



Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»



Кафедра інженерії
програмного
забезпечення в енергетиці
(ІПЗЕ)

ПРОГРАМУВАННЯ МІКРОКОНТРОЛЕРІВ

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	<i>Перший (бакалаврський)</i>
Галузь знань	12 «Інформаційні технології»
Спеціальність	121 «Інженерія програмного забезпечення»
Освітня програма	<i>Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці</i>
Статус дисципліни	Вибіркова
Форма навчання	Очна (денна)
Рік підготовки, семестр	III курс, осінній семестр
Обсяг дисципліни	120 годин / 4 кредити ЕКТС (лекції – 36 год., практичні заняття – 18 год., самостійна робота – 66 год.)
Семестровий контроль/ контрольні заходи	Залік / захист практичних занять, модульні тести, підсумковий тест або захист міні-проєкту
Розклад занять	http://rozklad.kpi.ua
Мова викладання	Українська
Інформація про керівника курсу / викладачів	Лектор:ст. викл. Сарибіга Ганна Володимирівна
Розміщення курсу	(LMS кафедри / репозиторій — заповнюється кафедрою)

Програма навчальної дисципліни

Дисципліна розглядає принципи та інженерні процеси створення програмного забезпечення для вбудованих систем — програм, що виконуються на мікроконтролерах або системах-на-кристалі (SoC), взаємодіють з апаратними інтерфейсами, датчиками та виконавчими пристроями і працюють в умовах жорстких обмежень щодо пам'яті, обчислювальних ресурсів, енергоспоживання та часу реакції. Курс орієнтований на інженерний підхід до розробки *embedded*-систем і охоплює як фундаментальні принципи архітектури мікроконтролерів, так і практичні аспекти створення *firmware* для реальних промислових пристроїв.

Навчальна дисципліна реалізується у форматі **дуальної освіти у співпраці з ІТ-компанією N-iX**, що дозволяє інтегрувати академічну підготовку з практичним досвідом індустріальної розробки *embedded*-систем. У межах курсу студенти знайомляться з сучасними підходами до розробки *firmware*, що використовуються в інженерних командах компаній-розробників програмного забезпечення для електроніки, IoT-пристроїв, енергетичних систем, автомобільної електроніки та промислової автоматизації.

Курс охоплює архітектуру сучасних мікроконтролерів (ARM Cortex-M, AVR, RISC-V), організацію пам'яті та регістрів, програмування портів введення-виведення, системи

переривань, таймерів, аналого-цифрових і цифро-аналогових перетворювачів, широтно-імпульсної модуляції, а також реалізацію обміну даними через комунікаційні інтерфейси UART, SPI та I²C. Особлива увага приділяється взаємодії програмного забезпечення з апаратними ресурсами системи, інтеграції сенсорів і виконавчих пристроїв, а також розробці програм, що працюють у режимі реального часу.

Важливою складовою дисципліни є знайомство студентів з інженерними практиками розробки *embedded*-програмного забезпечення, що застосовуються в індустрії. Зокрема розглядаються принципи модульної архітектури *firmware*, використання апаратно-абстрактних шарів (HAL), драйверів периферії та сервісних модулів, організація крос-компіляції та відтворюваної збірки програмного забезпечення, використання систем керування версіями та інструментів командної розробки. Окремий блок дисципліни присвячений налагодженню *embedded*-систем за допомогою апаратних відладчиків (JTAG/SWD), використанню логування, трасування виконання програм, а також методам тестування *firmware* на цільовій платформі (target).

З огляду на дуальний формат навчання значна увага приділяється інтеграції теоретичних знань з практичними задачами, що відображають реальні інженерні процеси в ІТ-компаніях. Практичні заняття та проєктні завдання базуються на індустріальних кейсах і передбачають розробку програмного забезпечення для мікроконтролерних платформ з використанням сучасних інструментів розробки, налагодження та тестування. Студенти набувають досвіду командної роботи, організації процесу розробки програмного забезпечення, використання систем керування версіями, систем відстеження задач і сучасних практик інженерії програмного забезпечення.

