МІНІСТЕРСТВО ВНУТРІШНІХ СПРАВ УКРАЇНИ
ЛЬВІВСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ ВНУТРІШНІХ СПРАВ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ УПРАВЛІННЯ, ПСИХОЛОГІЇ ТА БЕЗПЕКИ

Кафедра інформаційних технологій

РОЗРОБЛЕННЯ СИСТЕМИ СТАБІЛІЗАЦІЇ РУХОМИХ ПЛАТФОРМ З ДАВАЧАМИ ТА МАНІПУЛЯТОРАМИ КОЛІСНОГО ТА ГУСЕНИЧНОГО РОЗМІНОВУЮЧОГО ТРАНСПОРТУ

Кваліфікаційна робота
здобувача вищої освіти
4 курсу денної форми навчання
Христина МАРТИНІВ

Науковий керівник:
доцент, кандидат технічних наук
Ігор ФАРМАГА

Рецензент:
доцент, кандидат технічних наук
_____

Кваліфікаційна робота допущена до захисту
«___» _____ 2026 р., протокол № ____

Завідувач кафедри інформаційних технологій
_____Олег ЗАЧЕК
(підпис)

Львів
2026

СПИСОК УМОВНИХ СКОРОЧЕНЬ

ПК – індивідуальний (персональний) комп’ютер

МК – мікроконтролер; спеціалізована мікросхема для керування електронними пристроями

АЗ/ПЗ – апаратне (hardware) та програмне (software) забезпечення системи

ВС – вбудована (ембеддед) система; обчислювальний вузол, інтегрований безпосередньо в об’єкт керування

ЦП – цифровий пристрій

МЕМС (MEMS) – мікроелектромеханічні системи; технології, що поєднують мікроелектронні та мікромеханічні компоненти

RAM (ОЗП) – пам’ять із довільним доступом; використовується для тимчасового зберігання даних під час роботи

ROM (ПЗП) – пам’ять лише для зчитування; енергонезалежне сховище для незмінної інформації

EEPROM – тип енергонезалежної пам’яті, що допускає багаторазове електричне стирання та запис

FLASH – різновид напівпровідникової технології пам’яті (тип EEPROM), що характеризується високою швидкістю перепрограмування

AVR – лінійка популярних 8-бітних мікроконтролерів, розроблена Atmel (нині Microchip)

ADC (АЦП) – модуль для перетворення аналогового сигналу у цифровий код.

LCD (РКД) – дисплей на основі рідких кристалів

SPI – синхронний протокол для швидкого обміну даними між МК та периферією у повнодуплексному режимі

I2C – послідовна шина обміну даними, що використовує лише два сигнали: лінію даних (SDA) та лінію синхронізації (SCL)

Bare-metal програмування – пряме написання коду для МК без операційної системи, що забезпечує максимальну продуктивність і контроль над ресурсами МК.

АНОТАЦІЯ

Мартинів Х., Фармага І. (керівник). Розроблення системи стабілізації рухомих платформ з давачами та маніпуляторами колісного та гусеничного розміновуючого транспорту. Бакалаврська кваліфікаційна робота. – Львівсьий державний університет внутрішніх справ, Львів, 2026.

Кваліфікаційна робота присвячена створенню програмно-апаратного комплексу для стабілізації мобільних платформ. Система базується на 8-бітному мікроконтролері сімейства AVR та використовує тривісний MEMS-акселерометр для корекції положення маніпуляторів і датчиків, встановлених на колісних чи гусеничних роботах-саперах.

Функціонал розробленої системи полягає у безперервному моніторингу просторової орієнтації об’єкта. Отримані дані про кути відхилення візуалізуються на РК-панелі, водночас система формує відповідні керуючі імпульси для приводів, що забезпечують стабілізацію платформи.

Апаратна конфігурація системи стабілізації базується на обчислювальних потужностях мікроконтролера родини AVR. Інформаційну базу системи складає цифровий триосьовий MEMS-акселерометр, а для візуалізації поточних параметрів у схему інтегровано рідкокристалічний індикатор.

У межах проєкту було спроєктовано принципову електричну схему, а також сформовано віртуальну модель системи стабілізації мобільних об’єктів у середовищі Proteus VSM. Програмний комплекс системи реалізовано мовою програмування C, з використанням спеціалізованого середовища CodeVisionAVR. Верифікацію працездатності розробленого алгоритму та всього пристрою виконано шляхом імітаційного моделювання в емуляторі Proteus ISIS.

Ключові слова: мікроконтролерний модуль, динамічна стабілізація, засоби розмінування, тривісний акселерометр, MEMS-технології, LCD-індикація, архітектура AVR, середовище Proteus VSM, мова C, вбудовані системи (Embedded Systems), компілятор CodeVisionAVR.

ABSTRACT

Martyniv Kh., Farmaha I. (supervisor). Development of a stabilization system for mobile platforms with sensors and manipulators for wheeled and tracked mine-clearing vehicles. Bachelor's thesis. – Lviv State University of Internal Affairs, Lviv, 2026.

The qualification work is dedicated to the creation of a hardware-software complex for stabilizing mobile platforms. The system is based on an 8-bit microcontroller of the AVR family and uses a three-axis MEMS accelerometer to correct the position of manipulators and sensors installed on wheeled or tracked mining robots.

The functionality of the developed system consists in continuous monitoring of the spatial orientation of the object. The obtained data on the angles of deviation are visualized on the LCD panel, while the system generates the corresponding control pulses for the drives, which ensure the stabilization of the platform.

The hardware configuration of the stabilization system is based on the computing power of the AVR microcontroller. The information base of the system is a digital three-axis MEMS accelerometer, and an LCD indicator is integrated into the circuit to visualize the current parameters.

Within the framework of the project, a schematic electrical circuit was designed, as well as a virtual model of the mobile object stabilization system was formed in the Proteus VSM environment. The software complex of the system was implemented in the C programming language, using the specialized CodeVisionAVR environment. Verification of the operability of the developed algorithm and the entire device was performed by simulation modeling in the Proteus ISIS emulator.

Keywords: microcontroller module, dynamic stabilization, demining tools, three-axis accelerometer, MEMS technologies, LCD display, AVR architecture, Proteus VSM environment, C language, embedded systems, CodeVisionAVR compiler.

ЗМІСТ

ВСТУП

Розвиток мікроконтролерної техніки та впровадження MEMS-акселерометрів відкрили шлях до розробки бюджетних інтелектуальних комплексів для вимірювання та контролю. Одним із перспективних напрямків застосування цих технологій є створення систем стабілізації для мобільних об’єктів, зокрема колісних та гусеничних роботів-саперів, що оснащені сенсорними вузлами та маніпуляторами.

Сучасна мікропроцесорна техніка базується на використанні мікроконтролерів як центральних вузлів керування. Висока інтеграція периферійних модулів — таких як порти введення-виведення, АЛП та енергонезалежна пам’ять – безпосередньо в структурі кристала дозволяє мінімізувати кількість зовнішніх компонентів. Це не лише спрощує програмну реалізацію складних алгоритмів, а й значно оптимізує габарити пристрою та його енергоефективність.

Система на базі мікроконтролера та триосьового MEMS-акселерометра має широкий спектр застосування: від забезпечення стійкості мобільних об’єктів до використання у стаціонарних спорудах як прецизійного інклінометра для високоточного визначення кутів нахилу.

Для побудови апаратної частини системи стабілізації рухомих об’єктів на основі тривісного MEMS-акселерометра обрано мікроконтролер архітектури AVR виробництва компанії Atmel. Дане рішення зумовлене оптимальним балансом продуктивності та надійності при обробці даних з інерціальних датчиків.

Мета роботи полягає у проєктуванні апаратної бази та розробці вбудованого ПЗ для системи стабілізації мобільних об’єктів. Функціонування пристрою ґрунтується на використанні триосьового MEMS-акселерометра, що забезпечує моніторинг прискорень за трьома координатами. На основі аналізу отриманих даних мікроконтролер реалізує алгоритм прийняття рішень щодо активації виконавчих механізмів, водночас здійснюючи вивід службової інформації на символьний рідкокристалічний індикатор.

Для реалізації поставленої мети визначено такі **завдання**:

- виконати розробку архітектури системи стабілізації в середовищі Proteus VSM, забезпечивши коректне підключення периферійних модулів до контролера;
- сформувати логіку функціонування вбудованого ПЗ, визначивши послідовність обробки сигналів від сенсорів та механізмів корекції положення платформи;
- створити програмну бібліотеку для ініціалізації та зчитування даних з датчика ADXL345 за допомогою цифрових протоколів передачі;
- написати код для взаємодії з LCD-модулем 16x2, забезпечивши вивід поточних параметрів стабілізації в режимі реального часу;
- реалізувати комплексне програмне забезпечення для ATmega128, яке об’єднує математичні розрахунки положення та керування приводами відповідно до закладеного алгоритму;
- здійснити імітаційне моделювання роботи системи в Proteus ISIS. Оцінити ефективність стабілізації платформи при зміні кутів нахилу та провести аналіз результатів тестування.

Об’єкт дослідження – процеси функціонування та стабілізації положення рухомих платформ колісного та гусеничного розміновувального транспорту, що оснащений сенсорними системами (давачами) та робототехнічними маніпуляторами, в умовах складного рельєфу та динамічних навантажень.

Предмет дослідження – методи, алгоритми, апаратні засоби та програмне забезпечення мікроконтролерної системи, що забезпечують компенсацію коливань, підтримку горизонту та точність позиціонування маніпулятора/датчиків на рухомій базі під час виконання спеціальних завдань.

Методи дослідження. Під час виконання роботи застосовано методи теорії автоматичного керування, математичне моделювання, методи цифрової обробки сигналів, методи програмування вбудованих систем (bare-metal, interrupt-driven, модульне та структурне програмування, оптимізація пам’яті та

енергоспоживання) для розробки програмного забезпечення мікроконтролера, схемотехнічне проектування для розробки апаратної частини (вибір давачів, інтерфейсів зв’язку та систем живлення).

Структура роботи. Кваліфікаційна робота складається із вступу, чотирьох розділів, висновків, списку використаних джерел, додатків. Обсяг основного тексту роботи складає 77 сторінок, 36 рисунків, 13 таблиць, 1 додаток і 22 бібліографічних джерела. Загальний обсяг роботи – 88 сторінок.

РОЗДІЛ 1
СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1. Системи стабілізації рухомих платформ

Традиційні методи стабілізації рухомих об’єктів базуються на застосуванні механічних гіростабілізаторів. Попри високу точність, їх виробництво є технологічно складним та витратним процесом. Такі пристрої виконують подвійну функцію: безпосереднє утримання заданої орієнтації приладів та вимірювання їхніх кутових координат. Відповідно до механізму функціонування, прийнято виділяти три основні класи гіроскопічних систем: силові, індикаторні та пристрої безпосередньої дії [1].

Гіростабілізатори безпосередньої дії функціонують завдяки використанню інерційних властивостей триступеневого гіроскопа. Такі пристрої знаходять застосування для мінімізації бортової хитавиці морських суден, стабілізації вагонів монорейкових залізниць тощо. Слід зазначити, що подібні конструкції вирізняються значною масою та габаритами. Крім того, їх застосовують для підтримання стійкості прецизійних елементів у системах автоматичного керування (приклад наведено на Рис. 1.1).

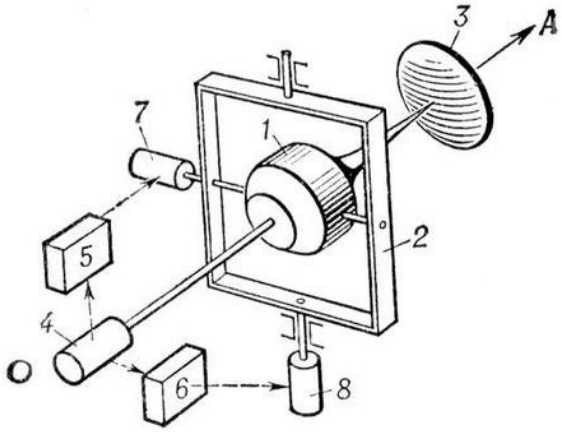


Рис. 1.1. Структурно-принципова схема гіроскопічного стежного вузла: 1 — гіроскопічна камера з інтегрованим ротором; 2 — зовнішня рама (карданне підвісся); 3 — антена пристрою; 4 — блок координатора; 5, 6 — блоки підсилення та перетворення сигналів; 7, 8 — датчики (давачі) моментів.

Конструкція гіростабілізатора базується на гірокамері з ротором (1), що інтегрована у зовнішню карданну раму (2) для забезпечення безпосередньої стабілізації антени (3) та координатора (4). Останній генерує керуючі сигнали, величина яких залежить від кута відхилення антени від заданої осі ОА. Ці дані через підсилювально-перетворювальні блоки (5, 6) передаються на датчики моментів (7, 8) контуру корекції. Така система забезпечує автоматичне супроводження антеною встановленого напрямку, що класифікує її як гіроскопічну стежну систему.

Силові гіростабілізатори (гірорами) належать до класу електромеханічних вузлів, які, окрім гіроскопічних елементів, оснащені спеціалізованими двигунами для нейтралізації зовнішніх дестабілізуючих моментів. Сфера їхнього застосування охоплює морські судна, авіаційну техніку та інші об’єкти, де необхідна стабілізація вимірювальних пристроїв. На принципах силової стабілізації базується робота гіровертикалей, гіроскопів напрямку, а також комплексних агрегатів — гіроазимутгоризонтів. За конструктивним виконанням такі системи поділяють на одно- чи двогіроскопні, а за кількістю ступенів вільності – на одно-, дво- та тривісні.

Комбінація двох одноосьових модулів дозволяє створити двовісний гіростабілізатор, призначений для утримання платформи в горизонтальній площині; така конфігурація здатна функціонувати як силова гіровертикаль. У разі поєднання трьох одноосьових пристроїв формується тривісна силова система (гіроазимутгоризонт). Цей комплекс об’єднує властивості гіровертикалі та гіроскопа напрямку (гіроазимута), що дає змогу вимірювати три просторові кути орієнтації об’єкта. Подібні рішення є критично важливими для навігації морських суден та авіації, зокрема в інерційних системах для забезпечення просторової стабілізації платформ.

Індикаторні гіростабілізатори функціонують як системи автоматичного регулювання, де гіроскопічні вузли відіграють роль чутливих або задавальних елементів. Встановлені на рухомій платформі, вони визначають її просторову орієнтацію, тоді як безпосереднє вирівнювання об’єкта реалізується за

допомогою стежних систем. Роль сенсорів, що реагують на зміну кутової швидкості або кути нахилу, зазвичай виконують двоступеневі (зокрема, поплавкові інтегрувальні) гіроскопи, гіротахометри або триступеневі астатичні пристрої. Такі стабілізатори є невід'ємною частиною інерційних навігаційних комплексів на морських суднах та в авіації.

Сфера впровадження гіроскопічних стабілізаторів охоплює широкий спектр галузей: від видобувної промисловості (стабілізація бурових платформ) до побутової електроніки та персонального транспорту. Зокрема, ці системи є критично важливими для стабілізації оптичних осей фото- та відеокамер, систем наведення рухомих артилерійських установок, а також у робототехніці. Останнім часом технологія набула масового поширення в індивідуальних засобах пересування (гіроборди, сигвеї), дорожньо-будівельній техніці (асфальтоукладальники) та високоточних вимірювальних приладах.

1.2. Огляд підходів до проєктування систем стабілізації мобільних платформ з використанням сенсорів руху

Результатом ґрунтовного аналізу спеціалізованої літератури та сучасних інтернет-ресурсів став збір вичерпного обсягу даних стосовно існуючих розробок у сфері систем стабілізації мобільних об'єктів. Опрацьована інформація дозволила систематизувати ключові технічні рішення, що використовуються для підтримки просторової орієнтації рухомих платформ.

Сучасний стан розвитку мікропроцесорної техніки характеризується широким спектром підходів до проектування цифрових модулів стабілізації мобільних об'єктів. Основним вектором розробки є інтеграція високопродуктивних мікроконтролерів із прецизійними MEMS-сенсорами, що дозволяє створювати компактні та енергоефективні системи керування.

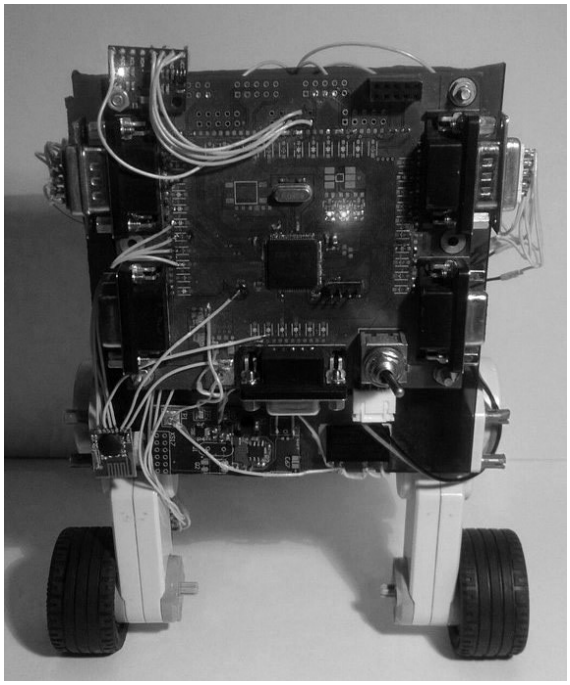
Автоматизована система стабілізації мобільних платформ функціонує як автономний комплекс, що забезпечує детекцію відхилень від заданих координат та їх подальшу компенсацію без участі оператора. Отримані параметри просторової орієнтації можуть фіксуватися у внутрішній

енергонезалежній пам’яті (наприклад, EEPROM) або транслюватися на віддалений термінал через дротові чи бездротові канали зв’язку для подальшого моніторингу.

Традиційні механічні комплекси стабілізації базуються на використанні класичних гіроскопів. На противагу їм, на Рис. 1.2–1.4 представлено сучасні варіації цифрових систем керування просторовим положенням мобільних платформ, розроблені на основі актуальних мікропроцесорних технологій [2,3].

Рис. 1.2. Макетний зразок системи стабілізації для проведення експериментальних досліджень

Рис. 1.3. Спеціалізовані стабілізатори (стедіками) для фіксації положення фото- та відеокамер



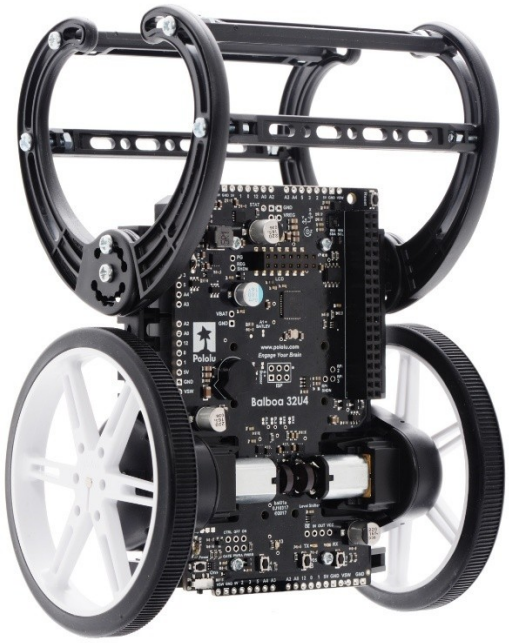


Рис. 1.4. Двоколісний робот-балансир з системою активної стабілізації вертикального положення

У кваліфікаційній роботі представлена система стабілізації для колісних та гусеничних платформ спеціалізованого розмінувального транспорту, реалізована на базі тривісного MEMS-акселерометра. Запропоноване рішення функціонує за принципом індикаторного гіростабілізатора, проте вирізняється розширеними аналітичними можливостями щодо обробки та візуалізації даних. Завдяки відсутності масивних рухомих компонентів, розробка є компактною, надійною в умовах вібрацій та економічно ефективнішою за традиційні механічні аналоги того ж призначення

РОЗДІЛ 2
ЗАСОБИ РОЗРОБКИ АПАРАТНОГО І ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ СТАБІЛІЗАЦІЇ РУХОМИХ ПЛАТФОРМ

2.1. Середовище автоматизованого проєктування та моделювання мікроконтролерних систем Proteus VSM

Proteus VSM представляє собою комплексне середовище автоматизованого проектування (САПР), призначене для розробки та імітаційного моделювання електронних схем. Ключовою особливістю пакета є можливість наскрізного проектування пристроїв на базі мікроконтролерів різноманітних архітектур, а також популярних платформ прототипування, зокрема Arduino [4].

Середовище розробки Proteus VSM від компанії Labcenter Electronics забезпечує потужний інструментарій для симуляції електронних схем. Програма базується на використанні точних математичних моделей компонентів та підтримує спільне моделювання апаратної частини з вбудованим програмним забезпеченням мікроконтролерів, DSP-процесорів та модулів Arduino. Це робить її оптимальним вибором для проектування складних цифрових систем стабілізації.

Середовище Proteus надає інструментарій для схемотехнічного проектування, комплексного моделювання роботи електронних вузлів та верифікації прошивок мікроконтролерів. Завдяки наявності віртуальної вимірювальної апаратури, розробник може провести детальне тестування системи стабілізації ще до створення фізичного прототипу.

