (synthèse).** Modèle de périphériques Linux : platform_driver, Device Tree, GPIO et interruptions; intégration matérielle-logicielle, diagnostic, validation et synthèse du cours. • **Séance de TP5 - Vendredi 11 décembre 2026** : module noyau et pilote caractère.	7 au 11 déc. 2026
15	• **Examen final portant sur tous les chapitres (1 à 6).**	16 déc. 2026

6. Évaluation du cours :

La note finale est établie selon la pondération suivante :

Évaluation	Pondération	Portée	Indicateurs évalués
TP1	4 %	Prise en main, GPIO, chronométrage et méthode de validation.	4.3 et 4.4
TP2	5 %	Interruptions, variables partagées et analyse de pertes.	4.3 et 4.4
TP3	6 %	I2C, conversion, communication et commande.	4.3 et 4.4
TP4	7 %	Architecture micrologicielle, Zephyr et mécanismes temps réel.	4.3 et 4.4
TP5	8 %	Linux, module noyau, pilote caractère et application.	4.3 et 4.4

Examen intra	30 %	Chapitres 1 à 3.	1.4
Examen final	40 %	Chapitres 1 à 6; Note : À partir d'un cahier des charges, une question intégratrice de l'examen final demandera de proposer et justifier un modèle, une architecture ou un prototype, d'en définir les tests et critères d'acceptation, puis d'en évaluer la conformité.	1.4, 4.3 et 4.4

Les consignes détaillées, critères de correction, modalités de remise et règles applicables aux évaluations seront diffusés sur Moodle. Les séances de TP sont obligatoires conformément au descriptif officiel du cours.

Par **indicateur mesuré**, on entend qu'à la fin du cours, un niveau de performance (0, 1, 2, 3) est donné pour chaque indicateur et pour chaque étudiant(e) selon la grille ci-dessous.

Indicateurs	**Niveau 0**	**Niveau 1**	**Niveau 2**	**Niveau 3**
1.4 – Comprendre et appliquer les concepts de l'ingénierie propres au programme.	Moins de 52 %	Entre 52 % et 63 %	Entre 64 % et 83 %	Plus de 84 %
4.3 – Créer des modèles, simulations, prototypes, et faire des tests.	Création de modèles, simulations, prototypes et/ou exécution de tests inadéquate ou inexistante	Création acceptable de modèles, simulations, prototypes, mais exécution de tests insuffisante	Création de modèles, simulations, prototypes et exécution de tests adéquates	Création de modèles, simulations, prototypes et exécution de tests remarquables
4.4 – Vérifier la conformité de la conception par rapport au cahier des charges.	Vérification inadéquate ou inexistante	Vérification partielle	Vérification acceptable	Vérification exhaustive

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 :

Chapitre	Références recommandées
1-6	Notes de cours, Mustapha Bennai, 2026.
1	• White, Elecia. Making Embedded Systems: Design Patterns for Great Software. 2nd ed., O'Reilly Media, 2024; • Ünsalan et al., Embedded System Design with ARM Cortex-M Microcontrollers, Springer, 2022; documentation officielle Arduino UNO Q.
2	• Ünsalan et al., 2022; Seacord, Robert C., Effective C, 2nd ed., No Starch Press, 2024; • documentation STM32U585, Zephyr et ArduinoCore-zephyr.
3	• Ünsalan et al., 2022; • White, 2024; • documentation Arduino Wire; • fiche technique officielle Microchip MCP9808.
4	• White, 2024; • Wang, K. C., Embedded and Real-Time Operating Systems, 2nd ed., Springer, 2023; • documentation officielle Zephyr RTOS.
5	• Vasquez, Frank et Chris Simmonds, Mastering Embedded Linux Development, 4th ed., Packt, 2025; documentation Arduino Debian pour UNO Q; • Linux Kernel Documentation.
6	• Madieu, John, Linux Device Driver Development, 2nd ed., Packt, 2022; • Vasquez et Simmonds, 2025; • Linux Kernel Driver API Documentation.

Arduino UNO Q, Zephyr RTOS, MCP9808 et noyau Linux

Organisation	Référence technique	Utilité dans le cours	Lien direct
Arduino	Arduino UNO Q - documentation matérielle	Page officielle regroupant le manuel, le brochage, la fiche technique, les schémas et les fichiers CAO.	Ouvrir la ressource officielle
Arduino	Arduino UNO Q - fiche technique (PDF)	Caractéristiques matérielles, architecture MPU/MCU, alimentation, interfaces et brochage.	Ouvrir la ressource officielle
Arduino	Arduino UNO Q - brochage complet (PDF)	Affectation détaillée des connecteurs et signaux de la carte.	Ouvrir la ressource officielle
Arduino	Arduino UNO Q - schémas électriques (PDF)	Schémas officiels de la plateforme.	Ouvrir la ressource officielle
Arduino	ArduinoCore-zephyr - dépôt officiel	Code source, instructions d'installation, historique et documentation du cœur Arduino basé sur Zephyr.	Ouvrir la ressource officielle
Arduino	ArduinoCore-zephyr 0.56.0 - publication	Version 0.56.0 et notes de publication officielles.	Ouvrir la ressource officielle
Arduino	Arduino IDE 2 - documentation	Installation, gestionnaire de cartes, téléversement, moniteur série et débogage.	Ouvrir la ressource officielle
Arduino	Arduino CLI - documentation	Compilation, gestion des cartes et bibliothèques, automatisation et intégration en ligne de commande.	Ouvrir la ressource officielle

Zephyr Project	Arduino UNO Q - documentation de la carte	Cible Zephyr arduino_uno_q, présentation de la carte et prise en charge du STM32U585.	Ouvrir la ressource officielle
Zephyr Project	Zephyr RTOS - documentation générale et API	Guides, services du noyau, pilotes, synchronisation, ordonnancement et API officielles.	Ouvrir la ressource officielle
Zephyr Project	Arduino Core API sous Zephyr	Documentation de l'intégration de l'API Arduino dans l'environnement Zephyr.	Ouvrir la ressource officielle
Microchip Technology	MCP9808 - page officielle du composant	Informations produit, état de production et accès à la documentation technique.	Ouvrir la ressource officielle
Microchip Technology	MCP9808 - fiche technique officielle (PDF)	Registres, adressage I2C, résolution, conversion des données et limites de précision.	Ouvrir la ressource officielle
Linux Kernel Development Community	Documentation officielle du noyau Linux	Documentation générale du noyau, développement, interfaces et sous-systèmes.	Ouvrir la ressource officielle
Linux Kernel Development Community	Driver Implementer's API Guide	API et conventions utilisées pour la conception de pilotes Linux.	Ouvrir la ressource officielle
Linux Kernel Development Community	Linux Driver Model	Modèle de périphériques, bus, classes, pilotes et gestion des objets du noyau.	Ouvrir la ressource officielle

9. Page Web du cours :

https://moodle.uqo.ca