Метою навчальної дисципліни є формування та закріплення у студентів наступних компетентностей:

здатність до абстрактного мислення, аналізу та синтезу;
 здатність застосовувати знання у практичних ситуаціях;
 здатність вчитися і оволодівати сучасними знаннями;
 здатність до пошуку, оброблення та аналізу інформації з різних джерел;
 здатність працювати в команді;

здатність аналізувати архітектуру мікроконтролерних систем та їх апаратні можливості;

здатність розробляти програмне забезпечення для вбудованих систем з урахуванням апаратних обмежень;

здатність проектувати програмні модулі, драйвери периферійних пристроїв та системні компоненти *firmware*;

здатність забезпечувати якість програмного забезпечення відповідно до технічних вимог, стандартів та інженерних практик;

здатність використовувати сучасні інструментальні засоби розробки та налагодження *embedded*-програм;

здатність реалізовувати програмні компоненти для систем реального часу;

здатність інтегрувати програмне забезпечення з апаратними компонентами системи та забезпечувати їх узгоджену роботу;

здатність обґрунтовано обирати інструментарій для розробки, тестування та супроводу програмного забезпечення для *embedded*-пристроїв.

Зокрема, дисципліна спрямована на формування навичок організації повного циклу розробки програмного забезпечення для мікроконтролерних систем: від аналізу архітектури апаратної платформи та визначення функціональних і ресурсних вимог до створення програмної архітектури *firmware*, програмування периферійних модулів та інтеграції програмного забезпечення з апаратною частиною системи. Значна увага приділяється забезпеченню якості *embedded*-програмного забезпечення, використанню інструментів автоматизованого тестування, налагодженню програм на цільовому обладнанні та документуванню результатів розробки.

Предмет навчальної дисципліни — методи та засоби програмування мікроконтролерів і створення програмного забезпечення для вбудованих систем: архітектура мікроконтролерів; організація пам'яті та регістрів; програмування портів введення-виведення; система переривань; таймери та лічильники; робота з комунікаційними інтерфейсами (UART, SPI, I²C); використання аналого-цифрових та цифро-аналогових перетворювачів; реалізація широтно-імпульсної модуляції; інтеграція сенсорів та виконавчих механізмів; використання операційних систем реального часу; налагодження та тестування *embedded*-програм; оптимізація використання ресурсів мікроконтролера.

Програмні результати навчання, на формування та покращення яких спрямована дисципліна:

знати принципи функціонування мікроконтролерних систем та архітектуру *embedded*-пристроїв;

застосовувати сучасні інструменти та технології програмування *embedded*-систем; використовувати методи взаємодії програмного забезпечення з апаратними ресурсами системи;

застосовувати ефективні підходи до проектування програмного забезпечення для систем реального часу;

використовувати інструментальні засоби розробки, тестування та налагодження *embedded*-програм;

мати навички командної розробки та документування програмного забезпечення;

застосовувати методи тестування та перевірки програмного забезпечення;

забезпечувати якість програмного забезпечення для вбудованих систем;

використовувати інструменти керування програмними проектами;

документувати результати розробки та презентувати створені програмні рішення.

Після завершення дисципліни студент уміє: аналізувати архітектуру мікроконтролера та його периферійних модулів; програмувати апаратні ресурси мікроконтролера (GPIO, таймери, переривання); реалізовувати обмін даними через інтерфейси UART, SPI та I²C; працювати з аналого-цифровими та цифро-аналоговими перетворювачами; реалізовувати широтно-імпульсну модуляцію для керування виконавчими пристроями; інтегрувати сенсори та інші апаратні компоненти вбудованої системи; використовувати сучасні інструменти налагодження *embedded*-програм; оптимізувати програмний код з урахуванням обмежених ресурсів мікроконтролера; реалізовувати програмні рішення для систем реального часу та забезпечувати надійне функціонування *embedded*-пристроїв.

1. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

Пререквізити: основи програмування; компоненти програмної інженерії; основи комп'ютерних систем і мереж. Постреквізити: курсові/проектні роботи з мікроконтролерами, робототехнікою, IoT, промисловими та енергетичними системами; дисципліни з архітектури ПЗ, тестування, DevOps/CI/CD, надійності та безпеки; практика/стажування в embedded/firmware командах.