Можливості Proteus охоплюють не лише схемотехнічне моделювання, а й проектування друкованих плат із застосуванням тривимірного відображення. Пакет підтримує симуляцію значної кількості мікроконтролерів (AVR, PIC, ARM7, Motorola тощо) завдяки розгалуженій бібліотеці моделей. Наявність вбудованих засобів підтримки для архітектур AVR, PIC та ARM7

робить цей програмний продукт універсальним рішенням для відладки мікропроцесорних систем стабілізації.

Велика номенклатура вбудованих моделей (понад 6000 одиниць) та підтримка поширених компліяторів та асемблерів роблять Proteus універсальним засобом розробки. Програма дозволяє здійснювати верифікацію складних апаратних комплексів, що містять декілька мікропроцесорних модулів. Висока достовірність симуляції динамічних процесів у таких системах забезпечує ефективне виявлення помилок у програмному коді та схемотехніці.

Варто усвідомлювати, що жодне комп'ютерне моделювання не здатне на 100% передати поведінку фізичних електронних компонентів. Проте для тестування та коригування алгоритмів управління мікроконтролером можливостей симуляції цілком достатньо. Середовище Proteus вирізняється розгалуженою базою електронних компонентів з можливістю власноручного створення відсутніх моделей. Завдяки підтримці стандарту SPICE, користувачі можуть інтегрувати специфікації безпосередньо від розробників заліза. Якщо потрібний елемент не має вбудованої логіки моделювання, його актуальну SPICE-модель зазвичай можна знайти та завантажити на офіційному ресурсі виробника.

Система Proteus інтегрує в собі два головні компоненти: графічний редактор схем ISIS, що слугує початковою точкою проектування (Рис.2.1), та модуль ARES, який використовується для розведення друкованих плат. Останній підтримує автоматизацію рутинних процесів (трасування та розміщення) завдяки вбудованому двигуну ELECTRA.

Серед ключових особливостей Proteus варто виділити підтримку зовнішніх комунікацій:

- COMPIM – синхронізація моделі з реальним RS-232 (COM-портом).
- USBCONN – забезпечення доступу віртуальної схеми до фізичного USB-порта.

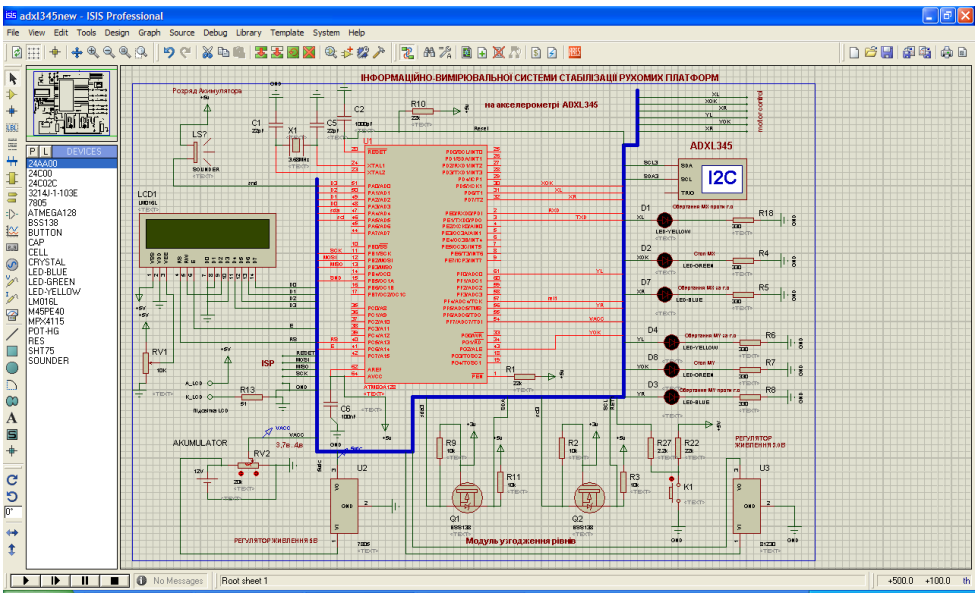


Рис. 2.1. Графічний інтерфейс модуля Proteus ISIS

Використання САПР Proteus VSM дозволяє здійснювати розробку, дебаггінг та верифікацію мікропроцесорного програмного забезпечення на етапі проектування, випереджаючи створення реального апаратного прототипу.

Взаємодія Proteus із зовнішніми компіляторами та середовищами програмування дозволяє симулювати код, написаний у:

1. Популярних екосистемах: Наприклад, Arduino (за умови додавання необхідних бібліотечних компонентів).
2. Інструментах для AVR: Зокрема, у середовищах WinAVR та CodeVisionAVR.
3. Спеціалізованому ПЗ для PIC та 8051: Такому як компілятори HiTECH.
4. Мультиплатформених рішеннях: Наприклад, ICC, що підтримує ARM7, msp430 та AVR.

2.2. Інструментарій розробки ПЗ в середовищі CodeVisionAVR

CodeVisionAVR — це функціональне інтегроване середовище розробки (IDE), спеціально оптимізоване для створення програмного забезпечення для мікроконтролерів сімейства AVR від компанії Atmel [5]. Програма поєднує в собі компілятор мови C та зручні засоби автоматизації програмування.

Архітектура CodeVisionAVR базується на власному високоефективному С-компіляторі, оптимізованому під архітектуру AVR. Набір вбудованих інструментів забезпечує повний цикл проектування — від написання початкового коду до прошивки мікроконтролера, що значно спрощує розробку програмного забезпечення для кристалів Atmel [6, 7]. На Рис.2.2 представлено інтерфейс IDE, що включає редактор вихідного коду та панелі керування проектом для мікроконтролерів AVR

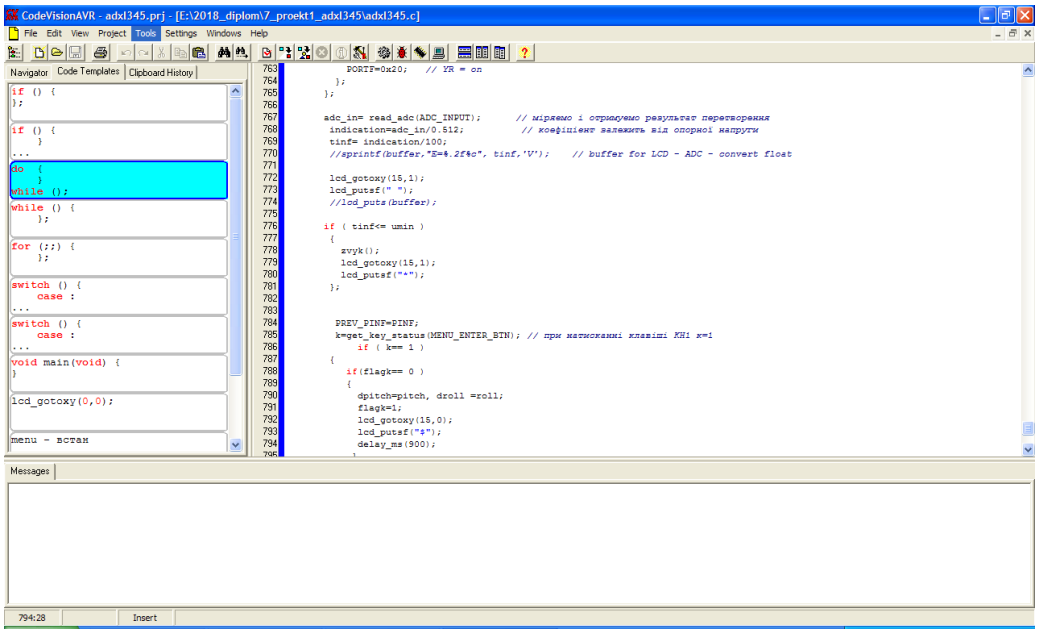


Рис. 2.2. Робоче вікно інтегрованого середовища CodeVisionAVR

Середовище CodeVisionAVR відрізняється інтуїтивно зрозумілим інтерфейсом та наявністю детальної документації. Компілятор, інтегрований у програму, генерує ефективний та компактний машинний код, а синтаксис мови максимально наближений до класичного стандарту ANSI C.

Завдяки інтеграції CodeVisionAVR із середовищем Atmel Studio, розробник отримує комплексний інструментарій для проектування та відлагодження ПЗ під усю лінійку мікроконтролерів Atmel. Технічна документація та демонстраційні проекти доступні в директорії встановлення програми (`\cvavr\bin`). Слід зауважити, що робочий інтерфейс представлений виключно англійською мовою, а офіційні засоби локалізації не передбачені.

Програма забезпечує стабільну роботу на базі операційних систем Windows. Її функціонал реалізовано через набір взаємопов'язаних утиліт, де за компіляцію відповідає GNU GCC, за роботу з функціями – бібліотека avr-libc, а за редагування тексту – Programmers Notepad. Важливою перевагою є підтримка "класичного" С та Асемблера, а також можливість інтеграції в середовище AVR Studio для зручного відлагодження.

Таблиця 2.1. Типи даних мови С для AVR

Тип	Розмір (біт)	Діапазон значень
bit	1	0, 1
char	8	-128…127
unsigned char	8	0…255
signed char	8	-128…127
int	16	-32768…32767
short int	16	-32768…32767
unsigned int	16	0…65535
signed int	16	-32768…32767
long int	32	-2147483648…2147483647
unsigned long int	32	0…4294967295
signed long int	32	-2147483648…2147483647
float	32	±1.175e-38…±3.402e38
double	32	±1.175e-38…±3.402e38

Таблиця 2.2. Зарезервовані слова

break	else	int	struct
bit	enum	interrupt	switch
case	extern	long	typedef
char	flash	register	union
const	float	return	unsigned
continue	for	short	void
default	funcused	signed	volatile
do	goto	sizeof	while
double	if	sfrw	
eeprom	inline	static	

РОЗДІЛ 3
АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ СТАБІЛІЗАЦІЇ РУХОМИХ ПЛАТФОРМ

3.1. Визначення комплектуючих для проєктування системи стабілізації рухомих платформ

Для реалізації системи стабілізації, що інтегрується в колісні та гусеничні платформи для розмінування, оснащені маніпуляторами й сенсорними модулями, було визначено наступний склад апаратних засобів:

- обчислювальний центр: 8-бітний мікроконтролер архітектури AVR – Atmel ATmega128, що забезпечує необхідну швидкість обробки даних;
- модуль візуалізації: рідкокристалічний символьний індикатор WH1602B-YGK-CTK (формат 16x2) із жовто-зеленою колірною схемою для виводу службової інформації;
- сенсорна підсистема: прецизійний триосьовий МЕМС-акселерометр ADXL345, призначений для відстеження просторової орієнтації платформи;

Опис мікроконтролера AVR ATmega128. На сучасному ринку мікропроцесорної техніки провідні позиції займають 8-бітні архітектури PIC від Microchip та AVR від Atmel, а також високопродуктивні рішення на базі архітектури ARM. Популярність лінійки AVR серед розробників зумовлена її гармонійною архітектурою, високим показником швидкодії та розширеною системою команд у поєднанні з ефективними режимами енергозбереження та доступною вартістю. Враховуючи баланс між обчислювальною потужністю, енергоефективністю та зручністю впровадження, для реалізації даного проекту було обрано мікроконтролер ATmega128 [8,9].

Мікроконтролери AVR (Atmel) – це потужні 8-бітні RISC-пристрої, що широко використовуються в сучасній електроніці. Вони вирізняються високою продуктивністю та простотою розробки. Зокрема, підтримка

багаторазового перепрограмування (до 10 000 циклів) у складі готового модуля робить їх ідеальними для налагодження складних алгоритмів.

Сімейство мікроконтролерів AVR традиційно поділяється на три ключові категорії, кожна з яких адаптована під конкретні технічні завдання:

- Tiny AVR: Найбільш компактні та бюджетні рішення. Зазвичай постачаються у 8-контактних корпусах і призначені для найпростіших пристроїв, де критичними є розмір та вартість.
- Classic AVR: Базова лінійка, що поєднує оптимальну продуктивність (до 16 MIPS) із помірними ресурсами. Вони оснащені Flash-пам'яттю об'ємом 2–8 Кб, енергонезалежною пам’яттю EEPROM (64–512 байт) та оперативною пам’яттю SRAM (128–512 байт).
- Mega AVR: Найпотужніша серія для реалізації складних проектів. Ці МК мають розширений обсяг Flash-пам’яті (до 128 Кб) та SRAM (до 4 Кб). Окрім високої обчислювальної потужності (4–16 MIPS), вони містять розвинену периферію, зокрема інтегрований 10-бітовий 8-канальний АЦП.

Характерною рисою цієї лінійки мікроконтролерів є повна спадкоємність архітектури. Це дозволяє безперешкодно переносити програмний код з базових моделей на продуктивніші рішення в межах одного сімейства без необхідності переробляти систему команд.

Мікроконтролер моделі ATmega128 належить до 8-бітних пристроїв архітектури AVR. Його ключовою характеристикою є наявність вбудованої FLASH-пам’яті об’ємом 128 КБ, яка підтримує можливість внутрішньосистемного програмування.

Архітектура та продуктивність мікроконтролера ATmega128. Серце пристрою базується на високоефективній RISC-архітектурі, яка включає 133 інструкції. Переважна більшість команд виконується за один такт генератора, що забезпечує швидкість обробки даних до 16 MIPS при частоті 16 МГц. Регістрова модель: 32 загальних 8-бітних робочих регістри. Режим роботи: повністю статична архітектура. Поділ пам’яті: незалежний доступ до пам’яті

програм та даних (Гарвардська структура). Пам’ять та інтерфейси програмування. Мікроконтролер оснащений трьома типами пам’яті з високим ресурсом перезапису: FLASH (128 Кбайт): внутрішньосистемне програмування, до 10 000 циклів; EEPROM (4 Кбайт): енергонезалежне зберігання даних, до 100 000 циклів; SRAM (4 Кбайт): внутрішня оперативна пам’ять. Інтерфейс JTAG (IEEE1149.1): використовується для відладки на апаратному рівні, програмування пам’яті та налаштування бітів захисту.

Багатий набір вбудованих функцій дозволяє реалізувати складні системи керування: таймерна підсистема: два 8-бітних та два 16-бітних таймера/лічильника з підтримкою режимів порівняння, захоплення та програмованих дільників; сигнали та вимірювання: 8-канальний 10-бітний АЦП, аналоговий компаратор та 6 каналів ШІМ; комунікації: два програмованих порти USART та послідовний інтерфейс SPI (режими Master/Slave); контроль стабільності: таймер реального часу (RTC), сторожовий таймер (Watchdog) та вбудований калібрований RC-генератор; електротехнічні параметри та енергоспоживання: живлення: модель ATmega128L працює в діапазоні 2,7 – 5,5 В (до 8 МГц), стандартна версія ATmega128 – 4,5 – 5,5 В (до 16 МГц); енергозбереження: підтримка 6 режимів сну (Idle, Power-down, Standby та інші) для мінімізації споживання струму; введення/виведення: 53 програмовані лінії I/O; корпуси: випускається у 64-вивідних корпусах типу TQFP або MLF.

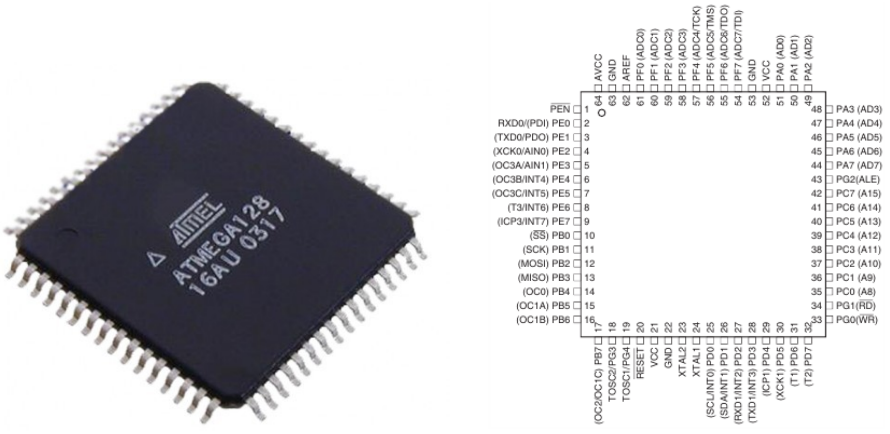


Рис. 3.1. Функціональне призначення пінів мікроконтролера AVR ATmega128 для типів корпусів TQFP та MLF

В основі архітектури AVR лежить поєднання розгалуженої системи команд із 32-бітним блоком регістрів загального призначення (РЗП). Пряме підключення всіх 32 регістрів до арифметико-логічного пристрою (АЛП) дозволяє зчитувати два операнди та виконувати операцію всього за один системний такт. Такий підхід забезпечує значну перевагу в обчислювальній потужності, що в середньому вдесятеро перевищує показники аналогічних CISC-мікроконтролерів.

Мікроконтролер ATmega128/L оснащений 128 Кбайт вбудованої Flash-пам’яті з підтримкою технології Self-Programming (читання під час запису). Об’єм незалежної пам’яті EEPROM та оперативної пам’яті SRAM становить по 4 Кбайт. Архітектура пристрою включає 32 робочі регістри, інтерфейс JTAG для налагодження та чотири таймери-лічильники з функцією порівняння. Периферія представлена модулями USART, I2C, SPI та 8-канальним АЦП з розрядністю 10 біт. Для підвищення надійності передбачено Watchdog-таймер, а для оптимізації споживання – шість енергоощадних режимів, зокрема Idle, у якому робота центрального процесора призупиняється.

Для мінімізації цифрових шумів під час перетворення в ATmega128 передбачено особливий режим, де працюють виключно АЦП та асинхронний таймер. Інші режими енергозбереження включають: Power-down: зупиняє роботу пристрою із збереженням даних у регістрах до зовнішнього виклику або Reset. Standby: підтримує роботу лише тактового генератора, вимикаючи решту периферії.

Висока енергоефективність мікроконтролера ATmega128 забезпечується його здатністю до миттєвого виходу з енергозберігаючих станів у робочий режим, зокрема через зовнішні переривання. У конфігурації Extended Standby активними залишаються як основний, так і асинхронний тактові генератори. Гнучкість розробки підтримується завдяки інтегрованій ISP FLASH пам’яті: оновлення коду можливе безпосередньо в системі через інтерфейс SPI або за допомогою завантажувача (Boot Loader), що функціонує з

виділеної області пам’яті ядра AVR. Структурну схему мікроконтролера AVR ATmega128 представлено на Рис. 3.2.

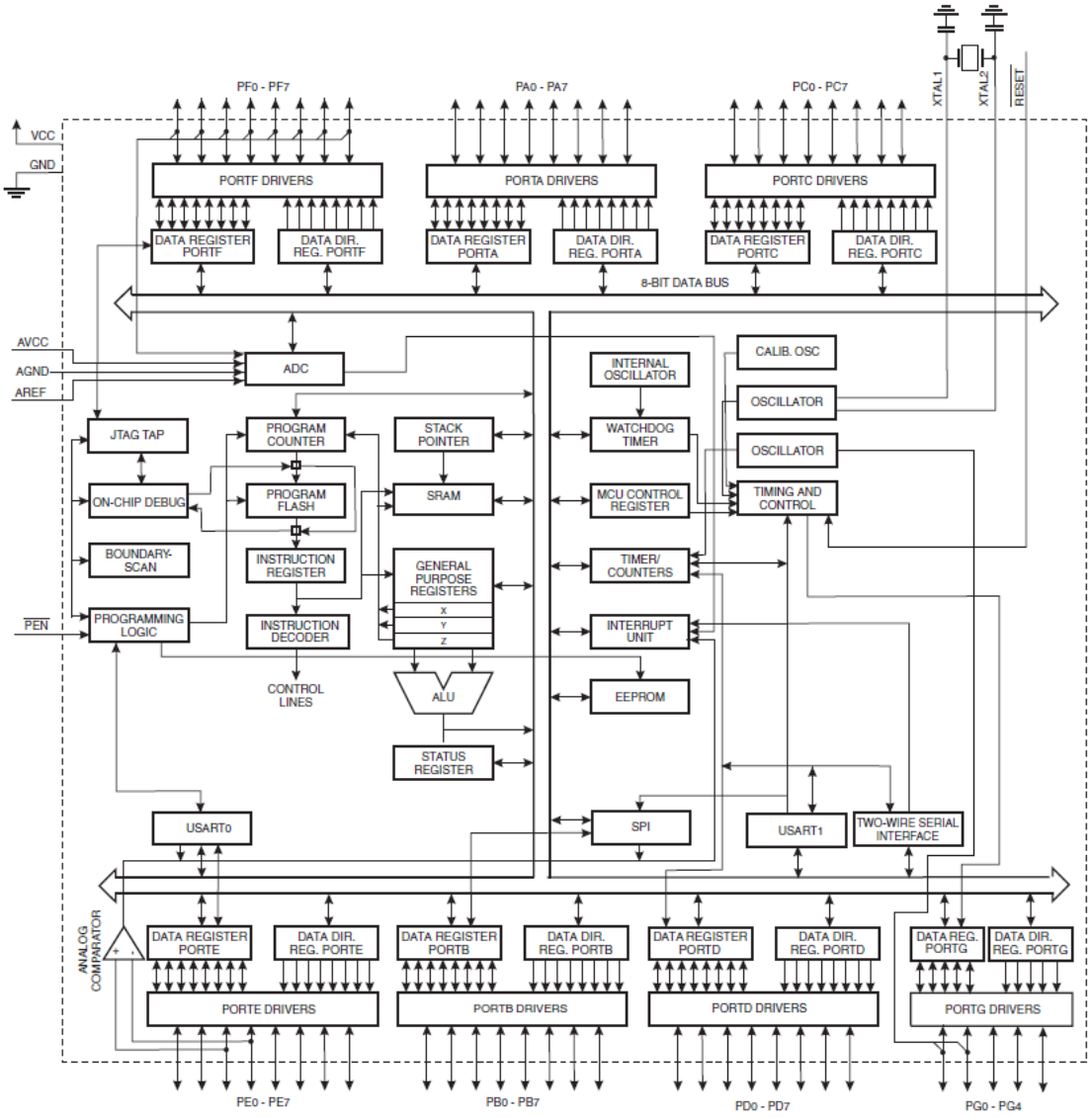


Рис. 3.2. Структурна схема архітектури мікроконтролера ATmega128

Завдяки поєднанню прогресивної 8-бітної RISC-архітектури центрального процесора та інтегрованої FLASH-пам’яті, мікроконтролер ATmega128 стає універсальним і рентабельним рішенням для розробки сучасних вбудованих систем керування.