2. Зміст навчальної дисципліни

Розділ 1. Архітектура мікроконтролерів та базові механізми програмування

Тема 1.1. Контекст embedded-систем та роль мікроконтролерів у сучасних технічних системах

Тема 1.2. Архітектура сучасних мікроконтролерів (ARM Cortex-M, RISC-V, AVR)

Тема 1.3. Організація пам'яті та карта пам'яті мікроконтролера

Тема 1.4. Порти введення-виведення (GPIO) та взаємодія з апаратними компонентами

Тема 1.5. Система переривань у мікроконтролерних системах

Розділ 2. Периферійні модулі та інтерфейси мікроконтролерів

Тема 2.1. Таймери та лічильники

Тема 2.2. Широтно-імпульсна модуляція (PWM) та керування приводами

Тема 2.3. Комунікаційні інтерфейси UART, SPI та I²C

Тема 2.4. Промислові інтерфейси та протоколи (CAN, RS-485, Modbus)

Тема 2.5. Аналого-цифрові та цифро-аналогові перетворювачі

Розділ 3. Архітектура firmware та системи реального часу

Тема 3.1. Архітектура firmware та організація програмного коду

Тема 3.2. Керування пам'яттю та ресурсами мікроконтролера

Тема 3.3. Багатозадачність та операційні системи реального часу (RTOS)

Тема 3.4. Bootloader та оновлення firmware

Розділ 4. Інтеграція мікроконтролерних систем

Тема 4.1. Інтеграція сенсорів та виконавчих механізмів

Тема 4.2. Керування електроприводами та робототехнічними модулями

Тема 4.3. Передача телеметрії та взаємодія з зовнішніми системами

Тема 4.4. Оптимізація енергоспоживання embedded-систем

Розділ 5. Інженерні практики розробки embedded-систем

Тема 5.1. Toolchain та крос-компіляція програм для мікроконтролерів

Тема 5.2. Налаштування embedded-програм (JTAG, SWD, trace)

Тема 5.3. Тестування embedded-програм та забезпечення якості

Тема 5.4. Документування та супровід embedded-проектів

Розділ 6. Індустріальні кейси та командні проекти

Тема 6.1. Інтеграційні задачі програмування мікроконтролерів у кіберфізичних системах
 Тема 6.2. Розробка *firmware* для робототехнічних та IoT-пристроїв
 Тема 6.3. Командна розробка *embedded*-проектів у співпраці з компанією N-iX

3. Навчальні матеріали та ресурси

Основна література

1. Барретт С., Пак Д. **Microcontroller Programming and Interfacing with ARM Cortex-M** : навчальний посібник / Steven F. Barrett, Daniel J. Pack. – San Rafael : Morgan & Claypool Publishers, 2019. – 182 с.
2. Валвано Дж. **Embedded Systems: Real-Time Interfacing to ARM Cortex-M Microcontrollers** / Jonathan W. Valvano. – 2nd ed. – CreateSpace Independent Publishing Platform, 2017. – 522 с.
3. Яцюк В. М., Кравець І. І. **Мікроконтролери та їх застосування у вбудованих системах** : навчальний посібник / В. М. Яцюк, І. І. Кравець. – Львів : Видавництво Львівської політехніки, 2018. – 240 с.
4. Barr M., Massa A. **Programming Embedded Systems: With C and GNU Development Tools** / Michael Barr, Anthony Massa. – 2nd ed. – Sebastopol : O'Reilly Media, 2006. – 336 с.
5. Yiu J. **The Definitive Guide to ARM Cortex-M Microcontrollers** / Joseph Yiu. – 3rd ed. – Oxford : Newnes, 2014. – 784 с.
6. Wilmshurst T. **Designing Embedded Systems with ARM Microcontrollers** / Trevor Wilmshurst. – Oxford : Newnes, 2010. – 532 с.
7. STMicroelectronics. **STM32 Reference Manual та STM32 Programming Manual** : офіційна технічна документація. – Режим доступу: <https://www.st.com>
8. FreeRTOS Organization. **FreeRTOS Reference Manual та Developer Documentation** : офіційна документація операційної системи реального часу. – Режим доступу: <https://www.freertos.org>

Допоміжна література:

1. Simon Monk. **Programming Arduino: Getting Started with Sketches**. – McGraw-Hill Education, 2016.
2. Massimiliano Pagani. **Mastering STM32**. – Leanpub Publishing, 2017.
3. Axelson J. **Serial Port Complete: COM Ports, USB Virtual COM Ports, and Ports for Embedded Systems**. – Lakeview Research, 2007.
4. Microchip Technology Inc. **AVR Microcontroller Datasheet та Application Notes**. – Режим доступу: <https://www.microchip.com>
5. ARM Ltd. **ARM Cortex-M Architecture Reference Manual**. – Режим доступу: <https://developer.arm.com>
6. MISRA Consortium. **MISRA C: Guidelines for the Use of the C Language in Critical Systems**. – MISRA, 2012.

Інформаційні ресурси:

1. Електронний ресурс: **Software Engineering 2014 Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering**. A Volume of the Computing Curricula Series. IEEE Computer Society, 2015.
2. Електронний ресурс: **Guide to the Software Engineering Body of Knowledge (SWEBOK®), Version 3.0**. IEEE Computer Society.
3. Інтранет-сайт кафедри інженерії програмного забезпечення вбудованих систем [Електронний ресурс] – <http://ipze.kpi.ua/metodichki>
4. Модульне середовище університету **eCampus КПІ ім. Ігоря Сікорського** [Електронний ресурс] – <https://ecampus.kpi.ua/>
5. Портал технічної документації виробників мікроконтролерів (STMicroelectronics, Microchip, NXP)
[Електронний ресурс] – <https://www.st.com>
[Електронний ресурс] – <https://www.microchip.com>
[Електронний ресурс] – <https://www.nxp.com>
6. Офіційна документація **FreeRTOS**
[Електронний ресурс] – <https://www.freertos.org>
7. Онлайн-курс **Основи програмування мікроконтролерів**
[Електронний ресурс] – <http://www.intuit.ru/studies/courses/2188/174/info>
8. ARM Developer Documentation – технічна документація архітектури **ARM Cortex-M**
[Електронний ресурс] – <https://developer.arm.com>
9. Репозиторії прикладів embedded-проектів та бібліотек
[Електронний ресурс] – <https://github.com>

Навчальний контент

4. Методика опанування навчальної дисципліни(освітнього компонента)

Лекційні заняття

№ з/п	Назва теми лекції та перелік основних питань
1	Лекція 1. Тема 1.1. Контекст embedded-систем та роль мікроконтролерів. Основні питання: поняття вбудованих систем; області застосування (IoT, робототехніка, промислова автоматизація); особливості програмування мікроконтролерів; обмеження ресурсів системи (пам'ять, продуктивність, енергоспоживання); огляд сучасних мікроконтролерних платформ.
2	Лекція 2. Тема 1.2. Архітектура сучасних мікроконтролерів. Основні питання: архітектури AVR, ARM Cortex-M, RISC-V; структура ядра; регістри процесора; системні шини; периферійні модулі мікроконтролера.