Для взаємодії із зовнішніми пристроями мікроконтролер використовує порти введення-виведення, оперуючи цифровими рівнями: логічним “нулем” та логічною “одиницею”. Зокрема, у моделі ATmega128 при живленні 5 В порогове значення для логічного “0” становить 0...1,5В, тоді як логічна “1” охоплює діапазон від 1,9 до 5В. Оскільки в реальних задачах часто виникає

необхідність обробки аналогових сигналів із довільним значенням напруги в межах джерела живлення, архітектура AVR передбачає наявність інтегрованого аналого-цифрового перетворювача (АЦП).

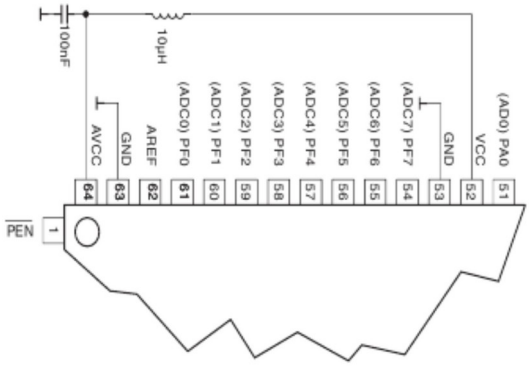


Рис. 3.3. Конфігурація вхідних каналів АЦП мікроконтролера ATmega128

Для взаємодії з периферійним модулем АЦП у ATmega128, входи якого суміщені з портом F, виділено наступні керуючі та інформаційні ресурси:

1. ADCSRA (Control and Status Register A): Керує ініціалізацією, частотою дискретизації та прапорцями переривань.
2. ADMUX (Multiplexer Selection Register): Забезпечує комутацію вхідних ліній порту до вимірювального вузла.
3. SFIOR (Special Function IO Register): Розширює можливості керування тригерами запуску.
4. ADCH & ADCL: Зберігають 10-бітний результат перетворення у двох 8-бітних комірках.

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Таблиця 3.1. Режими вибору еталонної напруги (V_{REF}) для модуля АЦП

REFS1	REFS0	Джерело опорної напруги ADC
0	0	Використання зовнішнього опорного сигналу через вивід AREF при деактивованому внутрішньому джерелі VREF.
0	1	Використання напруги живлення аналогового каскаду (AVCC) як опорної, за умови встановлення фільтруючої ємності на вивід AREF.
1	0	Зарезервовано
1	1	Використання інтегрованого прецизійного джерела на 2,56 В як опорного рівня; при цьому вивід A_{REF} задіюється для підключення зовнішньої фільтруючої ємності.

Параметри діапазону перетворення АЦП безпосередньо залежать від обраної опорної напруги (AREF). У випадку, коли рівень вхідного однополярного сигналу перевищує встановлений поріг AREF, вихідне значення фіксується на максимальній позначці – 0x3FF. Для мікроконтролера доступні три конфігурації еталона: використання напруги аналогового живлення (AVCC), внутрішнього стабілізатора на 2,56 В або подача напруги від зовнішнього пристрою через відповідний вивід.

Комутація напруги AVCC до модуля АЦП здійснюється за допомогою пасивного ключа. Формування внутрішнього еталона на рівні 2,56 В забезпечується джерелом опорної напруги (U_{ET}), сигнал якого проходить через буферний підсилювач. Оскільки в будь-якій конфігурації вивід AREF має пряме з'єднання з перетворювачем, для придушення шумів та стабілізації опорного сигналу доцільно використовувати фільтруючий конденсатор, підключений між магістраллю AREF та загальною шиною (“землею”).

Рівень напруги на виводі AREF можна проконтролювати за допомогою вольтметра з високим вхідним опором. Оскільки внутрішнє джерело AREF має значний вихідний опір, воно не розраховане на роботу з активним навантаженням; допускається лише підключення зовнішньої ємності для фільтрації.

ADLAR = 0:

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	–	–	ADC9	ADC8	ADCH
	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ADLAR = 1:

Bit	15	14	13	12	11	10	9	8	
	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
	ADC1	ADC0	–	–	–	–	–	–	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

Рис. 3.4. Організація даних у регістрах ADCH та ADCL

Результатом функціонування АЦП є 10-розрядне цифрове значення, що зберігається в регістрах даних ADCH та ADCL (Рис.3.4). Стандартна конфігурація передбачає правостороннє вирівнювання, при якому дані займають молодші 10 біт 16-бітної структури. Проте, за допомогою біта ADLAR у регістрі ADMUX, розробник може активувати лівостороннє вирівнювання, зміщуючи результат у старші розряди регістрової пари.

Конфігурація аналогового входу та диференціального каскаду підсилення здійснюється за допомогою встановлення відповідних значень бітів MUX у керуючому регістрі ADMUX. Система дозволяє використовувати як однополярні входи (канали ADC0–ADC7), так і додаткові джерела: потенціал “землі” (GND) або вбудоване джерело опорної напруги з номіналом 1,22 В (Табл.3.2).

Таблиця 3.2. Визначення параметрів каналу та встановлення значень коефіцієнта підсилення

MUX4..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
00000	ADC0	N/A		
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			
01000(1)		ADC0	ADC0	10x
01001		ADC1	ADC0	10x
01010(1)	N/A	ADC0	ADC0	200x
01011		ADC1	ADC0	200x
01100		ADC2	ADC2	10x
01101		ADC3	ADC2	10x
01110		ADC2	ADC2	200x
01111		ADC3	ADC2	200x
10000		ADC0	ADC1	1x
10001		ADC1	ADC1	1x
10010		ADC2	ADC1	1x
10011		ADC3	ADC1	1x
10100		ADC4	ADC1	1x
10101		ADC5	ADC1	1x
10110		ADC6	ADC1	1x
10111		ADC7	ADC1	1x
11000		ADC0	ADC2	1x
11001		ADC1	ADC2	1x
11010		ADC2	ADC2	1x
11011		ADC3	ADC2	1x
11100		ADC4	ADC2	1x
11101		ADC5	ADC2	1x
11110	1.23V (V_{BG})	N/A		
11111	0V (GND)			

Диференціальний режим роботи передбачає гнучке налаштування інвертуючих та неінвертуючих ліній, що підключаються до відповідного підсилювача. У цьому разі пристрій обробляє різницю потенціалів між парою обраних входів, множачи її на встановлений коефіцієнт. Отриманий посилений сигнал спрямовується безпосередньо на вхід АЦП. Варто зазначити, що при переході до однополярної схеми вимірювання блок підсилення виключається з ланцюга обробки. Функціональне призначення керуючих бітів регістра ADCSRA:

Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Таблиця 3.3. Конфігурація коефіцієнта ділення тактової частоти ADC

ADPS2	ADPS1	ADPS0	Коефіцієнт ділення
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Блок АЦП оснащений вбудованим дільником, що синхронізується з тактовим сигналом мікроконтролера. Налаштування коефіцієнта ділення (у діапазоні від 2 до 128 з кроком 2^n) здійснюється через конфігурацію бітів ADPS, розташованих у керуючому регістрі ADCSRA.

Процес ділення частоти активується одночасно з увімкненням модуля АЦП через встановлення прапорця ADEN у регістрі ADCSRA. Дільник функціонує лише за умови, що біт ADEN перебуває в одиничному стані; його скидання до нуля зупиняє роботу вузла. Для забезпечення повної 10-бітної точності перетворення необхідно підтримувати частоту тактування схеми послідовного наближення в межах 50 – 200 кГц.

Тривалість обробки сигналу в АЦП зазвичай становить 13 тактів. Однак під час першого циклу перетворення, що слідує за встановленням біта ADEN, цей час збільшується до 25 тактів, оскільки системі потрібен додатковий

ресурс для ініціалізації аналогового тракту. Підсумкове значення автоматично переноситься до вихідних регістрів даних, одночасно з чим активується індикатор завершення операції – біт ADIF.

Режим одиничного перетворення передбачає скидання біта ADSC після завершення роботи. Повторне встановлення цього біта ініціює новий цикл з першим тактом АЦП. У режимі автоматичного перезапуску перетворення відбуваються циклічно без паузи, тому біт ADSC не скидається і залишається активним протягом усього часу роботи.

Для активації поодинокого циклу АЦП необхідно встановити біт ADSC у стан логічної “1”. Цей прапор активності залишається піднятим протягом усього періоду оцифрування і автоматично скидається апаратно, як тільки результат стає готовим. Якщо в момент виконання операції змінити налаштування вхідного каналу, контролер не перерве роботу миттєво, а спочатку коректно завершить поточне вимірювання.

Функціонування АЦП у режимі автозапуску передбачає постійне перетворення аналогового сигналу з регулярним оновленням результатів у відповідному регістрі даних. Для активації цієї опції необхідно встановити прапорець ADFR у регістрі керування ADCSRA (записати логічну одиницю).

Активація першого циклу оцифрування відбувається за умови встановлення біта ADSC у регістрі ADCSRA (запис логічної “1”). У такому стані аналого-цифровий перетворювач працює безперервно, здійснюючи ітерації одну за одною. При цьому стан прапорця переривання ADIF не впливає на стабільність процесу перетворення.

Активація режиму автоматичного запуску АЦП здійснюється шляхом встановлення біта ADATE у стан логічної одиниці. Після цього перетворювач ініціює кожен новий цикл оцифрування за позитивним фронтом сигналу від обраного тригера. Конкретне джерело, що викликає старт перетворення, визначається комбінацією бітів ADTS у регістрі SFIOR.

Сигналом про закінчення циклу оцифрування та оновлення значень у регістрах результату (ADCH:ADCL) є встановлення прапорця переривання

ADIF. У випадку, якщо в системі дозволені переривання (активований біт ADIE та глобальний прапорець I у регістрі SREG), контролер автоматично переходить до виконання відповідної підпрограми обробки переривання після завершення конвертації.

Очищення прапорця ADIF відбувається автоматично, як тільки процесор переходить до виконання відповідного вектора переривання. Крім того, розробник може скинути цей біт програмно, записавши в нього логічну одиницю. Проте слід бути обережним під час маніпуляцій з регістром ADCSRA: використання операцій типу «читання-модифікація-запис», а також команд SBI чи CBI, може призвести до небажаного скасування запланованого переривання.

Локальний дозвіл переривань від АЦП активується шляхом запису логічної “1” у біт ADIE. Однак для того, щоб контролер почав обробляти ці запити після завершення кожного циклу перетворення, необхідно також забезпечити глобальний дозвіл переривань через прапорець I у регістрі SREG.

Таблиця 3.4. Конфігурація регістра SFIOR: біти вибору джерела автозапуску АЦП

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	–	ACME	PUD	PSR2	PSR10	SFIOR
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free Running mode
0	0	1	Analog Comparator
0	1	0	External Interrupt Request 0
0	1	1	Timer/Counter0 Compare Match
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter1 Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

Функціонування бітів ADTS безпосередньо залежить від стану прапорця ADATE у регістрі ADCSRA. Якщо цей прапорець встановлено у значення логічної “1”, активується режим автоматичного запуску, де вибір конкретного

джерела тригера визначається конфігурацією бітів ADTS. У випадку, коли ADATE дорівнює “0”, механізм автозапуску вимкнено, а поточний стан бітів ADTS ігнорується системою.

Старт перетворення синхронізовано з наростаючим фронтом обраного джерела синхронізації. У випадку використання конфігурації ADTS[2:0]=0 (Free Running mode), перехід у цей режим сам по собі не створює пускового імпульсу. Стан прапорця переривання АЦП при цьому не впливає на генерацію сигналу запуску.

Для сполучення акселерометра з мікроконтролером доступні інтерфейси SPI або I2C. Зокрема, архітектура ATmega128 підтримує апаратний модуль TWI (Two-Wire Interface), який є повним аналогом двопровідної шини. Цей інтерфейс використовує дві лінії для двонаправленого обміну даними: послідовну лінію тактування (SCL) та лінію передачі інформації (SDA).

Шина дозволяє паралельне підключення до 128 периферійних пристроїв. Стабільність передачі сигналів SDA та SCL забезпечується шляхом їхнього з’єднання з лінією живлення через підтягувальні резистори, номінал яких зазвичай становить від 4 до 10 кОм. Ідентифікація кожного учасника обміну на шині здійснюється за допомогою унікальної апаратної адреси.

При використанні тактового генератора мікроконтролера з частотою 16 МГц, пропускна здатність шини може досягати 400 кГц. З огляду на особливості фізичного рівня інтерфейсу I2C, сумарна довжина сполучних ліній обмежена і не має перевищувати 6 метрів для забезпечення цілісності сигналів.

Опис трьохосьового MEMS-акселерометра ADXL345. Пристрій, що фіксує інтенсивність зміни швидкості руху тіла, називається акселерометром. Він вимірює прискорення, яке математично виражається у m/s2 або через кратність до вільного падіння – у G-силах (g).

Для земних умов показник прискорення вільного падіння становить 9.8 м/с2, проте на інших небесних тілах це значення варіюється. Використання

акселерометрів дозволяє ефективно фіксувати рівень вібрації в механізмах, а також визначати просторове положення об’єктів [10].

MEMS-акселерометр представляє собою мікроелектромеханічну систему, призначену для реєстрації прискорень обох типів: статичних та динамічних. До першої категорії належить земне тяжіння (гравітація), тоді як до другої — будь-які механічні коливання, вібрації або зміна положення об’єкта в просторі.

На сьогодні ринок електронних компонентів представлений великою кількістю MEMS-акселерометрів із різними технічними характеристиками. Ключовими критеріями вибору таких пристроїв є їхня чутливість, робоча напруга живлення та розміри корпусу. За конструктивним виконанням вони поділяються на одно-, дво- та трьохосьові. Останні вважаються найбільш універсальними, оскільки дозволяють реєструвати вектори прискорення одночасно за трьома декартовими осями (X, Y, Z).

У Табл.3.6 наведено технічні специфікації MEMS-акселерометрів виробництва компанії Analog Devices. Аналіз представлених даних свідчить, що при виборі датчика ключовими критеріями є діапазон вимірюваного прискорення, електричні характеристики (напруга живлення та споживання струму), а також чутливість пристрою. Крім того, важливе значення мають кількість осей вимірювання, підтримувані протоколи передачі даних для зв’язку з мікроконтролерами (МК) та економічна доцільність (вартість).

Згідно з даними таблиці, представлені моделі можна розподілити за кількістю осей вимірювання. До одноосьових пристроїв належать ADXL1004, ADXL1005 та ADXL151. Категорію двохосьових датчиків складають ADXL295, ADXL251 та ADXL288. Найбільш функціональними є трьохосьові акселерометри, зокрема моделі ADXL357, ADXL355 та популярний ADXL345.

Таблиця 3.5. Порівняльна характеристика експлуатаційних параметрів та вартості акселерометрів

Part#	Accelerometer Range	Is (typ) (A)	Vs+ (min) (V)	Temp Range	BW (typ) (Hz)	Output Type	Sensing Axis	# of Axes	Price (1000+) ($ US)
ADXL1005	100 g	1m	3	-40 to 125°C	23k	Analog	X	1	$39.41 (ADXL1005BCPZ)
ADXL1004	500 g	1m	3.3	-40 to 125°C	24k	Analog	X	1	$35.85 (ADXL1004BCPZ)
ADXL372	200 g	22µ	1.6	-40 to 105°C	3.2k	I²C, SPI	X, Y, Z	3	$6.33 (ADXL372BCCZ-RL7)
ADXL357	10 g, 20 g, 40 g	200µ	2.25	-40 to 125°C	1k	I²C, SPI	X, Y, Z	3	$31.87 (ADXL357BEZ)
ADXL356	10 g, 20 g, 40 g	150µ	2.25	-55 to 125°C	1.5k	Analog	X, Y, Z	3	$28.68 (ADXL356BEZ)
ADXL251	120 g, 240 g, 480 g, 60 g	6m	3.135	-	400	PSI5, SPI	X, Y	2	-
ADXL1002	50 g	1m	3.3	-40 to 125°C	11k	Analog	X	1	$29.66 (ADXL1002BCPZ)
ADXL1001	100 g	1m	3.3	-40 to 125°C	11k	Analog	X	1	$29.66 (ADXL1001BCPZ)
ADXL355	2 g, 4 g, 8 g	200µ	2.25	-40 to 125°C	1k	I²C, SPI	X, Y, Z	3	$28.33 (ADXL355BEZ)
ADXL354	2 g, 4 g, 8 g	150µ	2.25	-40 to 125°C	1.5k	Analog	X, Y, Z	3	$25.50 (ADXL354BEZ)
ADXL151	120 g, 240 g, 480 g	5m	4.5	-	400	PSI5	X	1	-
ADXL316	16 g	350µ	1.8	-40 to 105°C	1.6k	Analog	X, Y, Z	3	$3.76 (ADXL316WBCSZ)
ADXL700	14.2	5m	3.135	-	-	SPI	X, Y, Z	3	-
ADXL363	2 g, 4 g, 8 g	1.8µ	1.6	-40 to 85°C	200	SPI	X, Y, Z	3	$4.57 (ADXL363BCCZ-RL7)

Серед трьохосьових моделей найбільш економічно вигідним є датчик ADXL345 (Рис.3.5). Його технічні характеристики цілком відповідають вимогам даного проєкту, що робить його оптимальним вибором для практичної реалізації [10].

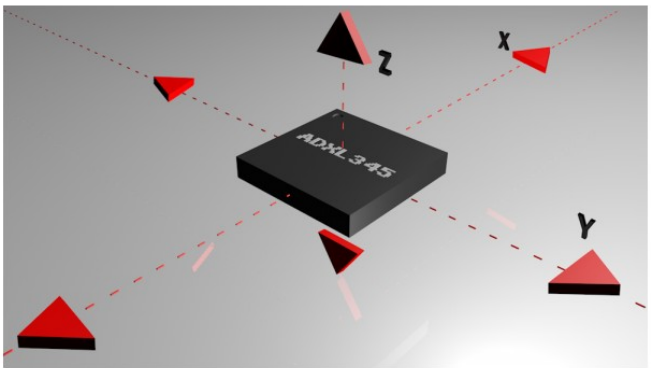


Рис. 3.5. Орієнтація вимірювальних осей у тривимірній системі координат акселерометра ADXL345

Сучасні акселерометри класифікуються за кількістю осей вимірювання (від однієї до трьох). Завдяки тенденції до зниження вартості виробництва, трьохосьові моделі стають дедалі популярнішими, оскільки забезпечують повний аналіз руху в тривимірному просторі за доступною ціною.

Функціонування мікроелектромеханічних акселерометрів ґрунтується на ємнісному методі детектування. Пристрій містить нерухомі статори та підвішені на мікропружинах рухомі структури. Коли на сенсор діє прискорення, відстань між пластинами змінюється, що безпосередньо впливає

на ємність системи. Отримана різниця потенціалів перетворюється електронною схемою у цифрове або аналогове значення прискорення.

Окрему категорію становлять п’єзоелектричні акселерометри, принцип дії яких базується на властивості певних матеріалів генерувати електричний заряд під впливом механічних напружень. У таких сенсорах прикладене прискорення створює силу, що деформує п’єзокристал, викликаючи появу різниці потенціалів на його гранях.

Для більшості сучасних акселерометрів характерна можливість налаштування динамічного діапазону вимірювань. Залежно від моделі, цей показник варіюється від +/-1g до +/-250g. Варто зауважити, що між шкалою вимірювання та точністю пристрою існує зворотна залежність: використання вужчого діапазону дозволяє досягти вищої чутливості сенсора.

Для реєстрації незначних вібрацій поверхні робочого столу доцільно застосовувати датчики з низьким порогом вимірювання. Водночас широкі діапазони, що сягають +/-250g, використовуються у специфічних галузях, наприклад, для аналізу критичних прискорень під час запуску ракет.

Будову типового MEMS-акселерометра зручно вивчити на прикладі триосьової моделі ADXL-345. Внутрішня архітектура та логіка роботи цього датчика відображені на функціональній схемі (Рис. 3.6).

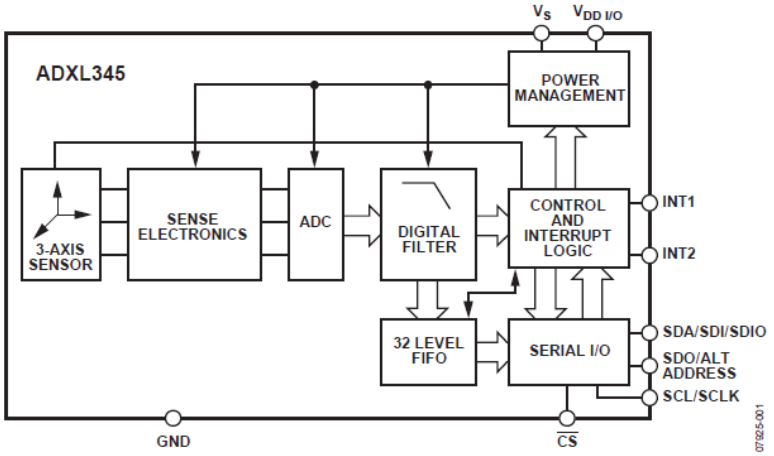


Рис. 3.6. Структурно-функціональна організація компонентів акселерометра ADXL345

Основним чинником статичних похибок вимірювання виступає триосьовий чутливий елемент, виконаний у вигляді мікромеханічної структури.

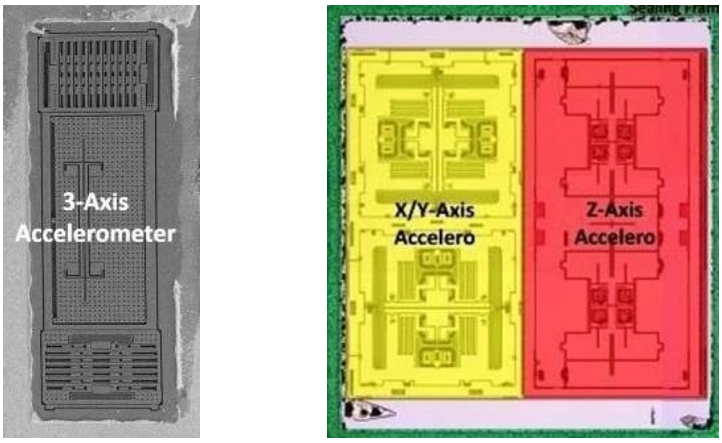


Рис. 3.7. Тривимірні чутливі елементи акселерометрів, виготовлені за технологією MEMS

Через можливу невідповідність вихідних сигналів по різних осях при однаковому прискоренні виникає потреба в індивідуальному налаштуванні. Для усунення таких похибок у кожному каналі передбачені спеціальні регістри зміщення (OFSX, OFSY та OFSZ), які заповнюються за результатами процедури калібрування пристрою.

Модель ADXL345 представляє собою компактний трьохосьовий акселерометр, що характеризується низьким енергоспоживанням та високою точністю (роздільна здатність становить 13 біт). Датчик підтримує гнучке налаштування діапазонів вимірювання прискорення: від +/-2g до +/-16g із кроком у 2 рази. Передача виміряних значень у 16-бітному форматі здійснюється через цифрові послідовні інтерфейси I2C або SPI.

Акселерометр ADXL345 є представником класу ємнісних мікроелектромеханічних систем (MEMS). Завдяки вузькій смузі пропускання (від 0,05 до 1600 Гц) цей пристрій демонструє високу ефективність при аналізі низькочастотних вібраційних процесів, реєстрації статичного прискорення, а також при визначенні просторової орієнтації та кутів нахилу з точністю понад 1,0°.

Смуга пропускання визначає спроможність датчика реєструвати високочастотні коливання прискорення. Зокрема, цей параметр вказує на ефективність відстеження інтенсивних динамічних процесів, таких як вібраційні навантаження з частотою порядку 1200 Гц.

Даний параметр безпосередньо залежить від частоти квантування інтегрованого аналого-цифрового перетворювача (АЦП). Згідно з теоремою Котельникова (Найквіста), для коректної реєстрації швидкоплинних процесів частота дискретизації повинна щонайменше вдвічі перевищувати смугу пропускання. Таким чином, для сенсора ADXL345 мінімальне значення частоти роботи АЦП становить 3200 Гц.

Процес вибору акселерометра для точної реєстрації динамічних процесів вимагає попереднього аналізу характеристик очікуваного прискорення. Зокрема, для вимірювання ударних навантажень необхідно визначити їх амплітудний діапазон та часову тривалість. Відповідні нормативні дані щодо типових прискорень наведені у спеціалізованій довідковій літературі.

Під час випробувань систем безпеки варто враховувати фізіологічні межі людського організму. Зокрема, критичне перевантаження на рівні 10g призводить до втрати свідомості пілотом, якщо воно триває протягом декількох секунд. Водночас аналогічне за силою навантаження, що діє короткочасно (наприклад, протягом десяти мілісекунд), не становить серйозної загрози навіть для непідготовленої людини.

Під час тестування промислових об’єктів основна увага приділяється не самому факту передачі ударного імпульсу, а безпосередньому відгуку конструкції на зовнішній вплив. Ключовими параметрами аналізу в такому разі виступають резонансна частота системи та швидкість затухання виникаючих коливань. Ключовими критеріями підбору акселерометра є його частотний діапазон та граничні значення прискорень, які пристрій здатний реєструвати. Саме ці параметри визначають відповідність сенсора конкретним умовам експлуатації. Застосування низькочастотних акселерометрів для

реєстрації швидкоплинних процесів призводить до часових спотворень у вихідних графіках. Водночас датчики з розширеним частотним діапазоном мають достатню чутливість для точного вимірювання навіть незначних прискорень. Високочастотні мікросхеми MEMS-акселерометрів, розраховані на значні перевантаження (±70g, ±250g та ±500g) із широкою смугою пропускання до 30 кГц, характеризуються високою вартістю та обмеженою доступністю на ринку. Проте в задачах, де амплітуда прискорення сягає від 20 до 500g при відносно низькій частоті процесів (до 400 Гц), використання спеціалізованих низькочастотних сенсорів дозволяє суттєво знизити витрати, оскільки такі компоненти у десятки разів дешевші. Враховуючи сукупність технічних характеристик, для побудови мікроконтролерної системи стабілізації рухомих платформ доцільно обрати MEMS-акселерометр моделі ADXL345. Його параметри роздільної здатності та енергоспоживання повністю відповідають вимогам до подібних систем керування.

Технічні параметри цифрового акселерометра ADXL345. Трьохосьовий сенсор ADXL345 є високоефективним рішенням для вимірювання прискорення, що поєднує низьку енергозатратність та високу точність. Електричні характеристики та живлення: модуль адаптований для роботи в межах 2.0–3.6 В; енергоефективність: пристрій споживає від 25 до 130 мкА в активній фазі, а в режимі глибокого сну (standby) цей показник падає до критично низьких 0,1 мкА. Система вимірювання та роздільна здатність; користувачу доступні гнучкі налаштування точності даних: фіксований режим: робота з роздільною здатністю 10 біт; адаптивний режим (Full Resolution): розрядність автоматично зростає разом із розширенням діапазону прискорення (максимум – 13 біт при навантаженні ±16 g). Стабільна чутливість: крок вимірювання залишається незмінним – 4 mg/LSB для будь-якого обраного ліміту **g**. Інтелектуальні функції: сенсор має вбудовану логіку для автоматичного розпізнавання подій: детекція ударів: ідентифікація одинарних та подвійних механічних поштовхів (тапів); аналіз динаміки: контроль стану активності або спокою об’єкта; безпека: можливість фіксації

моменту вільного падіння. Комунікація та фізична стійкість: інтерфейси підключення: підтримка цифрових шин I2C та SPI (конфігурації на 3 або 4 дроти); система переривань: наявність двох програмованих виводів для гнучкого керування подіями; налаштування: програмна корекція смуги пропускання та вимірювальних діапазонів; надійність: датчик витримує екстремальні механічні перевантаження до 10000 g та стабільно працює при температурах від -40°C до +85°C.

Розпіновка (Pinout) мікросхеми акселерометра ADXL345 представлено на Рис.3.8 , а призначення пінів в Табл.3.6.

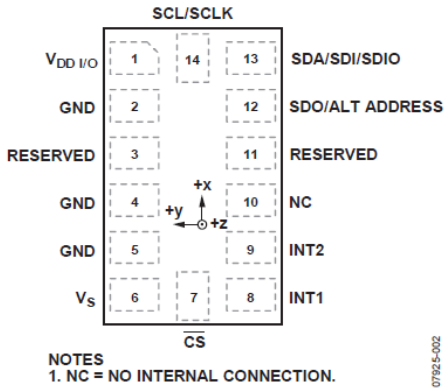


Рис. 3.8. Конструктивне виконання мікросхеми ADXL345 (MEMS-акселерометр)

Таблиця 3.6. Розпіновка (Pinout) мікросхеми ADXL345

№ контакту	Маркування	Технічна характеристика та призначення
1	$V_{DDI/O}$	Шина живлення для портів цифрового вводу-виводу.
2, 4, 5	GND	Спільний провід (заземлення схеми).
3	RESERVED	Резервний вивід: допускається з'єднання з V_s або вільний стан.
6	V_s	Основний вхід для подачі робочої напруги на датчик.
7	CS	Активація модуля (Chip Select) при низькому логічному рівні.
8	INT1	Перший програмований канал зовнішніх переривань.
9	INT2	Другий програмований канал зовнішніх переривань.
10	NC	Вільний контакт (внутрішнє підключення відсутнє).
11	RESERVED	Резерв: рекомендується заземлити або залишити непідключеним.
12	SDO/ALT ADDR	Вихідний потік даних (SPI) або визначення адреси пристрою (I2C).
13	SDA/SDI/SDIO	Двонаправлена лінія даних для I2C або вхідний канал для SPI.
14	SCL/SCLK	Вхідний сигнал тактування для синхронізації інтерфейсів зв'язку.

Склад внутрішньої архітектури акселерометра ADXL345 охоплює 31 регістр; їхні цифрові адреси та функціональні ролі систематизовано в Табл. 3.7.

Таблиця 3.7. Перелік та функціональний опис регістрів датчика ADXL345

Адреса (Hex)	Назва (Label)	Доступ	Дефолтне значення	Функціональне призначення
0x00	DEVID	R	11100101	Ідентифікатор моделі пристрою (0xE5).
0x01…0x1C	Reserved	R/W	00000000	Резервні комірки пам’яті.
0x1D	THRESH_TAP	R/W	00000000	Поріг детектування удару (одиночного дотику).
0x1E…0x20	OFS(X,Y,Z)	R/W	00000000	Коригування зміщення (offset) за осями X, Y, Z.
0x21	DUR	R/W	00000000	Максимальний час тривалості імпульсу удару.
0x22	Latent	R/W	00000000	Пауза перед можливим подвійним ударом.
0x23	Window	R/W	00000000	Часовий інтервал для реєстрації подвійного удару.
0x24	THRESH_ACT	R/W	00000000	Граничне значення для активації вимірювань.
0x25	THRESH_INACT	R/W	00000000	Поріг для визначення стану спокою (неактивності).
0x26	TIME_INACT	R/W	00000000	Час, після якого пристрій вважається неактивним.
0x27	ACT_INACT_CTL	R/W	00000000	Налаштування осей для моніторингу активності.
0x28	THRESH_FF	R/W	00000000	Поріг спрацьовування при вільному падінні.
0x29	TIME_FF	R/W	00000000	Мінімальна тривалість фази вільного падіння.
0x2A	TAP_AXES	R/W	00000000	Вибір осей, за якими відстежуються удари.
0x2B	ACT\TAP_STS	R	00000000	Регістр стану подій активності та ударів.
0x2C	BW_RATE	R/W	00001010	Контроль частоти вихідних даних (ODR).
0x2D	POWER_CTL	R/W	00000000	Управління режимами живлення та сну.
0x2E	INT_ENABLE	R/W	00000000	Дозвіл генерації конкретних переривань.
0x2F	INT_MAP	R/W	00000000	Розподіл сигналів переривань на фізичні виводи.
0x30	INT_SOURCE	R	00000010	Прапорці-джерела виникнення переривань.
0x31	DATA_FORMAT	R/W	00000000	Вибір діапазону (g) та формату представлення даних.
0x32…0x37	DATA(X,Y,Z)	R	00000000	Результати вимірювань

				прискорення (старший/молодший байт).
0x38	FIFO_CTL	R/W	00000000	Конфігурація буфера FIFO.
0x39	FIFO_STATUS	R	00000000	Поточний стан заповнення буфера FIFO.

Алгоритм ініціалізації датчика виглядає так: після подачі напруги пристрій автоматично переходить у стан очікування (standby mode). Для гарантованого старту з вихідної позиції розробнику варто програмно скинути прапорець вимірювання (біт D3) у керуючому регістрі `POWER_CTL`. Щоб активувати процес збору даних та перевести акселерометр у робочий режим (measurement mode), необхідно записати логічну одиницю у згаданий біт D3. Повернення до режиму низького енергоспоживання (очікування) здійснюється шляхом повторного скидання цього біта.

Рівень енергоспоживання акселерометра ADXL345 безпосередньо залежить від встановленої частоти дискретизації інтегрованого аналого-цифрового перетворювача (АЦП). Конкретні значення цієї залежності наведено у Табл. 3.8.

Таблиця 3.8. Залежність струму споживання I_{DD} від частоти вибірки (ODR)

Частота вихідних даних (Hz)	Струм споживання (I_{DD}), μA	Режим функціонування
3200	145	Максимальна продуктивність
1600	90	Висока швидкість
800	140	Стандартний режим
400	140	Стандартний режим
200	140	Стандартний режим
100	140	Номінальний режим
50	90	Енергоефективний режим
25	60	Режим низького споживання
12.5	50	Мінімальна активність

Режим Auto Sleep. Енергоефективність системи можна підвищити завдяки функції Auto Sleep, яка автоматично переводить ADXL345 у стан глибокого сну під час відсутності активності. Налаштування цього режиму передбачає встановлення граничних значень у регістрах `THRESH_INACT` (поріг неактивності) та `TIME_INACT` (тривалість спокою). Активація здійснюється шляхом запису логічної одиниці в біти D4 (`AUTO_SLEEP`) та D5

керуючого регістра POWER_CTL. За таких умов, при частоті дискретизації 12,5 Гц та напрузі живлення 2,5 В, рівень енергоспоживання знижується до мінімальних 23 мкА.

Режим Standby. Для мінімізації енерговитрат пристрою передбачено режим очікування (Standby Mode), у якому споживання струму скорочується до критично низького рівня – 0,1 мкА. У цьому стані процес зняття показників призупиняється, проте дані, що вже знаходяться у буфері FIFO, залишаються незмінними. Перехід до цього режиму здійснюється шляхом деактивації (скидання) біта D3 у керуючому регістрі POWER_CTL.

Інтерфейс ADXL345. Взаємодія з акселерометром ADXL345 здійснюється через протоколи I2C або SPI, при цьому пристрій завжди функціонує як ведомий (Slave). Вибір конкретного інтерфейсу залежить від стану виводу CS:

- Режим I2C: Активується шляхом подачі високого логічного рівня (VDD I/O) на пін CS.
- Режим SPI: У цьому випадку лінія CS контролюється безпосередньо ведучим пристроєм (Master) для ініціалізації сеансу зв'язку.

Для забезпечення стабільної роботи акселерометра при високих частотах дискретизації (3200 Гц та 1600 Гц) критично важливо використовувати шину SPI з тактовою частотою не менше 2 МГц. Якщо ж частота оновлення даних становить 800 Гц, оптимальною швидкістю обміну буде 400 кГц. Параметри для інших режимів роботи можна розрахувати на основі прямої пропорційної залежності.

Для активації протоколу I2C у сенсорі ADXL345 необхідно з’єднати вивід CS із лінією живлення VDD I/O. Пристрій сумісний зі специфікаціями Standard (100 кГц) та Fast (400 кГц). Мережева адреса акселерометра залежить від стану піна ALT ADDRESS. Якщо на ALT ADDRESS подано високий рівень (“1”), 7-бітна адреса пристрою набуває значення 0x1D. З урахуванням біта напрямку передачі (R/W), повна 8-бітна адреса для запису даних становитиме 0x3A, а для читання – 0x3B. Конфігурація адресації ADXL345 за протоколом

I2C може бути змінена шляхом заземлення контакту ALT ADDRESS. У такому випадку пристрій отримує базовий 7-бітний ідентифікатор 0x53. З урахуванням керуючого біта напрямку даних (R/W), підсумкові 8-бітні значення будуть такими: 0xA6 – для виконання операцій запису; 0xA7 – для зчитування даних. При використанні I2C-інтерфейсу важливо, щоб виводи CS та ALT ADDRESS мали чітко визначений потенціал: їх слід під'єднати або до шини живлення VDD I/O, або до загального проводу (GND), як показано на схемі (Рис. 3.9).

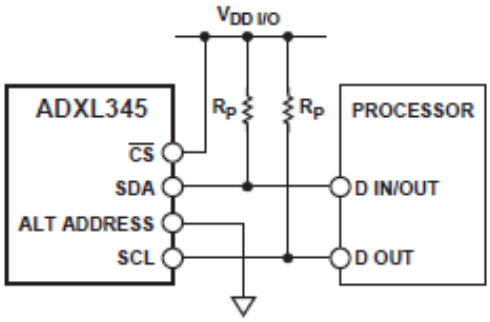


Рис. 3.9. Типова схема з'єднання компонентів для використання інтерфейсу I2C у модулі ADXL345

Переривання. Для взаємодії з мікроконтролером ADXL345 використовує два програмовані двотактні виводи переривань. Розробнику доступне налаштування наступних тригерів: готовність нових даних (DATA_READY), реєстрація ударів (TAP), моніторинг активності (ACTIVITY/INACTIVITY), детектування вільного падіння (FREE_FALL), а також службові прапорці стану буфера (WATERMARK та OVERRUN).

Стандартно для виходів переривань ADXL345 встановлено високий активний рівень. Проте розробник має можливість інвертувати цю логіку на низьку, встановивши біт INT_INVERT у регістрі DATA_FORMAT у стан "1". Важливою особливістю датчика є можливість паралельної роботи всіх доступних функцій переривання.

Для активації конкретних переривань необхідно встановити відповідні біти в регістрі INT_ENABLE. Розподіл цих сигналів між фізичними виходами INT1 та INT2 здійснюється через конфігурацію регістра INT_MAP. Особливістю роботи пристрою є те, що під час зчитування значень із регістрів

даних генерація нових переривань тимчасово блокується, щоб уникнути конфліктів.

Біт DATA_READY активується у разі появи свіжих результатів вимірювань у регістрах пристрою. Щойно дані зчитані або нові значення перестають надходити, цей біт автоматично скидається.

Прапорець SINGLE_TAP активується, коли пристрій фіксує поодинокий імпульс прискорення, амплітуда якого перевищує поріг, заданий у регістрі THRESH_TAP. При цьому тривалість такого імпульсу повинна бути меншою за часовий інтервал, визначений у регістрі DUR.

Прапорець DOUBLE_TAP активується, якщо після реєстрації першого удару та завершення паузи, визначеної в регістрі LATENT, фіксується повторний імпульс. Друга подія повинна відбутися в межах часового вікна, заданого в регістрі WINDOW. При цьому до другого поштовху застосовуються ті ж вимоги, що й до першого: його амплітуда має бути вищою за поріг THRESH_TAP, а тривалість – меншою за параметр DUR.

Біт ACTIVITY активується, коли виміряне прискорення перевищує порогову межу, встановлену в регістрі THRESH_ACT. Конкретні осі координат, які братимуть участь у детектуванні руху (або його відсутності), програмуються за допомогою відповідних налаштувань у регістрі ACT_INACT_CTL.

Реєстрація стану спокою (INACTIVITY) відбувається, якщо рівень прискорення за всіма вибраними осями залишається нижчим за поріг THRESH_INACT. Ця умова має виконуватися протягом часу, що перевищує інтервал, заданий у регістрі TIME_INACT. Варто зауважити, що гранична тривалість періоду очікування для цього параметра становить 255 секунд.

Реєстрація стану вільного падіння (FREE_FALL) відбувається, коли сумарне прискорення по всіх трьох осях (логічне AND) одночасно падає нижче порогу, заданого в регістрі THRESH_FF. Для активації переривання цей стан має тривати безперервно протягом часу, що перевищує значення,

Індикатор WATERMARK активується, коли кількість збережених результатів вимірювань у буфері FIFO досягає рівня, встановленого в регістрі FIFO_CTL. Цей прапорець слугує сигналом для мікроконтролера про необхідність зчитування даних. Після того як обсяг даних у буфері стане меншим за заданий поріг внаслідок читання, біт WATERMARK автоматично скидається.

Біт OVERRUN вказує на те, що нові дані надійшли раніше, ніж були оброблені попередні. У режимі прямого доступу (Bypass) це стосується регістрів даних, в інших режимах — повного заповнення черги FIFO. Прапорець знімається автоматично після читання даних. Для детальної діагностики події, що викликала запит на переривання, використовується регістр INT_SOURCE.