3	Лекція 3. Тема 1.3. Організація пам'яті мікроконтролера. Основні питання: Flash-пам'ять, SRAM, EEPROM; карта пам'яті; стек і статична пам'ять; розміщення програм і даних; роль компілятора та linker script.
4	Лекція 4. Тема 1.4. Порти введення-виведення (GPIO). Основні питання: режими роботи GPIO; цифрові входи та виходи; підтягувальні резистори; керування світлодіодами та кнопками; принципи роботи цифрових сигналів.
5	Лекція 5. Тема 1.5. Система переривань. Основні питання: апаратні та програмні переривання; таблиця векторів; пріоритети переривань; обробники переривань; використання переривань у системах реального часу.
6	Лекція 6. Тема 2.1. Таймери та лічильники. Основні питання: режими роботи таймерів; генерація періодичних подій; вимірювання часових інтервалів; використання таймерів у керуванні пристроями.
7	Лекція 7. Тема 2.2. Широтно-імпульсна модуляція (PWM). Основні питання: принцип роботи PWM; формування сигналу; застосування PWM для керування двигунами та освітленням.
8	Лекція 8. Тема 2.3. Послідовні інтерфейси UART, SPI, I ² C. Основні питання: принципи обміну даними; синхронні та асинхронні інтерфейси; підключення периферійних пристроїв і сенсорів.
9	Лекція 9. Тема 2.4. Промислові інтерфейси. Основні питання: CAN, RS-485; протокол Modbus; використання промислових шин у системах автоматизації та робототехніці.
10	Лекція 10. Тема 2.5. Аналогові підсистеми мікроконтролерів. Основні питання: аналого-цифрові перетворювачі (ADC); цифро-аналогові перетворювачі (DAC); робота з сенсорами; оцифрування сигналів.
11	Лекція 11. Тема 3.1. Архітектура firmware. Основні питання: модульна структура embedded-програм; драйвери пристроїв; апаратно-абстрактний шар (HAL); організація програмного коду.
12	Лекція 12. Тема 3.2. Керування ресурсами мікроконтролера. Основні питання: використання пам'яті; оптимізація програм; ефективне використання периферійних модулів.
13	Лекція 13. Тема 3.3. Операційні системи реального часу. Основні питання: принципи RTOS; задачі; планування виконання; синхронізація задач; огляд FreeRTOS.

14	<i>Лекція 14. Тема 3.4. Bootloader та оновлення firmware. Основні питання: призначення bootloader; процес оновлення програмного забезпечення; безпечне оновлення firmware.</i>
15	<i>Лекція 15. Тема 4.1. Інтеграція сенсорів. Основні питання: підключення датчиків; обробка сигналів сенсорів; приклади використання в IoT-системах.</i>
16	<i>Лекція 16. Тема 4.2. Керування приводами та робототехнічними системами. Основні питання: керування двигунами; драйвери моторів; енкодери; використання мікроконтролерів у робототехніці.</i>
17	<i>Лекція 17. Тема 5.1. Інструменти розробки embedded-програм. Основні питання: компілятори та toolchain; процес крос-компіляції; використання апаратних відладчиків; пошук помилок у програмі.</i>
18	<i>Лекція 18. Тема 6.1. Індустріальні embedded-проєкти. Основні питання: життєвий цикл embedded-проєкту; командна розробка; приклади індустріальних систем; підсумки курсу.</i>

№ з/п	Назва теми заняття та перелік основних питань
1	<i>Практичне заняття №1. Налаштування середовища розробки та створення першого embedded-проєкту. (пов'язано з лекціями 1–2, теми 1.1–1.2). Основні питання: встановлення середовища розробки та компілятора; створення першого проєкту для мікроконтролера; структура embedded-проєкту; компіляція та завантаження прошивки на мікроконтролер.</i>
2	<i>Практичне заняття №2. Робота з портами введення-виведення (GPIO). (пов'язано з лекцією 4, тема 1.4). Основні питання: конфігурація портів введення-виведення; керування світлодіодами; зчитування стану кнопок; реалізація простих алгоритмів керування цифровими сигналами.</i>
3	<i>Практичне заняття №3. Використання таймерів мікроконтролера. (пов'язано з лекціями 5–6, теми 1.5 та 2.1). Основні питання: налаштування таймерів; генерація періодичних сигналів; використання таймерів для керування часовими подіями.</i>
4	<i>Практичне заняття №4. Генерація PWM-сигналів. (пов'язано з лекцією 7, тема 2.2). Основні питання: налаштування PWM; керування яскравістю світлодіодів; керування сервоприводом; використання PWM для керування двигунами.</i>
5	<i>Практичне заняття №5. Послідовний обмін даними через UART. (пов'язано з лекцією 8, тема 2.3). Основні питання: налаштування UART-інтерфейсу; передача та прийом даних; робота з термінальними програмами; обробка послідовних повідомлень.</i>
6	<i>Практичне заняття №6. Робота з інтерфейсами SPI та I²C. (пов'язано з лекцією 8, тема 2.3). Основні питання: підключення периферійних пристроїв; зчитування даних із сенсорів; передача даних між пристроями.</i>
7	<i>Практичне заняття №7. Робота з аналоговими сигналами (ADC). (пов'язано з лекцією 10, тема 2.5). Основні питання: налаштування ADC; зчитування аналогових сигналів; обробка сигналів сенсорів.</i>
8	<i>Практичне заняття №8. Налагодження embedded-програм. (пов'язано з лекцією 17, тема 5.1). Основні питання: використання апаратного відладчика; запуск програми у режимі налагодження; покрокове виконання програм; перегляд значень змінних; пошук та виправлення помилок у програмі.</i>

9	Практичне заняття №9. Командний міні-проєкт з програмування мікроконтролерної системи. (пов'язано з лекцією 18, тема 6.1).інтеграція програмного забезпечення з апаратною платформою; тестування роботи сенсорів, приводів та комунікаційних інтерфейсів; демонстрація роботи пристрою; презентація архітектури <i>firmware</i> ; аналіз прийнятих інженерних рішень; захист командного проєкту, виконаного у співпраці з компанією N-iX ..
---	--

Самостійна робота студента

<i>№ з/п</i>	<i>Вид самостійної роботи</i>	<i>Кількість годин СРС</i>
1	<i>Опрацювання технічної документації мікроконтролерів (datasheet, reference manual), SDK та середовищ розробки</i>	8
2	<i>Вивчення архітектури мікроконтролерів та периферійних модулів (GPIO, таймери, ADC, комунікаційні інтерфейси)</i>	8
3	<i>Самостійне опрацювання матеріалів щодо програмування периферійних модулів мікроконтролера</i>	10
4	<i>Підготовка до практичних занять (аналіз завдань, написання та тестування програмного коду)</i>	12
5	<i>Розробка програмних модулів для командного embedded-проєкту</i>	18
6	<i>Тестування, налагодження та підготовка демонстрації роботи мікроконтролерної системи</i>	6
7	<i>Підготовка до модульного контролю та підсумкового контролю (залік / екзамен)</i>	4

5. Контрольні заходи

Поточний контроль: захист практичних занять; перевірка репозиторію та *project board*; модульні тести за темами.

Семестровий контроль: залік у формі підсумкового тесту та/або захисту міні-проєкту (визначається викладачем).

Політика та контроль

5. Політика навчальної дисципліни (освітнього компонента)

Система вимог, які викладач ставить перед студентом:

- *правила відвідування занять: відповідно до Наказу 1-273 від 14.09.2020 р. заборонено оцінювати присутність або відсутність здобувача на аудиторному занятті, в тому числі нараховувати заохочувальні або штрафні бали. Відповідно до РСО даної дисципліни бали нараховують за відповідні види навчальної активності на лекційних та практичних заняттях.*
- *правила поведінки на заняттях: студент має можливість отримувати бали за відповідні види навчальної активності на лекційних та практичних заняттях, передбачені РСО дисципліни. Використання засобів зв'язку для пошуку інформації на гугл-диску викладача, в інтернеті, в дистанційному курсі на платформі Сікорський здійснюється за умови вказівки викладача;*
- *політика дедлайнів та перескладань: якщо студент не проходив або не з'явиться на модульний тест (без поважної причини), його результат оцінюється у 0 балів. Перескладання результатів модульних тестів не передбачено.*
- *політика щодо академічної доброчесності: дозволено використання відкритих бібліотек/прикладів лише з коректним посиланням і поясненням власного внеску. Кодекс честі Національного технічного університету України «Київський політехнічний інститут» <https://kpi.ua/files/honorcode.pdf>.*
- *при використанні цифрових засобів зв'язку з викладачем (мобільний зв'язок, електронна пошта, переписка на форумах та у соцмережах тощо) необхідно дотримуватись загальноприйнятих етичних норм, зокрема бути ввічливим та обмежувати спілкування робочим часом викладача.*

6. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Поточний контроль: захист практичних занять; перевірка репозиторію та project board (GitHub Projects / Jira); модульні тести.