Використання FIFO-буфера. Для зниження навантаження на головний процесор та забезпечення проміжного зберігання даних, архітектура ADXL345 передбачає наявність 32-рівневого стека пам’яті типу FIFO (First In – First Out) для кожної з трьох осей. Цей механізм працює за принципом черги або кільцевого буфера. Розробник має можливість обрати один із чотирьох доступних режимів функціонування буфера: Bypass: буфер не використовується; FIFO: стандартне накопичення даних до заповнення; Stream: безперервне оновлення даних (нові значення витісняють найстаріші); Trigger: режим, що активується за певною подією. Вибір конкретного режиму здійснюється через регістр FIFO_CTL. Конфігурація залежить від значень, записаних у старші біти – D7 та D6 (поле FIFO_MODE).

Режим обходу (Bypass Mode). При активації режиму Bypass функціонал буфера FIFO повністю деактивується. У цьому стані пам’ять пристрою не задіяна для накопичення даних і постійно залишається порожньою. Дані вимірювань передаються безпосередньо, минаючи чергу зберігання.

Режим FIFO. У даному режимі результати вимірювань за трьома осями (x, y, z) послідовно накопичуються в буфері. Система генерує сигнал переривання WATERMARK, коли кількість записів у пам’яті збігається із

лімітом, встановленим у регістрі FIFO_CTL. Після повного заповнення стека (досягнення 32 відліків на кожну вісь) процес запису нових даних припиняється. При цьому сам акселерометр не зупиняє роботу, проте свіжа інформація в буфері вже не фіксується. Стан переривання WATERMARK залишається активним до моменту, поки обсяг даних у FIFO не впаде нижче встановленого порогу.

Режим Stream (потоковий). Функціонування цього режиму ідентичне алгоритму FIFO з тією лише різницею, що при повному заповненні буфера процес запису не припиняється. Нові результати вимірювань автоматично заміщують найстаріші дані, забезпечуючи безперервне оновлення інформації в пам’яті за принципом кільцевого буфера.

Режим Trigger. У цьому режимі буфер FIFO функціонує як циклічне сховище для 32 останніх результатів вимірювань по всіх трьох осях (x, y, z). Запис триває до моменту фіксації певної події, що активує переривання на виходах INT1 або INT2. Вибір конкретного виходу для сигналу здійснюється за допомогою тригер-біта в регістрі `FIFO_CTL`.

Алгоритм роботи FIFO. У цьому режимі буфер накопичує результати останніх вимірювань, обсяг яких визначається параметрами вибірок у регістрі `FIFO_CTL`. Після заповнення встановленого об’єму система діє за принципом класичної черги, припиняючи запис нових даних після переповнення пам'яті. Важливо забезпечити часовий інтервал щонайменше 5 мкс між моментом події та початком зчитування. Така пауза необхідна для коректної фіналізації процесів у буфері та надійного збереження отриманих значень.

Отримання результатів вимірювання. Щоб отримати результати вимірювань за всіма трьома осями (X, Y, Z), слід виконати послідовне зчитування шести байтів даних. Процес починається з регістра `DATAX0` (адреса 0x32) і охоплює наступні за ним регістри: `DATAX1`, `DATAY0`, `DATAY1`, `DATAZ0` та `DATAZ1`.

Конфігурація діапазону вимірювань здійснюється через регістр `DATA_FORMAT`. Оптимальним вибором є активація 13-бітного режиму

(шляхом встановлення біта D3), оскільки він забезпечує стабільну роздільну здатність 3,9 mg/LSB незалежно від обраного діапазону. Такий підхід особливо ефективний, коли динамічний діапазон прискорень заздалегідь невідомий, оскільки точність залишається незмінною.

З метою спрощення практичної реалізації розроблені готові модулі, які поєднують у собі чип акселерометра та стандартизований інтерфейсний роз'єм для підключення (Рис. 3.10).

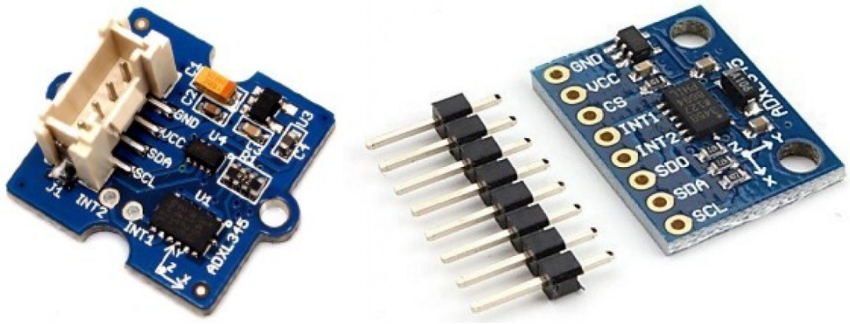


Рис. 3.10. Зовнішній вигляд готових апаратних модулів на базі сенсора ADXL345

Опис та вивід значень на рідкокристалічний дисплей РКД (LCD). Використання РК-модулів є економічно вигідним та практичним способом візуалізації стану системи. Вони дозволяють суттєво скоротити часові та матеріальні витрати під час проектування нових пристроїв, забезпечуючи при цьому високу інформативність за умови мінімального енергоспоживання. Завдяки наявності підсвічування РК-модулі можна ефективно використовувати в умовах недостатнього або відсутнього зовнішнього освітлення. Широкий робочий діапазон температур (від -20°C до +70°C) дозволяє застосовувати їх у складних середовищах, зокрема в мобільних, польових та бортових системах керування. Конструкція рідкокристалічного модуля базується на поєднанні РК-панелі та керуючого контролера. Пристрої цього типу зазвичай оснащені LED-підсвіткою та здатні виводити текстову інформацію у декілька рядків. На Рис.3.11 представлено зовнішній вигляд модуля формату 16x2, який забезпечує відображення 16 символів у кожному з двох рядків. Кожен знак формується матрицею розміром 5x8 точок, а для

чіткості візуалізації між символами передбачені інтервали завширшки в один піксель.

Рис. 3.11. Рідкокристалічний модуль WH1602B-YGK-CTK. Конфігурація: 2 рядки по 16 знаків, колірна схема – жовто-зелена

Модель WH1602B-YGK-CTK – це алфавітно-цифровий РК-дисплей, що забезпечує візуалізацію чорних символів на контрастному жовто-зеленому фоні. До стандартного набору компонентів додається штирьовий з'єднувач (“гребінка”) для швидкої інтеграції в макетні плати, потенціометр номіналом 10 кОм для точного налаштування контрастності, а також обмежувальний резистор на 10 Ом для захисту світлодіодного підсвічування.

- формат виводу: 2 рядки по 16 знакомісць у кожному;
- програмний інтерфейс: побудований на базі контролера, сумісного зі стандартом HD44780;
- тип підсвітки: світлодіодна (LED);
- колірна гама: жовто-зелений фон та підсвічування.

Принцип відображення інформації базується на відповідності між кодом символу та його адресою в оперативній пам’яті (ОЗП) модуля. Внутрішня структура пристрою включає два типи пам’яті: сховище кодів символів та область знакогенератора, що програмується користувачем. Крім того, модуль оснащений спеціалізованою логікою для керування рідкокристалічною матрицею.

Таблиця 3.9. Функціональні особливості РК-модуля WH1602B-YGK-CTK

Функціональна особливість	Опис можливостей
Знакогенератор	Дві сторінки (кирилиця/латиниця) з перемиканням через ПЗ.
Режими шини	4-бітна або 8-бітна передача даних.
Робота з пам’яттю	Повний доступ до ОЗП (запис, читання даних та прапорців стану).
Користувацькі символи	Створення до 8 унікальних графічних знаків.
Курсор	Підтримка двох видів курсора з опцією мерехтіння.

Оптимізація дисплея	Керування параметрами підсвітки та чіткості (контрасту).

Використання алфавітно-цифрових РК-модулів є економічно вигідним та практичним підходом до візуалізації даних. Вони дозволяють оптимізувати процес проектування нових пристроїв, забезпечуючи високу інформативність дисплея за умови мінімальних витрат електроенергії.

Програмування та керування РК-дисплеєм. Перш ніж аналізувати алгоритми керування РК-дисплеєм, доцільно вивчити внутрішню архітектуру контролера HD44780. Це дозволить глибше зрозуміти базові принципи функціонування та конструктивні особливості модулів, побудованих на цій платформі. Решта компонентів контролера, як-от знакогенератор, блоки формування курсора, регістри зсуву та драйвери, не потребують безпосереднього керування програмою, оскільки вони функціонують в автоматичному режимі регенерації зображення. Взаємодія з контролером здійснюється через системний інтерфейс, де ключовими елементами є регістри DR (даних) та IR (команд). Вибір конкретного регістра визначається станом сигнальної лінії RS: рівень логічного нуля (RS=0) активує регістр команд, тоді як логічна одиниця (RS=1) перемикає доступ на регістр даних. Через регістр даних (DR) інформація спрямовується або до відеопам’яті (DDRAM), або до області знакогенератора (CGRAM). Конкретний напрямок залежить від поточного режиму роботи, а запис здійснюється за адресою, яку визначає внутрішній лічильник (AC). Водночас дані, що надходять до регістра інструкцій (IR), обробляються декодером команд для виконання відповідних операцій керування.

Об’єм відеопам’яті (DDRAM) становить 80 байтів і використовується для зберігання кодів символів, що виводяться на екран. Структурно пам'ять організована у вигляді двох рядків по 40 комірок кожен. Важливою особливістю є те, що незалежно від фізичної конфігурації конкретного РК-модуля (наприклад, 8×1 чи 20×4), логічна адресація відеопам’яті завжди залишається незмінною: два віртуальні рядки по 40 символів.

Контролер функціонує за принципом динамічної індикації, забезпечуючи циклічне оновлення даних на екрані. Структурно РК-панель представлена у вигляді матриці, конфігурація якої залежить від обраного режиму: це можуть бути 8 ліній (один рядок, матриця символу 5×7), 11 ліній (один рядок, матриця 5×10) або 16 ліній (два рядки, матриця 5×7). Кожна лінія містить до 200 сегментів, що відповідає максимальній довжині рядка у 40 символів.

Вбудований драйвер контролера HD44780 оснащений лише 40 сегментними виходами (SEG1...SEG40), що дозволяє йому самостійно обслуговувати дисплеї об’ємом до 8 символів. Відповідно, модулі формату 8×2 і менше базуються на одному чипі. Для екранів з більшою кількістю знакомісць використовуються додаткові драйвери (наприклад, HD44100), кожен з яких забезпечує керування ще 40 сегментами. Крім того, архітектура HD44780 передбачає систему внутрішніх прапорців, що керують режимами функціонування окремих вузлів контролера.

Середовище розробки CodeVisionAVR містить інтегровану бібліотеку, що суттєво спрощує підключення таких індикаторів. Програмно підтримуються різноманітні формати дисплеїв: від 1x8 до 2x40 символів. Типова схема з’єднання LCD-модуля з мікроконтролером ATmega128 представлена на Рис. 3.12.

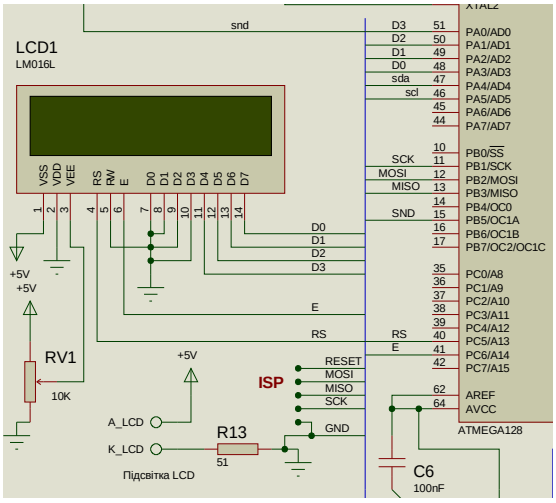


Рис. 3.12. Схема електрична принципова підключення символьного РК-модуля до мікроконтролера ATmega128

3.2. Проєктування системи стабілізації рухомих платформ колісного та гусеничного розміновуючого транспорту в САПР Proteus VSM

Проектована система має забезпечувати моніторинг прискорень за трьома осями (X, Y, Z) та визначати кути нахилу відносно вихідного положення рівноваги. Функціонал пристрою передбачає генерацію керуючих сигналів для драйверів двигунів, можливість фіксації та підтримки заданого кута платформи, а також візуалізацію поточних параметрів на рідкокристалічному дисплеї (РКД). Запропонована архітектура системи стабілізації для мобільних платформ розмінувальної техніки (як колісного, так і гусеничного типу) представлена на Рис. 3.13.

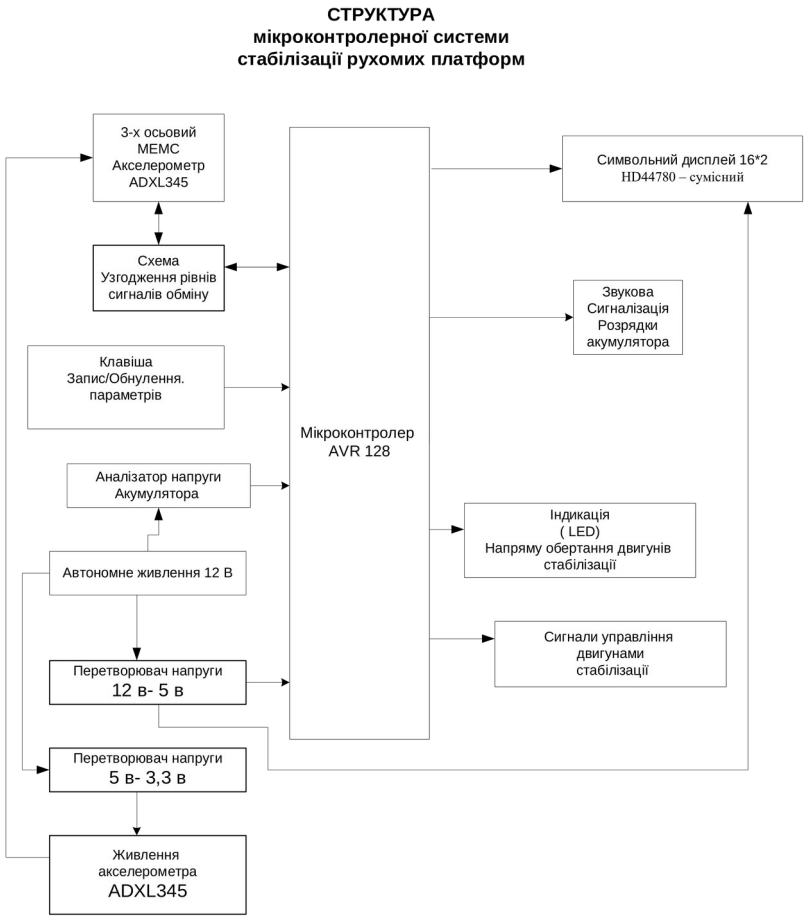


Рис. 3.13. Узагальнена структурна схема системи стабілізації рухомих платформ для засобів розмінування на колісному та гусеничному ходу

Дана розробка базується на інтеграції сенсорних вузлів та маніпуляторів, об’єднуючи в собі програмні алгоритми та апаратні компоненти [11, 12].

До складу апаратної частини системи стабілізації платформ спецтранспорту для розмінування входять наступні вузли та блоки:

- *Центральний обчислювальний модуль:* на базі мікроконтролера архітектури AVR (ATmega128), що здійснює основну обробку вхідної інформації.
- *Сенсорна підсистема:* представлена тривісним МЕМС-акселерометром, який забезпечує збір масиву даних про лінійні прискорення за координатами X, Y, Z для їх подальшого аналізу мікроконтролером.
- *Інтерфейс користувача:* включає графічний РК-дисплей для візуалізації результатів вимірювань, а також блок клавіш для конфігурування та збереження робочих параметрів системи.
- *Система енергозабезпечення:* складається з модуля автономного живлення, а також допоміжних схем моніторингу стану акумулятора (контроль заряду та індикація критичного рівня розряду).
- *Комунікаційна обв'язка:* спеціалізована схема узгодження рівнів для забезпечення коректного обміну сигналами між цифровим акселерометром та МК.

В основі архітектури вимірювального модуля лежить цифровий тривісний акселерометр ADXL345. Розрахунок кутових відхилень мобільної платформи відносно базової системи координат здійснюється шляхом аналізу проєкцій вектора гравітаційного прискорення на три взаємно перпендикулярні осі датчика: поздовжню (X), поперечну (Y) та вертикальну (Z). Розроблений алгоритм забезпечує визначення кутів у діапазоні ±180° із гарантованою точністю, де похибка не виходить за межі 1°.

Центральний мікроконтролер ATmega128 здійснює координацію роботи цифрового акселерометра ADXL345, забезпечуючи зчитування та обробку вхідних даних. На основі отриманої інформації МК генерує відповідні керуючі сигнали для силових драйверів двигунів системи стабілізації, одночасно реалізуючи вивід поточних параметрів на рідкокристалічний індикатор (РКД).

Поточні значення кутів відхилення за відповідними осями візуалізуються на рідкокристалічному індикаторі. Функціональна клавіша призначена для фіксації поточного просторового положення як опорної (базової) системи координат, відносно якої надалі здійснюватиметься стабілізація платформи. Процес скидання встановлених значень до початкових (нульових) налаштувань реалізується шляхом повторного натискання на ту саму клавішу. Для ідентифікації режиму роботи з відносними координатами на РКД передбачено виведення спеціального символу $.

Процедура калібрування МК-системи стабілізації передбачає встановлення акселерометра на попередньо підготовлену прецизійну горизонтальну площину. Фіксація датчика здійснюється в положенні, що відповідає нульовим показникам за осями X та Y, тоді як вектор прискорення по осі Z має становити 90° відносно площини горизонту. Центральний МК здійснює обробку отриманих від акселерометра даних про прискорення за координатами X, Y, Z з подальшою візуалізацією результатів на графічному РК-індикаторі. Результати проектування АЗ системи стабілізації для мобільних платформ у середовищі Proteus VSM представлені на Рис. 3.13.

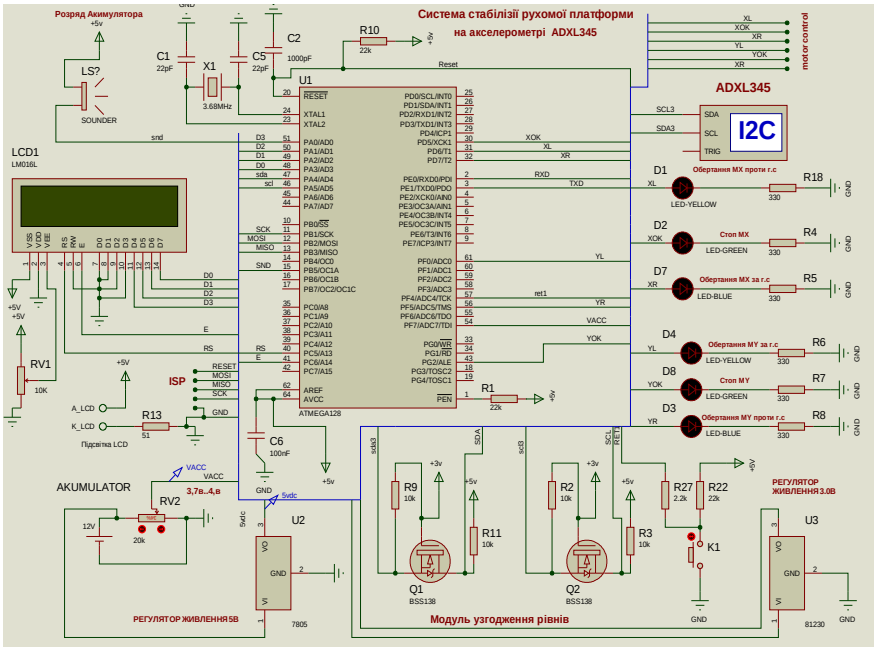


Рис. 3.14. Схемотехнічна реалізація апаратної частини системи стабілізації рухомих платформ розмінувальної техніки, розроблена в середовищі САПР Proteus VSM

У конструкції пристрою застосовано готовий сенсорний модуль на основі акселерометра ADXL345 (Рис. 3.15). Даний модуль оснащений інтегрованим інтерфейсним роз'ємом, що спрощує його інтеграцію в загальну схему системи.

Рис. 3.15. Зовнішній вигляд сенсорного модуля на базі триосьового акселерометра ADXL345

На розробленій схемі (Рис. 3.13) цифровий акселерометр інтегрований до системи через послідовний інтерфейс I2C. Комунікація здійснюється за лініями даних (SDA) та синхронізації (SCL). Оскільки рівні напруги сенсора та МК різняться, застосовано двонаправлений вузол узгодження рівнів, реалізований на польових транзисторах Q1 та Q2, що формують сигнали SCL3 та SDA3.

Система енергозабезпечення побудована на використанні двох рівнів напруги: 5 В для функціонування мікроконтролера ATmega128 та 3,3 В для живлення сенсорного модуля акселерометра. Для забезпечення сумісності компонентів у схему інтегровано додаткові стабілізатори напруги та вузли узгодження логічних рівнів. Обмін даними по шині I2C здійснюється через порти введення-виведення мікроконтролера, де лініям синхронізації (SCL) та даних (SDA) відповідають піни PA5 та PA4. У стандартній апаратній реалізації ATmega128 апаратна шина TWI (I2C) зазвичай закріплена за пінами PD0 (SCL) та PD1 (SDA).

Взаємодія МК з периферійними пристроями реалізована через розподіл функцій між портами введення-виведення. Візуалізація даних забезпечується рідкокристалічним дисплеєм (РКД), підключеним до ліній портів A та C. Для оперативного керування та фіксації (коригування) вимірюваних параметрів

акселерометра використано функціональну клавішу K1, під'єднану до піна PF4. Моніторинг стану автономного джерела живлення (AKUMULATOR) здійснюється за допомогою вбудованого АЦП через вхід PF7, що дозволяє контролювати рівень розряду батареї. Акустична індикація реалізована за допомогою п’єзовипромінювача (LS), підключеного до виходу OC1A (PB5), що дозволяє використовувати апаратний ШІМ для генерації звукових сигналів.

Формування рівнів напруги живлення +5 В та +3,3 В забезпечується завдяки інтегральним стабілізаторам типів 7805 та 81230. Керування світлодіодною індикацією та іншими виконавчими механізмами здійснюється через лінії введення-виведення портів PD, PF та PG.

Для візуального контролю роботи двигуна стабілізації по осі X передбачена світлова індикація. Світлодіод D1 (жовтого кольору) сигналізує про обертання привода проти годинникової стрілки, тоді як D3 (синій) активується при русі за годинниковою стрілкою. Стан спокою, що свідчить про успішну стабілізацію платформи за цією віссю, відображається зеленим світлодіодом D2.

Аналогічну функцію виконують світлодіоди D4, D5 та D6, що відповідають за моніторинг стану привода стабілізації по осі Y. Стан повної фіксації платформи у площині XOY ідентифікується одночасною активацією світлодіодів D2 та D8. Для передачі керуючих сигналів на виконавчі механізми (двигуни) у схемі передбачено спеціалізований інтерфейсний роз’єм “motor control”.

Для забезпечення можливості внутрішньосхемного програмування та оновлення мікрокоду системи, на окремий роз’єм ISP виведено відповідні шини: SCK, MOSI, MISO та лінію RESET. Це дозволяє здійснювати прошивку мікроконтролера без його демонтажу з плати.

РОЗДІЛ 4
ПРОГРАМНО-АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ СТАБІЛІЗАЦІЇ РУХОМИХ ПЛАТФОРМ

4.1. Алгоритм визначення кутового відхилення рухомої платформи

Вимірювання просторової орієнтації та кутового відхилення об’єктів щодо вектора земного тяжіння виконується за допомогою пристроїв, відомих як інклінометри. Функціонування таких систем може базуватися на різних фізичних явищах. Найбільш розповсюдженими методами визначення нахилу є використання гравітаційної складової, аналіз параметрів геомагнітного поля, експлуатація гіроскопічного ефекту або впровадження методів непрямих (опосередкованих) обчислень. Кожна з наведених концепцій характеризується певним набором переваг та обмежень, що зумовлюють специфіку їхнього практичного застосування.

Розглянемо методику встановлення кутового положення об’єкта на основі аналізу гравітаційного вектора. У статичних умовах, коли на систему впливає виключно сила тяжіння, для розрахунку кута нахилу доцільно застосовувати MEMS-акселерометр. Цей тип сенсора фіксує проєкцію прискорення вільного падіння на власну вимірювальну вісь [13]. На основі отриманих даних виконується подальше обчислення кутових координат об’єкта.

В реальних умовах експлуатації на об’єкт, окрім гравітаційного вектора, впливає низка сторонніх чинників: вібрації, механічні удари та відцентрові сили, зумовлені обертанням. Оскільки акселерометр реєструє сумарне прискорення, наявність цих додаткових динамічних компонентів викривлює вихідний сигнал датчика. Як наслідок, використання стандартних алгоритмів розрахунку призводить до виникнення суттєвої похибки при визначенні кута нахилу.

Одноосьовий датчик прискорення. Розглянемо випадок, коли вимірювальна вісь X пристрою постійно збігається з площиною дії вектора

земного тяжіння. У такій системі координат математичний вираз для розрахунку проєкції гравітаційного прискорення на зазначену вісь матиме вигляд:

$$A_x = g \cdot \sin(\alpha) \tag{4.1}$$

де α – кут відхилення чутливої осі акселерометра відносно лінії горизонту; g ≈ 9,8 м/с^2 – прискорення вільного падіння.

У системі координат приладу за площину горизонту приймають поверхню, що є ортогональною відносно вектора сили тяжіння (див. рис. 4.1). Оскільки вихідний сигнал акселерометра має пряму тригонометричну залежність від синуса кута відхилення в гравітаційному полі, математичний вираз для визначення кута нахилу набуває вигляду:

$$\alpha = \arcsin\left(\frac{A_x}{g}\right) \tag{4,2}$$

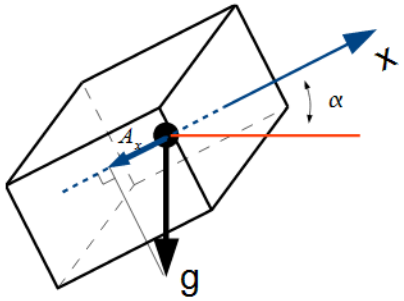


Рис. 4.1. Одноосьовий датчик прискорення

Проаналізуємо характер зміни проєкції A_x залежно від величини кута нахилу. Показник чутливості інклінометра визначається як відношення приросту вихідного сигналу до відповідного приросту вимірюваного кута. В одноосьових системах спостерігається суттєва нелінійність: при значеннях кута, що наближаються до 90°, значні кутові переміщення викликають лише незначні варіації прискорення. Внаслідок цього в околі 90° чутливість приладу різко падає, прямуючи до нуля.

Одним із ключових параметрів інклінометра є поріг його чутливості, що визначається як мінімальне кутове відхилення, яке здатна зафіксувати вимірювальна система. Оскільки поріг чутливості самого акселерометра (як первинного перетворювача) є величиною сталою, то в перерахунку на кутові

одиниці поріг чутливості інклінометра стає змінним параметром. Це зумовлено нелінійною залежністю між проєкцією прискорення та кутом нахилу.

Принцип підбору датчика базується на здатності акселерометра реєструвати варіацію проєкції сили тяжіння, що відповідає заданому порогу чутливості інклінометра. Різниця між двома послідовними показниками приладу при зміні кутового положення на величину приросту Δα визначається за наступною залежністю:

$$\Delta A_x = g \cdot \left[\sin(\alpha + \Delta\alpha) - \sin(\alpha) \right] \tag{4.3}$$

де α – поточне значення кута нахилу; Δα – крок приросту кута (поріг чутливості). Побудуємо зміну різниці залежно від кута нахилу та показника приросту.

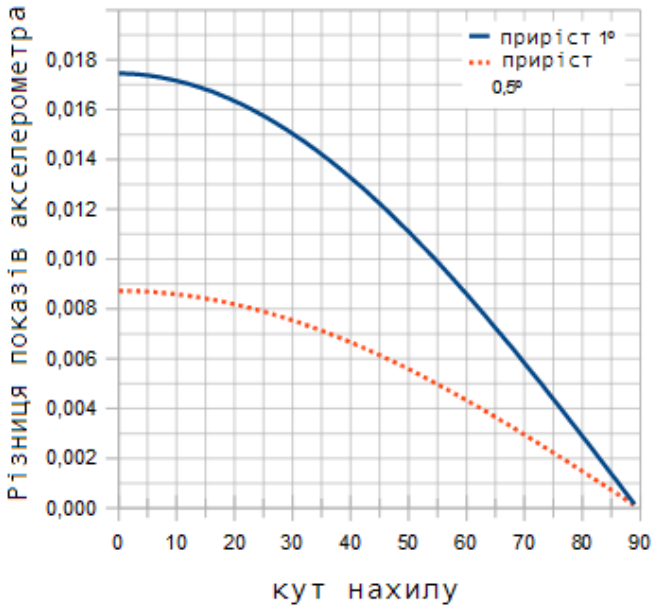


Рис. 4.2. Графічне представлення зміни різниці залежно від кута нахилу та показника приросту

Виявлена закономірність дозволяє встановити граничне значення роздільної здатності акселерометра, необхідне для досягнення цільового порогу чутливості. Зокрема, аналіз графіка показує: для забезпечення чутливості у 0,5° в межах кутів ±55° доцільно використовувати датчик із характеристикою 5 mg/LSB. Слід зауважити, що одноосьові моделі обмежені у

роботі в повному діапазоні 0° – 360°, оскільки синусоїдальна залежність дає ідентичні результати для кутів N° та 180° – N°.

Двохосьовий датчик прискорення. Для розширення вимірювальних можливостей у систему впроваджується друга вісь чутливості – Y. Вона розташована ортогонально відносно осі X і перебуває безпосередньо в площині впливу вектора сили тяжіння (Рис. 4.3).

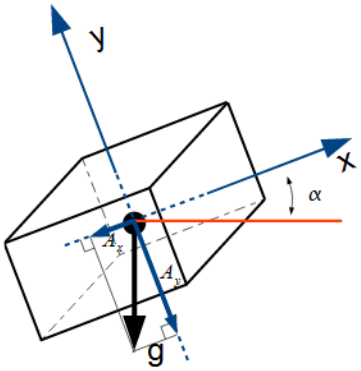


Рис. 4.3. Конфігурація вимірювальних осей двохосьового акселерометра

За аналогією з одноосьовою моделлю, компоненти прискорення вздовж осей X та Y визначаються через тригонометричні функції: проєкція на вісь X пропорційна синусу кута нахилу, тоді як на вісь Y – його косинусу. Враховуючи математичні властивості цих функцій, спостерігається ефект взаємної компенсації: при зниженні чутливості за однією з координат відбувається її закономірне зростання за іншою.

Обчислення кута нахилу здійснюється за допомогою виразів:

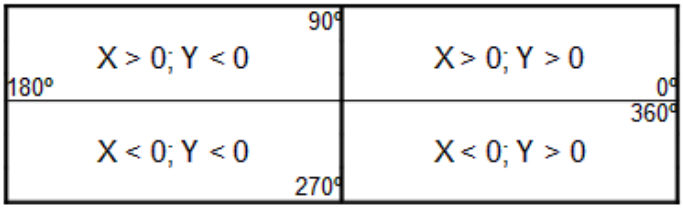
$$\tan(\alpha) = \frac{A_x}{A_y} \tag{4.4}$$

$$\alpha = \arctan\left(\frac{A_x}{A_y}\right) \tag{4.5}$$

Завдяки тому, що зростання чутливості вздовж однієї осі супроводжується її еквівалентним зниженням вздовж іншої, сумарну чутливість системи можна прийняти за умовно сталу величину. Таке припущення суттєво полегшує процес підбору акселерометра з відповідними характеристиками роздільної здатності.

Обчислений для конкретного значення кута поріг чутливості залишається актуальним для всього робочого діапазону вимірювань. Слід зауважити, що будь-яке відхилення від основної вимірювальної осі спричиняє суттєві похибки при використанні одноосьових пристроїв. Впровадження другої вимірювальної осі забезпечує високу точність даних навіть за наявності нахилу вздовж третьої осі. Це пояснюється тим, що результуюча чутливість інклінометра визначається як квадратний корінь із суми квадратів проєкцій вектора гравітації на робочі осі сенсора.

За умови, що вектор сили тяжіння діє виключно у площині XY, сумарне прискорення, що реєструється сенсором, становить 1g. Відхилення у площинах XZ або YZ призводять до зменшення вимірюваного значення прискорення, що, як наслідок, знижує роздільну здатність інклінометра. Попри це, зберігається можливість отримання достовірних даних щодо кута нахилу в площині XY, проте лише за умови незначних кутових відхилень у площинах XZ та YZ. Зі збільшенням кута нахилу вплив гравітаційної складової на осі X та Y нівелюється, що унеможливлює точне обчислення кута. Слід зазначити, що використання додаткової осі дозволяє розширити діапазон вимірювань до 360° шляхом аналізу знаків проєкцій відповідно до поточного квадранта.



Квадрант, у якому перебуває кут, визначається шляхом аналізу даних, отриманих з кожної чутливої осі датчика.

Трьохосьовий датчик прискорення. Використання третьої осі чутливості забезпечує можливість визначення повного спектра кутів нахилу об’єкта в тривимірному просторі. За початкових умов пристрій орієнтований так, що осі x та y збігаються з горизонтальною площиною, тоді як вісь z розташована перпендикулярно до них (Рис. 4.4).

На наведеному рисунку позначено кути просторової орієнтації: α та β визначають відхилення осей x та y акселерометра відносно площини горизонту, а γ відображає кут між вектором прискорення вільного падіння та віссю z. У вихідному стані системи значення всіх зазначених кутів дорівнюють нулю.

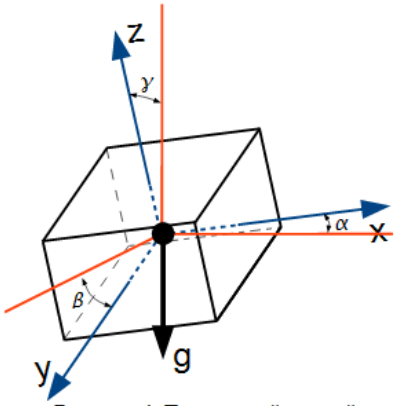


Рис. 4.4. Просторова орієнтація осей та кутів нахилу трикомпонентного акселерометра

За умови, коли вектор гравітації збігається виключно з напрямком осі z, значення всіх кутів нахилу дорівнюватимуть нулю. Математичне визначення цих кутів здійснюється на основі виміряних проєкцій прискорення A_x, A_y та A_z за наступними залежностями:

$$\alpha = \arctan\left(\frac{A_x}{\sqrt{A_y^2 + A_z^2}}\right) \tag{4.6}$$

$$\beta = \arctan\left(\frac{A_y}{\sqrt{A_x^2 + A_z^2}}\right) \tag{4.7}$$

$$\gamma = \arctan\left(\frac{A_z}{\sqrt{A_x^2 + A_y^2}}\right) \tag{4.8}$$

де A_x, A_y, A_z – компоненти прискорення, зафіксовані сенсором уздовж відповідних вимірювальних осей.

Стабільність порогу чутливості забезпечує високу точність вимірювання кутових величин у межах усієї робочої сфери.

Калібрування датчика прискорення. З огляду на механічну природу MEMS-акселерометрів, навіть після заводського налаштування вони зазнають впливу статичного «навантаження». Це зумовлює дрейф нульового рівня та

варіативність коефіцієнта чутливості інклінометра, що безпосередньо знижує метрологічну точність вимірювання кутів. Для мінімізації похибок обчислення орієнтації необхідно проводити процедуру калібрування зсуву нуля та масштабувального коефіцієнта [14-20].

Спеціалізована інструментальна платформа, призначена для калібрування акселерометра, представлена на Рис. 4.5. Ця установка забезпечує прецизійне позиціонування закріпленого датчика на задані кутові величини.

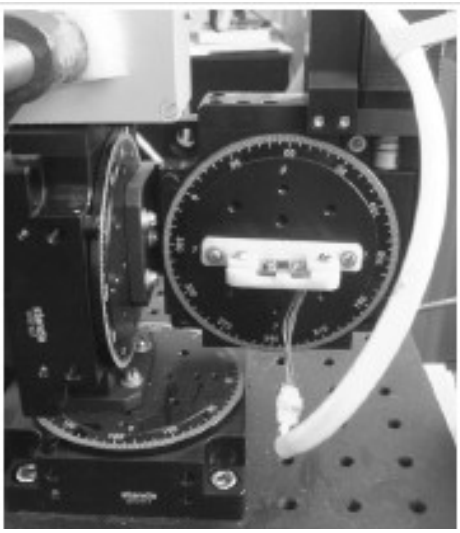


Рис. 4.5. Спеціалізована установка для прецизійного позиціонування акселерометра в процесі калібрування

Процедура калібрування акселерометра може бути реалізована без залучення високовартісного спеціалізованого обладнання. Для визначення необхідних коефіцієнтів достатньо зареєструвати серію вихідних показників датчика в умовах, коли на нього впливає виключно вектор прискорення вільного падіння.

Вихідний сигнал акселерометра з урахуванням адитивної та мультиплікативної складових похибки можна представити у такому вигляді:

$$A_{meas} = A_0 + K \cdot A_d \cdot \cos(\alpha) \tag{4.9}$$

де A_0 – зміщення нульового рівня; K – масштабувальний коефіцієнт (чутливість); A_d – істинне значення прискорення вільного падіння (1g); α – кут

відхилення чутливої осі сенсора від вектора гравітації. Таким чином, процедура первинного калібрування полягає у визначенні параметрів A_0 та K.

Для обчислення зазначених параметрів необхідно зареєструвати вихідні показники акселерометра в чотирьох статичних положеннях. Процедура передбачає послідовний поворот осі чутливості сенсора на кути 0°, 90°, 180° та 270° відносно її вихідної орієнтації. Отримані результати вимірювань доцільно представити у вигляді наступної системи рівнянь:

$$A_1 = A_0 + K \cdot A_Д \cdot \sin(\alpha) \tag{4.10}$$

$$A_2 = A_0 + K \cdot A_Д \cdot \sin(\alpha + \frac{\pi}{2}) \tag{4.11}$$

$$A_3 = A_0 + K \cdot A_Д \cdot \sin(\alpha + \pi) \tag{4.12}$$

$$A_4 = A_0 + K \cdot A_Д \cdot \sin(\alpha - \frac{\pi}{2}) \tag{4.13}$$

Беручи до уваги періодичність тригонометричних функцій, зокрема те, що $\sin(\alpha) = -\sin(\alpha + \pi)$, а $\sin\left(\alpha + \frac{\pi}{2}\right) = -\sin\left(\alpha - \frac{\pi}{2}\right)$,, після підсумовування лівих і правих частин рівнянь (4.10–4.13) маємо:

$$A_0 = \frac{1}{4}(A_1 + A_2 + A_3 + A_4) \tag{4.14}$$

Розрахунок чутливості сенсора проводиться з використанням наступних аналітичних виразів:

$$\sin\left(\alpha + \frac{\pi}{2}\right) = \cos(\alpha) \text{ і } \sin^2(\alpha) + \cos^2(\alpha) = 1$$

Одержимо: $(A_1 - A_3)^2 + (A_2 - A_4)^2 = 4 \cdot K^2 \cdot A_\partial^2 \cdot (\sin^2(\alpha) + \cos^2(\alpha))$

Звідки:

$$K \cdot A_\partial = \frac{1}{2} \cdot \sqrt{(A_1 - A_3)^2 + (A_2 - A_4)^2} \tag{4.15}$$

Запропонована методика калібрування акселерометра відзначається відсутністю вимог до його попереднього позиціювання у просторі, що суттєво полегшує практичну реалізацію процесу. Описаний алгоритм операцій слід застосувати почергово для кожної з вимірювальних осей пристрою.

Процедура калібрування акселерометра ADXL345 передбачає визначення похибок зміщення (Δx, Δy, Δz) для кожної з трьох осей. Отримані розрахункові дані вносяться до спеціалізованих регістрів OFSX, OFSY та OFSZ. Надалі апаратна частина датчика самостійно підсумовує ці коригувальні коефіцієнти з поточними показниками прискорення, що забезпечує автоматичну корекцію результатів вимірювання.

Дискретність молодшого розряду (LSB) для регістрів корекції зміщення становить 15,6 mg/LSB. Розглянемо приклад: при налаштуванні акселерометра на діапазон ±16g із роздільною здатністю 3,9 mg/LSB було отримано відхилення по осі Z у розмірі Δz = 420. У такому разі до регістра *OFSZ* необхідно внести додатковий код (комплементарне число) offsetZ, що відповідає значенню (Δz×3,9/15,6).

$$offsetz = 255 - \left(\Delta z \cdot \frac{3,9}{15,6}\right) = 255 - \left(420 \cdot \frac{3,9}{15,6}\right) = 150 \ (0X96) \qquad (4.16)$$

тут величина 255 слугує показником дискретності для заданої чутливості приладу (3,9 mg/LSB).

Залежно від просторового положення акселерометра, вихідні дані можуть набувати як позитивних, так і негативних значень відносно осей координат (Рис. 4.6). Це зумовлено вектором дії сили тяжіння, що проєктується на відповідні осі чутливості.