Календарний контроль: провадиться двічі на семестр як моніторинг поточного стану виконання вимог силабусу. Умовою позитивної атестації є значення поточного рейтингу студента не менше 50 % від максимально можливого на час атестації.

Семестровий контроль: залік.

Умови допуску до семестрового контролю: семестровий рейтинг більше 30 балів.

Таблиця відповідності рейтингових балів оцінкам за університетською шкалою:

Кількість балів	Оцінка
95-100	Відмінно
85-94	Дуже добре
75-84	Добре
65-74	Задовільно
60-64	Достатньо
Менше 60	Незадовільно
Менше 30	Не допущено

Загальна рейтингова оцінка студента після завершення семестру складається з балів, отриманих за:

- виконання та захист практичних занять;
- виконання модульних тестів (2 модульні тести);
- підсумковий контроль

<i>Практичні заняття (9×4)</i>	<i>Модульні тести (2×11)</i>	<i>Підсумковий контроль (залік)</i>
36	22	42

Практичні заняття

Ваговий бал 4. Максимальна кількість балів за практичні заняття – 4 бали × 9 занять = 36 балів.

Практичні заняття здаються з демонстрацією відтворюваності: репозиторій + білд + запуск тестів + project board (GitHub Projects або Jira); для on-target частини – лог/демонстрація на стенді (за наявності). Артефакти (вимоги, моделі, дизайн-ноти, тести, реліз-ноти) мають бути версіоновані у репозиторії.

Критерії оцінювання

- завдання виконано повністю, вчасно, з відтворюваним результатом – 4 бали;
- виконано з незначними зауваженнями – 3 бали;
- виконано з суттєвими зауваженнями або із затримкою – 2 бали;
- виконано частково – 1 бал;
- не виконано – 0 балів.

Модульні тести

Ваговий бал 11. Максимальна кількість балів – 11 балів × 2 = 22 бали. Модульний тест 1 проводиться після Розділу 1 (теми 1.1–1.5: процеси, вимоги, моделювання, project management). Модульний тест 2 проводиться після Розділу 2 (теми 2.1–2.9: архітектура, тестування, CI, реліз).

Критерії оцінювання

- повна правильна відповідь – 11 балів;
- відповідь з незначними помилками – 8–10 балів;
- відповідь з суттєвими помилками – 4–7 балів;
- відповідь переважно неправильна – 1–3 бали;
- відповідь відсутня або повністю неправильна – 0 балів.

Підсумковий контроль

Форма семестрового контролю – залік. Максимальний бал – 42.

Підсумковий контроль проводиться у формі залікового тесту або захисту міні-проєкту (визначається викладачем на початку семестру).

Максимальна сума балів складає 100. Для отримання заліку «автоматом» потрібно мати рейтинг не менше 60 балів та виконані умови допуску. Студенти з рейтингом менше 60 балів або ті, хто хоче підвищити оцінку, виконують залікову контрольну роботу (при цьому набрані бали анулюються, оцінка за контрольну є остаточною).

Робочу програму навчальної дисципліни (силабус):

Складено Сарибогою Нанною Володимирівною, ст. викл. кафедри ІПЗЕ

Ухвалено кафедрою: _____ (протокол № ____ від __.__.20__)

Погоджено методичною комісією: _____ (протокол № ____ від __.__.20__)

Тематика командних проєктів (у співпраці з N-iX)

Студенти виконують командний міні-проєкт, який передбачає розробку програмного забезпечення для мікроконтролерної системи.

Можливі теми проєктів:

Система збору даних з сенсорів
(температура, освітленість, рух) з передачею даних через UART або CAN.

Контролер робототехнічної платформи
керування двигунами, енкодерами та сенсорами.

IoT-пристрій моніторингу середовища
збір і передача телеметрії.

Система керування сервоприводами
для роботизованого маніпулятора.

Мікроконтролерний контролер двигуна
з використанням PWM та сенсорів.

Пристрій керування освітленням
з використанням датчиків та PWM.

Мікроконтролерна система збору телеметрії для промислового обладнання
із використанням RS-485 або CAN.

Контролер мобільного робота
керування рухом, датчиками відстані та моторними драйверами.