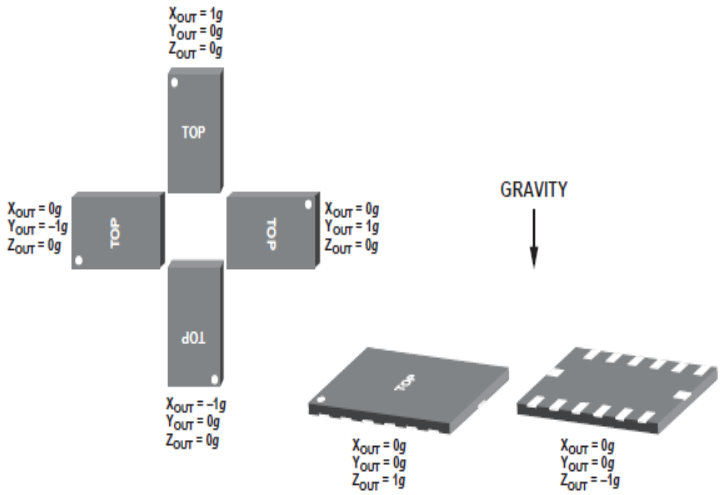


Рис. 4.6. Реакція сенсора на зміну положення відносно вектора гравітації

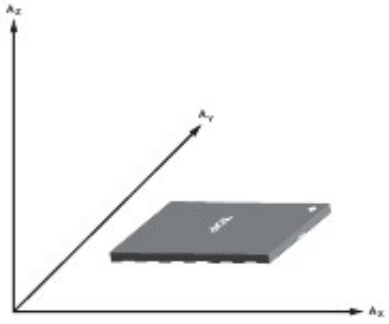


Рис. 4.7. Схема розташування осей чутливості та напрямків вимірювання прискорення для датчика ADXL345

У Табл. 4.1 систематизовано вихідні параметри акселерометра ADXL345 для різних діапазонів вимірювання прискорення та відповідних рівнів роздільної здатності.

Таблиця 4.1. Співвідношення вихідних значень сенсора ADXL345 із налаштуваннями діапазону прискорення та кроком дискретизації

Режим (діапазон та розрядність)	Чутливість (mg/LSB)	Прискорення (g)	Вихідні дані (Dec/Hex)
+/-2g (10 біт)	3,9	+1	256 (0×0100)
+/-2g (10 біт)	3,9	-1	768 (0×0300)
+/-4g (10 біт)	7,8	+1	128 (0×0080)
+/-4g (10 біт)	7,8 mg/LSB	-1	896 (0×0380)
+/-8g (10 біт)	15,6 mg/LSB	+1	64 (0×0040)
+/-8g (10 біт)	15,6 mg/LSB	-1	960 (0×03C0)
+/-16g (10 біт)	31,2 mg/LSB	+1	32 (0×0020)
+/-16g (10 біт)	31,2 mg/LSB	-1	992 (0×03E0)
+/-4g (11 біт)	3,9 mg/LSB	+1	256 (0×0100)
+/-4g (11 біт)	3,9 mg/LSB	-1	1792 (0×0700)
+/-8g (12 біт)	3,9 mg/LSB	+1	256 (0×0100)
+/-8g (12 біт)	3,9 mg/LSB	-1	3840 (0×0F00)
+/-16g (13 біт)	3,9 mg/LSB	+1	256 (0×0100)
+/-16g (13 біт)	3,9 mg/LSB	-1	7936 (0×1F00)

Для розрахунку коефіцієнтів зміщення акселерометр необхідно розмістити на статичній горизонтальній поверхні. При такій орієнтації вісь Z пристрою повинна бути спрямована паралельно вектору земного тяжіння (1g), тоді як проєкції прискорення на горизонтальні осі X та Y мають дорівнювати нулю. Після зчитування виміряних значень зміщення (Δx, Δy, Δz) виконується розрахунок коригувальних параметрів offsetX, offsetY, offsetZ згідно з формулою 4.16. Отримані результати заносяться до відповідних регістрів

OFSX, OFSY та OFSZ. Варто зауважити, що дана методика є найбільш елементарною, проте вона забезпечує лише базовий рівень точності калібрування. Альтернативний підхід до калібрування базується на експериментальному визначенні екстремальних значень (мінімуму та максимуму) для кожного вимірювального каналу. В результаті фіксується шість опорних показників – по два для кожної з трьох осей. Використовуючи отримані величини X_{amin} та X_{amax}, стає можливим розрахунок наступних параметрів:

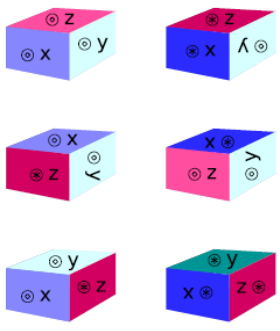
$$\Delta x = 0{,}5 \cdot (X_{amax} + X_{amin}) \tag{4.17}$$

Розглянемо приклад обчислення: припустимо, екстремальні значення по осі X становлять X_{amin}=-250 LSB та X_{amax}=270 LSB. Визначення середнього арифметичного цих величин за формулою (270 + (-250))·0,5 дає результат 10 LSB. Таким чином, зміщення для даної осі становить Δx=10 LSB. За ідентичним алгоритмом розраховуються показники Δy та Δz.

$$\Delta y = 0{,}5 \cdot (Y_{amax} + Y_{amin}) \tag{4.18}$$

$$\Delta z = 0{,}5 \cdot (Z_{amax} + Z_{amin}) \tag{4.19}$$

Для реалізації даної методики необхідно послідовно зорієнтувати вісь X акселерометра спочатку вздовж вектора сили тяжіння, а потім у протилежному напрямку. На основі отриманих екстремальних показників X_{amin} та X_{amax} обчислюється величина зміщення Δx. Ідентичну послідовність маніпуляцій слід виконати для каналів Y та Z з метою визначення відповідних похибок Δy та Δz.



План калібровочних замірів

Орієнтація ADXL345	Ax	Ay	Az
z вісь вверх	0	0	+1g
z вісь вниз	0	0	-1g
y вісь вверх	0	+1g	0
y вісь вниз	0	-1g	0
x вісь вверх	+1g	0	0
x вісь вниз	-1g	0	0

Рис. 4.8. Послідовність етапів калібрування сенсора при різних положеннях у просторі

На основі отриманих граничних значень (X_{amin}, X_{amax}, Y_{amin}, Y_{amax}, Z_{amin}, Z_{amax}) стає можливим визначення чутливості акселерометра для кожного вимірювального каналу. Зокрема, масштабний коефіцієнт $Gain_x$ для осі X розраховується як напівсума абсолютних величин отриманих екстремумів за наступною формулою:

$$Gain_x = \frac{(|X_{amin}| + |X_{amax}|)}{2} \tag{4.20}$$

Розглянемо числовий приклад визначення масштабного коефіцієнта: за умови, що екстремальні значення становлять X_{amin}=-250 та X_{amax}=270, параметр чутливості $Gain_x$ обчислюється як (|-250|+|270|)/2, що дорівнює 210. За аналогічним алгоритмом здійснюється розрахунок показників $Gain_y$ та $Gain_z$ для відповідних осей.

Для визначення дійсного значення прискорення за віссю X з урахуванням обчислених параметрів чутливості та зміщення використовується наступна математична залежність:

$$A_{xreal} = \frac{(A_x - A_{xoff})}{Gain_x} \tag{4.21}$$

де A_x – вихідне прискорення, зафіксоване сенсором за віссю X; A_{xoff} – величина нульового зміщення для даної осі; $Gain_x$ – масштабний коефіцієнт (чутливість) вимірювального каналу X.

Розрахунок дійсних значень прискорення для координатних осей Y та Z виконується за аналогічним алгоритмом:

$$A_{yreal} = \frac{(A_y - A_{yoff})}{Gain_y} \tag{4.22}$$

$$A_{zreal} = \frac{(A_z - A_{zoff})}{Gain_z} \tag{4.23}$$

Розраховані значення A_{xreal}, A_{yreal}, A_{zreal} подаються у відносних одиницях g, де 1g ≈ 9,81 м/с². Це дозволяє безпосередньо використовувати дані для визначення просторової орієнтації об’єкта.

4.2. Алгоритм функціонування системи стабілізації рухомих платформ розміновуючого транспорту

Після подачі напруги живлення активується виконання вбудованого програмного забезпечення (ПЗ) мікроконтролера архітектури AVR [21-26]. На початковому етапі здійснюється ініціалізація внутрішніх програмних змінних, після чого стартує процес конфігурування сенсорів та периферійних модулів. Програма виконує налаштування аналого-цифрового перетворювача (АЦП), акселерометра, каналу широтно-імпульсної модуляції (ШІМ) для керування звуковим випромінювачем LS, а також рідкокристалічного дисплея (РКД) через 4-бітну шину даних (виводи D0..D3).

Конфігурування РКД здійснюється за допомогою виклику спеціалізованої функції `lcd_init()`. Після успішного завершення процедури налаштування інтерфейсу на екран виводиться сервісне повідомлення “PLATFORM STABILIZATION”. Для забезпечення візуального контролю процесу завантаження повідомлення утримується на дисплеї протягом 2 секунд за допомогою програмної затримки.

Після етапу ініціалізації система переходить до виконання основного нескінченного циклу. Він охоплює послідовне зчитування даних прискорення з акселерометра ADXL345 мікроконтролером, математичну обробку отриманих значень та моніторинг стану кнопки керування K1. Паралельно здійснюється вимірювання напруги акумуляторної батареї з подальшим аналізом рівня її заряду. На основі обробленої інформації на РКД виводяться результати вимірювань та, згідно з логікою алгоритму, формуються керуючі сигнали для приводів системи стабілізації, а також засобів світлової та звукової індикації.

Рис. 4.9. Блок-схема алгоритму функціонування системи стабілізації для рухомої платформи транспортного засобу розмінування

4.3. Розроблення підпрограм для роботи з цифровим трьохосьовим сенсором прискорення ADXL345

Для забезпечення коректної взаємодії з цифровим сенсором прискорень ADXL345 було створено однойменну програмну бібліотеку. Структура цієї бібліотеки охоплює низку функціональних методів, призначених для конфігурування та зчитування даних:

unsigned char get_adxl_reg_byte(unsigned char reg_addr) – читання одного байта з регістра акселерометра ADXL345.

int get_adxl_16bit_val(unsigned char start_addr) - зчитування 16-бітного цілого значення (2 байти) з пари регістрів акселерометра ADXL345.

void set_adxl_reg_byte(unsigned char target_reg, unsigned char config_byte) - записування байта у вибраний регістр датчика ADXL345.

void set_accel_measurement_range(accel_range_t range_setting) - Налаштування діапазону вимірювання та точності (Data Format), range_setting: вибраний діапазон (2g, 4g, 8g або 16g). Функція встановлює режим Full Resolution (4mg/LSB) для забезпечення максимальної точності у будь-якому діапазоні.

void set_sensor_operation_mode(sensor_state_t state) - керування станом активності сенсора (режим вимірювання/очікування). Функціональний стан (ACTIVATE для роботи, SUSPEND для сну). Регістр POWER_CTL (0x2D), біт D3 (Measure) відповідає за роботу датчика.
void configure_adxl_interrupt(unsigned char event_mask, unsigned char is_enabled, unsigned char target_pin) – комплексне налаштування системи переривань ADXL345. Параметри event_mask: тип події (напр., FREE_FALL, SINGLE_TAP, ACTIVITY), is_enabled: ACTIVATE (1) або DEACTIVATE (0), target_pin: призначення на фізичний вивід (INT_PIN_1 або INT_PIN_2).

unsigned char check_interrupt_event(unsigned char event_bit) - перевірка статусу конкретного переривання. Параметри event_bit: бітова маска події (напр., MASK_FREE_FALL, MASK_SINGLE_TAP), повертає 1, якщо подія зафіксована, та 0, якщо ні. Читання регістра INT_SOURCE очищує всі прапорці переривань.

void update_tap_sensitivity(unsigned char threshold_val) – налаштування порогу спрацювання детектора поштовху (Tap Threshold). Параметри threshold_val: рівень чутливості (дискретність 62,5 mg/LSB). Приклад: значення 16 відповідає 1g (16*62,5=1000 mg). Значення 0 забороняє генерацію переривання поштовху.

void set_tap_time_limit(unsigned char time_limit) – налаштування максимального часу тривалості поштовху (DUR). Параметри time_limit: значення часу (дискретність 625 мкс/LSB). Визначає часове вікно, протягом якого прискорення має бути вище порогу THRESH_TAP, щоб подія була розпізнана як поштовх.

void set_tap_latency_period(unsigned char wait_period) – налаштування часового інтервалу затримки (Latency). Параметри wait_period: значення затримки (дискретність 1,25 мс/LSB). Визначає час очікування після першого поштовху, протягом якого будь-які нові події ігноруються перед відкриттям вікна Window.

void set_double_tap_window(unsigned char window_val) – налаштування часового вікна для детекції подвійного поштовху (Window). Параметри window_val: тривалість вікна (дискретність 1,25 мс/LSB) .Функція визначає інтервал часу після завершення Latency, протягом якого датчик очікує другий поштовх для реєстрації DOUBLE_TAP.

void set_activity_trigger_level(unsigned char threshold_val) – налаштування порогу детекції руху (Activity Threshold). Параметри threshold_val: рівень чутливості (дискретність 62.5 mg/LSB). Функція визначає величину прискорення, необхідну для реєстрації події ACTIVITY. Приклад: 16*62,5 mg = 1000 mg (1g).

void set_inactivity_limit(unsigned char threshold_val) - налаштування порогу детекції спокою (Inactivity Threshold). Параметри threshold_val: рівень чутливості (дискретність 62.5 mg/LSB). Якщо прискорення по всіх вибраних осях залишається нижче цього значення протягом часу TIME_INACT, генерується подія неактивності.

void set_inactivity_timer(unsigned char seconds_val) – налаштування часового інтервалу детекції спокою (Time Inactivity). Параметер seconds_val: тривалість очікування в секундах (дискретність 1 сек/LSB), range: 0 - 255 секунд. Функція визначає час, протягом якого прискорення має бути нижчим за поріг INACT_THRESH для реєстрації події неактивності.

void update_accel_averages(unsigned char num_samples) – оновлення усереднених значень прискорення по трьох осях. Параметри num_samples: кількість вимірювань для фільтрації (напр., 10). Функція зчитує сирі дані, підсумовує їх та обчислює середнє значення.

4.4. Розроблення програмного модуля для візуалізації даних на РК-дисплеї

Візуалізація даних на рідкокристалічному дисплеї з конфігурацією символів 16×2 реалізована за допомогою інструментарію спеціалізованої бібліотеки mylcd.c. Даний програмний модуль включає перелік функціональних можливостей наведених у Табл.4.2:

Таблиця 4.2. Функціональні можливості бібліотеки mylcd

Назва функції	Опис та призначення
`lcd_init`	Програмна ініціалізація контролера дисплея перед початком роботи.
`lcd_write`	Передача низькорівневих команд керування (напр., налаштування режиму шини).
`lcd_putchar`	Візуалізація поодинокого символу у поточній позиції.
`lcd_clear`	Повне очищення інформаційного поля РКД та повернення курсора в нульову позицію.
`lcd_putsf`	Читання та виведення рядкових констант, що зберігаються у Flash-пам’яті МК.
`lcd_puts`	Виведення текстових рядків, розташованих в оперативній пам’яті (SRAM).
`lcd_gotoxy`	Встановлення активної позиції (координат x, y) для наступного виведення.
`cursor_on/ off`	Керування видимістю курсора на матриці індикатора.

4.5. Програмна реалізація головного алгоритму функціонування системи стабілізації рухомої платформи розміновуючого транспорту

Програмне забезпечення, що забезпечує функціонування мікроконтролерної системи стабілізації мобільної платформи, розроблене у повній відповідності до алгоритму, наведеного на Рис. 4.9. Повний вихідний текст програми з відповідними коментарями міститься у додатку.

4.6. Моделювання роботи системи стабілізації рухомої платформи в Proteus ISIS

Результатом компіляції вихідного коду в середовищі CodeVisionAVR є об’єктний файл із розширенням .hex, призначений для завантаження в енергонезалежну пам’ять мікроконтролера ATmega128. Верифікація схемотехнічних рішень та програмних алгоритмів здійснювалася за допомогою імітаційного моделювання у програмному комплексі Proteus VSM (ISIS). Результати тестування віртуальної моделі розробленої системи представлені на наступних графічних матеріалах.

Для проєктування та верифікації системи було використано середовище Proteus VSM. Ця САПР дозволяє виконувати наскрізний цикл розробки: від створення принципової електричної схеми в модулі ISIS до проектування фізичного компонування друкованої плати в модулі ARES. Важливою перевагою пакету є підтримка віртуального моделювання роботи МК у реальному часі, що дозволяє відлагодити взаємодію периферійних модулів (РКД, акселерометра) до етапу створення фізичного прототипу.

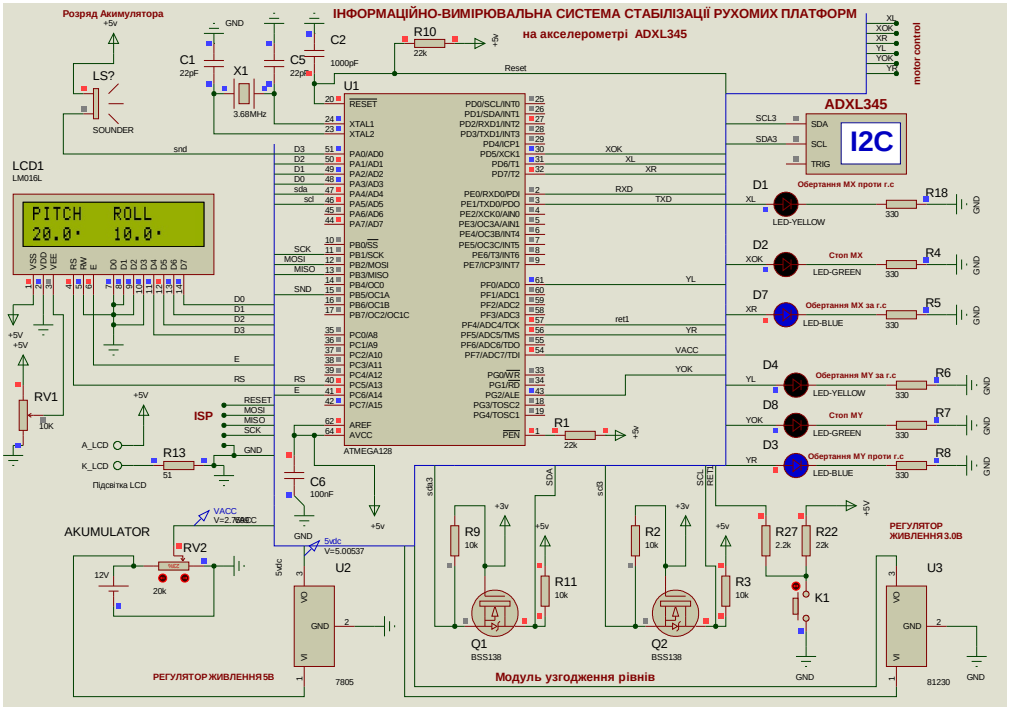


Рис. 4.10. Результати імітаційного моделювання (симуляції) функціонування системи стабілізації в середовищі Proteus ISIS (робочий режим)

У штатному режимі функціонування системи (Рис. 4.10) на екран виводяться поточні кути нахилу: тангаж (PITCH) та крен (ROLL). Ці показники визначають відхилення системи від початкового стану рівноваги та розраховуються за такими математичними залежностями:

$$Pitch = \arctan\left(\frac{x}{\sqrt{z^2+y^2}}\right) \tag{4.24}$$

$$Roll = \arctan\left(\frac{y}{\sqrt{z^2+x^2}}\right) \tag{4.25}$$

На Рис. 4.11 представлено приклади рухомих платформ із зазначенням кутів тангажу (PITCH) та крену (ROLL).

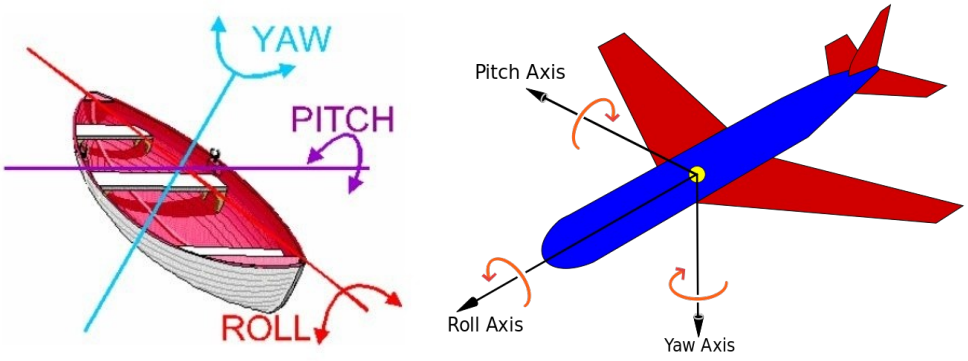


Рис. 4.11. Схематичне зображення рухомих платформ із позначенням кутів тангажу (PITCH) та крену (ROLL)

Поперечна вісь (Pitch): визначає кут тангажу, що відповідає за рух носової частини об’єкта (літака, судна, транспортного засобу) вгору чи вниз. Обертання відбувається навколо осі, яка проходить через центр мас від лівого до правого борту (крила). Поздовжня вісь (Roll): характеризує кут крену. Цей рух передбачає обертання корпусу навколо осі, що з’єднує хвостову частину та ніс транспортного засобу. Вертикальна вісь (Yaw): відповідає за кут рискання (поворот за курсом). Рух здійснюється навколо осі, спрямованої зверху вниз, і забезпечує відхилення носа літака або човна ліворуч чи праворуч від обраного напрямку.

Для візуального моніторингу роботи системи використовуються сині індикатори D7 та D3. Їхнє свічення вказує на активацію приводів стабілізації для обертання валів за годинниковою стрілкою. Паралельно з відображенням даних на РК-дисплеї, відповідні команди спрямовуються у вихідний порт motor control. Процес актуалізації даних у системі здійснюється в безперервному циклі.

Функціональна клавіша K1 відповідає за перемикання між режимами відліку координат. При її натисканні система активує режим відносних координат платформи, що супроводжується появою символу “$” на рідкокристалічному дисплеї (Рис. 4.12). Це слугує візуальним підтвердженням переходу до відносних величин. Повернення до відображення абсолютних кутів відхилення від точки рівноваги здійснюється шляхом повторного натискання тієї ж кнопки. Стан спокою системи, за якого двигуни знеструмлені, контролюється за допомогою зелених світлодіодів D2 та D8.

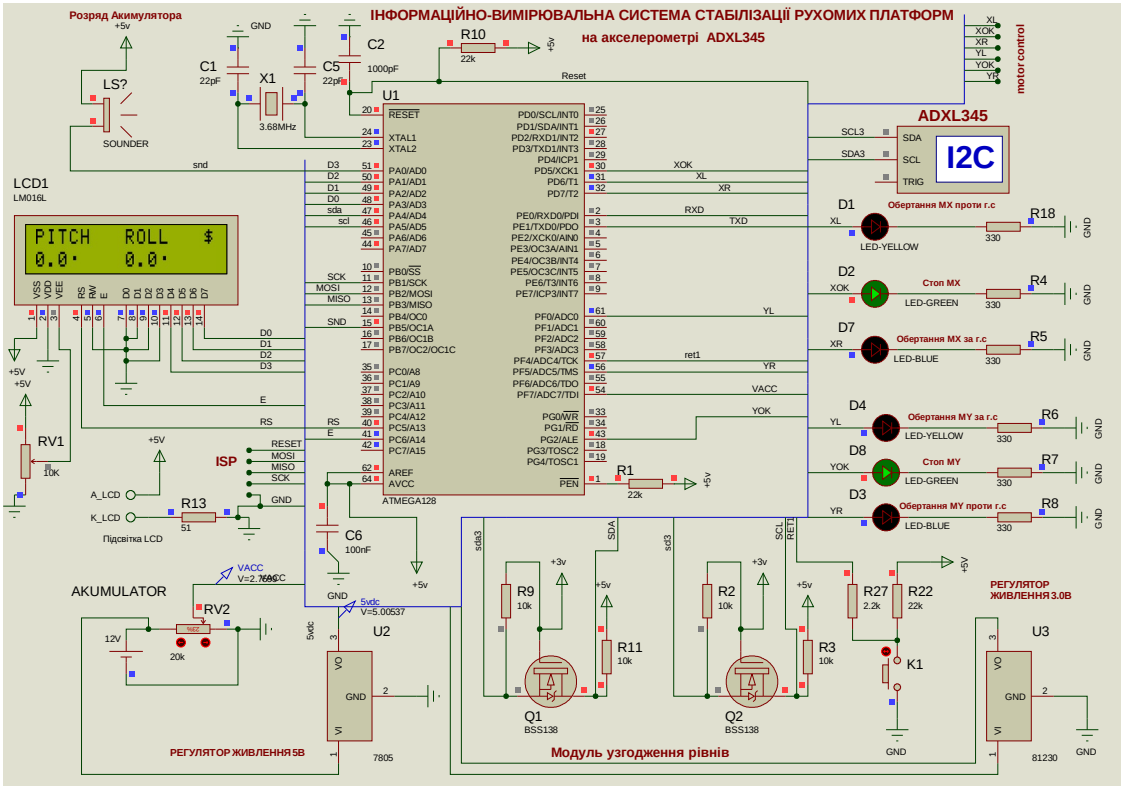


Рис. 4.12. Демонстрація переходу системи в режим відносних координат після активації клавіші K1

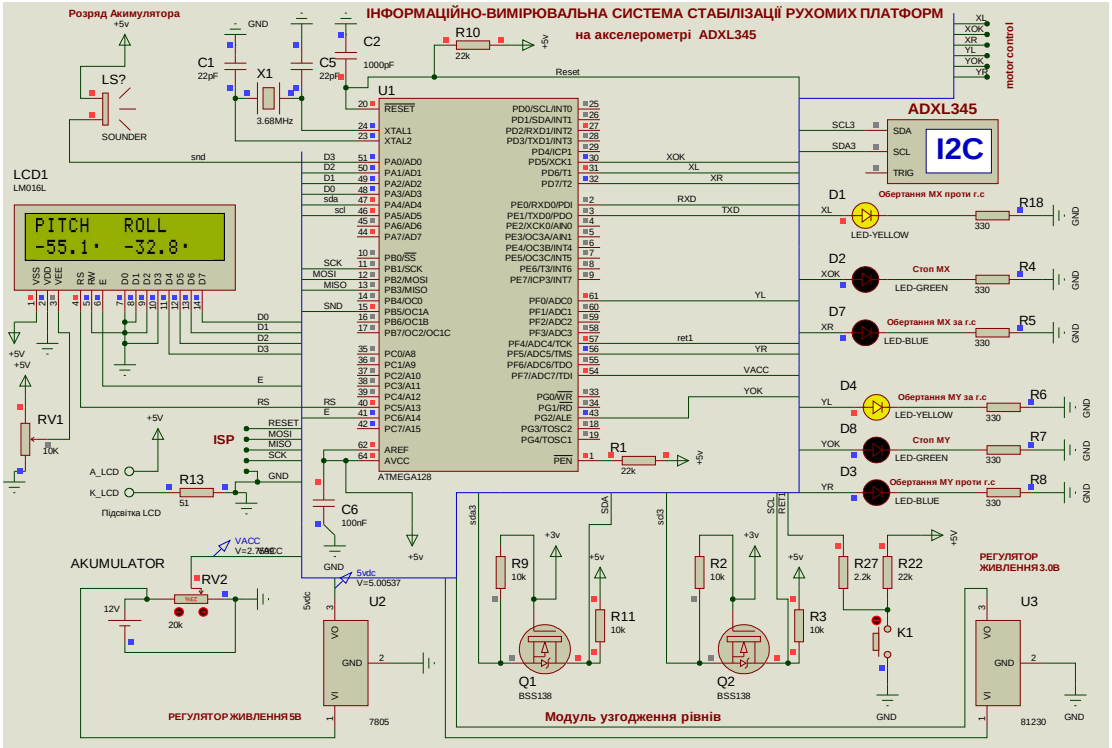


Рис. 4.13. Результати імітаційного моделювання системи стабілізації в середовищі Proteus ISIS під час функціонування в робочому режимі

Світлова індикація за допомогою жовтих діодів D1 та D4 підтверджує генерацію керуючих імпульсів для двигунів. Це відповідає режиму обертання стабілізаційної платформи в напрямку проти годинникової стрілки (Рис. 4.13).

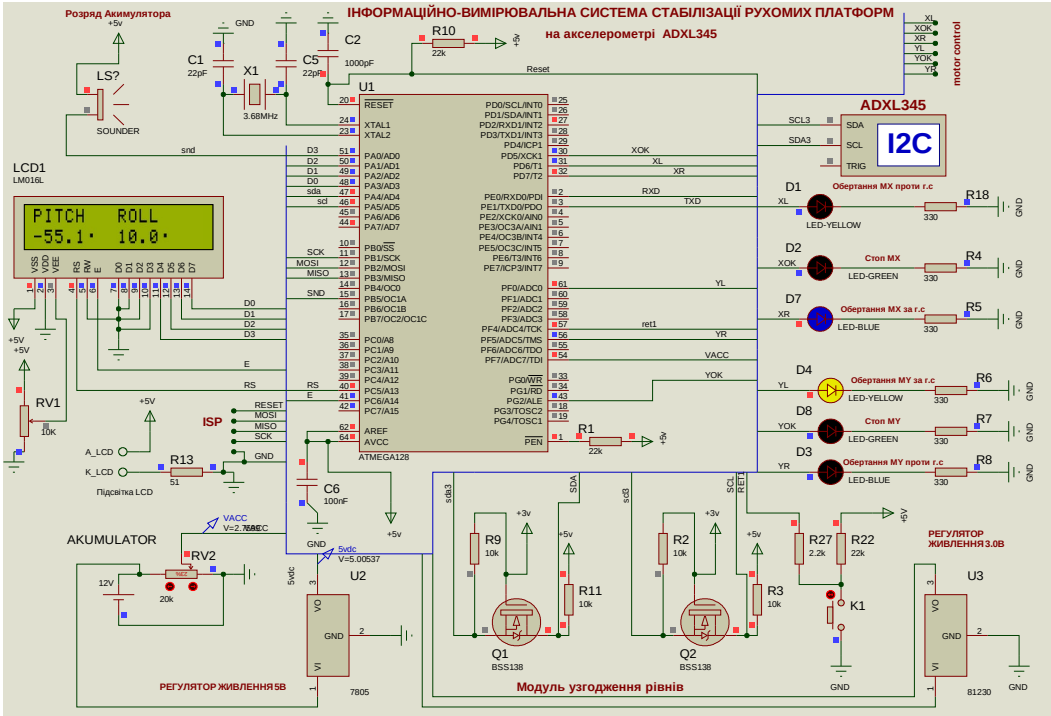


Рис. 4.14. Візуалізація функціонування розробленої системи стабілізації в інструментарії Proteus ISIS (робочий режим)

На Рис. 4.14 продемонстровано роботу приводів у відповідь на зміну просторової орієнтації: зафіксовано додатне значення кута ROLL та від'ємне – для PITCH. Процес падіння напруги джерела живлення відображено на рис. 4.15. Даний стан імітувався зміною опору потенціометра RV2. При досягненні критичного рівня розряду на рідкокристалічному екрані (у нижній стрічці) з'являється попереджувальний символ "*", а звуковий випромінювач LS активує тривожний сигнал за допомогою ШІМ.

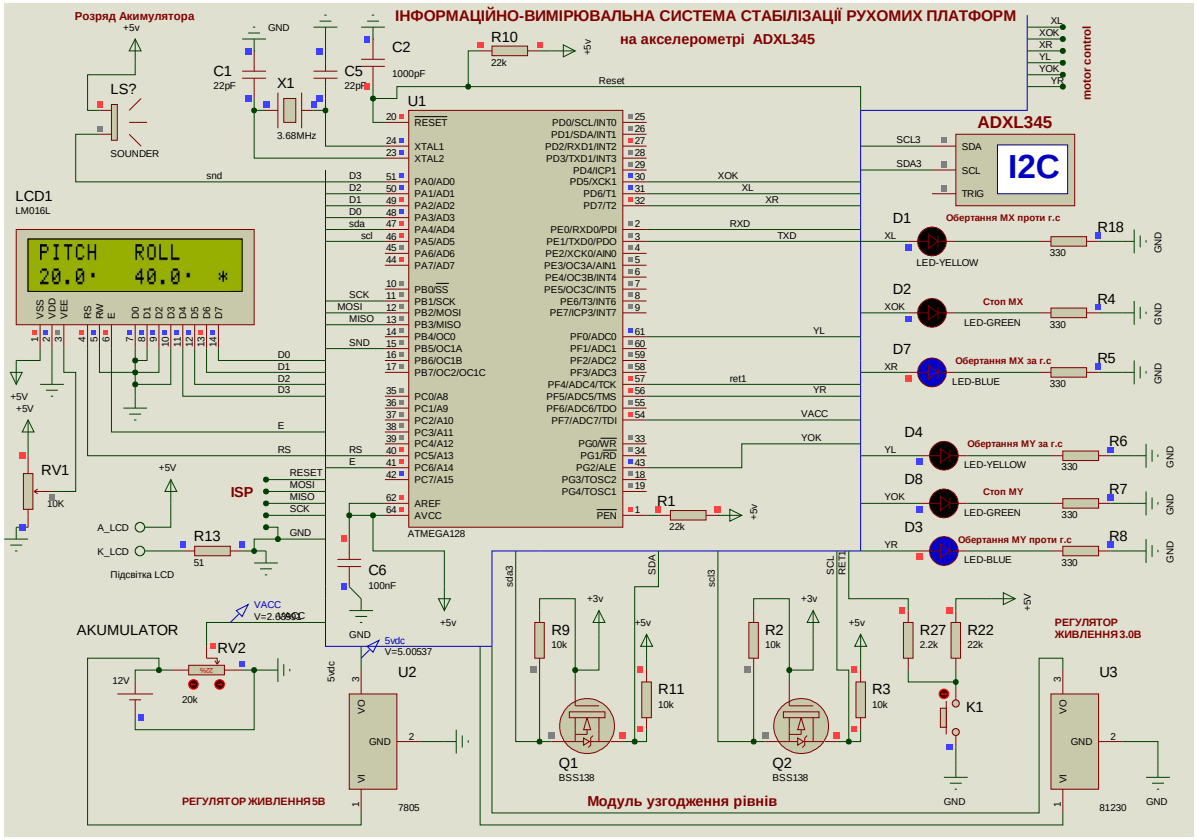


Рис. 4.15. Візуалізація результатів імітаційного моделювання системи при зниженні рівня напруги джерела живлення

ВИСНОВКИ

У межах кваліфікаційної роботи було спроєктовано та реалізовано програмно-апаратний комплекс, призначений для стабілізації рухомих платформ. Система інтегрована в конструкцію колісного та гусеничного транспорту для розмінування, забезпечуючи коректну роботу встановлених на них сенсорів та маніпуляційних механізмів.

Функціонування системи базується на зчитуванні показників триосьового акселерометра, жорстко закріпленого на платформі. Після аналізу отриманих даних пристрій генерує корегувальні сигнали для драйверів двигунів, що забезпечує стабілізацію положення. Поточні значення кутів відхилення відносно базової системи координат відображаються на рідкокристалічному дисплеї (РКД). Реалізована можливість оперативного перемикання між абсолютною та відносною системами координат за допомогою клавіш.

Система працює від акумуляторної батареї, забезпечуючи безперервний моніторинг її стану в режимі реального часу. У разі критичного зниження рівня заряду система активує звукове оповіщення та відображає відповідне попередження на РК-дисплеї.

Проєктування мікроконтролерної системи для стабілізації динамічних платформ здійснювалося з використанням актуальних електронних компонентів. Ключовими вузлами пристрою стали мікроконтролер архітектури AVR, прецизійний MEMS-акселерометр моделі ADXL345, а також рідкокристалічний модуль WH1602B-YGK-CTK формату 16x2.

Етап проектування включав розробку принципової електричної схеми та побудову цифрового прототипу системи стабілізації в середовищі Proteus VSM. Для керування пристроєм було створено відповідний алгоритм функціонування, реалізований мовою C. Програмний код для мікроконтролера AVR скомпільовано в інтегрованому середовищі розробки CodeVisionAVR.

У межах програмної реалізації було створено спеціалізовані модулі для взаємодії з цифровим акселерометром, а також підсистему візуалізації даних,

що забезпечує коректне відображення інформації на рідкокристалічному дисплеї (РКД).

Етап тестування мікроконтролерної системи стабілізації рухомих платформ було реалізовано шляхом моделювання в середовищі Proteus ISIS. Отримані результати підтвердили повну працездатність програмного забезпечення та коректність функціонування основного алгоритму стабілізації разом із допоміжними програмними модулями.

У процесі виконання роботи здобуто поглиблені теоретичні знання та практичний досвід у сфері проєктування й програмування цифрових систем на базі мікроконтролерів архітектури AVR. Особливу увагу приділено особливостям інтеграції та програмної обробки даних, отриманих із сучасних MEMS-акселерометрів.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Gyrostabilizer [Електронний ресурс] // The Free Dictionary. – URL: https://encyclopedia2.thefreedictionary.com/Gyrostabilizer.
2. How Gyrostabilizers Work [Електронний ресурс] : Whitepaper 1402. – URL: https://veemmarine.com/wp-content/uploads/2015/11/Whitepaper-1402-How_Gyrostabilizers_Work.pdf.
3. Arduino Self-Balancing Robot [Електронний ресурс] // Instructables. – URL: https://www.instructables.com/Arduino-Self-Balancing-Robot-1/.
4. Proteus VSM: Mixed Mode Circuit Simulation [Електронний ресурс] // Labcenter Electronics. – URL: https://www.labcenter.com/simulation/.
5. CodeVisionAVR C Compiler: User Manual [Електронний ресурс] / HP InfoTech S.R.L. – URL: https://www.thierry-lequeu.fr/data/CodeVision-AVR-3-20-User-Manual.pdf.
6. ATmega128: 8-bit AVR Microcontroller with 128K Bytes In-System Programmable Flash [Електронний ресурс] : Datasheet / Atmel Corporation. – 2011. – URL: https://ww1.microchip.com/downloads/en/DeviceDoc/-doc2467.pdf.
7. AVR Tutorials [Електронний ресурс] // Embedds. – URL: https://embedds-.com/avr-tutorials/.
8. AN2519: AVR Microcontroller Hardware Design Considerations [Electronic resource]: Application Note / Microchip Technology Inc. – 2017. – 27 p. – URL: https://ww1.microchip.com/downloads/en/Appnotes/AN2519-AVR-Microcontroller-Hardware-Design-Considerations-00002519B.pdf.
9. ADXL345: 3-Axis, ±2 g/±4 g/±8 g/±16 g Digital Accelerometer [Electronic resource] : Data Sheet / Analog Devices. – URL: https://www.analog.com/-media/en/technical-documentation/data-sheets/adxl345.pdf.
10. Fisher C. J. AN-1057: Using an Accelerometer for Inclination Sensing [Electronic resource] / Christopher J. Fisher // Analog Devices. – URL: https://www.analog.com/en/resources/app-notes/an-1057.html.

11. Begin J. D. Calibration of multi-axis accelerometer in vehicle navigation system : Patent US 6532419 B1 / Begin John David, Cheok Ka C. – 2003.
12. Travennor A. Experimenting with AVR Microcontrollers (Technology in Action) / Alan Travennor. – 1st ed. – Apress, 2014. – 206 p.
13. Kian S. T. Triaxial accelerometer static calibration / S. T. Kian, M. Awad, A. Dehghani, S. Zahedi // Proceedings of the World Congress on Engineering. – London, U.K., 2011. – Vol. III.
14. Artese G. Calibration of a low cost MEMS INS sensor for an integrated navigation system / G. Artese, A. Trecroci // The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. – Beijing, 2008. – Vol. XXXVII. Part B5. – P. 877–882.
15. Zang Hui. Research on Calibration method for the installation error of three-axis acceleration sensor / Zang Hui, Chai Wei, Zeng Huan-Tao, Lou Qiang // International Conference on Mechatronics and Control (ICMC). – 2014.
16. Larsdotter R. Automatic calibration and virtual alignment of MEMS-sensor placed in vehicle for use in road condition determination system [Electronic resource] : Master's Thesis / Rebecka Larsdotter, David Jaller. – Chalmers University of Technology, Sweden, 2014. – 72 p. – URL: https://odr.chalmers.se/items/8c609100-1ba3-4418-b127-cd5721cc3165.
17. Van Dam B. Microcontroller Systems Engineering: 45 projects for PIC, AVR and ARM / Bert van Dam. – Elektor Publishing, 2009. – 330 p.
18. Barrett S. F. Atmel AVR Microcontroller Primer: Programming and Interfacing / Steven F. Barrett, Daniel J. Pack. – 2nd ed. – Morgan & Claypool Publishers, 2012. – 194 p.
19. Williams E. Make: AVR Programming: Learning to Write Software for Hardware / Elliot Williams. – Make Community LLC, 2014. – 472 p.
20. Mazidi M. A. The AVR Microcontroller and Embedded Systems: Using Assembly and C / M. A. Mazidi, S. Naimi, S. Naimi. – 1st ed. – Pearson, 2015. – 752 p.

21. Barnett R. H. Embedded C Programming and the Atmel AVR / R. H. Barnett, S. A. Cox, L. O'Cull. – 2nd ed. – Cengage Learning, 2006. – 560 p.
22. Hung J. C. Calibration of accelerometer triad of an IMU with drifting Z-accelerometer bias / J. C. Hung, J. R. Thacher, H. V. White // Proceedings of the IEEE Aerospace and Electronics Conference NAECON-1989. – 1989. – Vol. 1. – P. 153–158.

ДОДАТКИ

Додаток 1

Головний програмний модуль системи стабілізації рухомої платформи з давачами та маніпуляторами колісного та гусеничного розміновуючого транспорту

```
/*****************************************************
Мікроконтролерна система стабілізації рухомих платформ з давачами та маніпуляторами колісного та гусеничного розміновуючого транспорту
Розробила: Христина Мартинів
*****************************************************/
#include <mega128.h>
#include <stdio.h>
#include <i2c.h>
#include <delay.h>
#include <math.h>
#include "mylcd.c"

// Налаштування шини I2C через асемблерні вставки
#asm
  .equ __i2c_port=0x1B ; /* PORTA */ .equ __sda_bit=4 .equ __scl_bit=5
#endasm

/* Адресація пристрою ADXL345 */
#define ADXL_I2C_READ    0xA7 #define ADXL_I2C_WRITE   0xA6

/* Регістри керування та ідентифікації */
#define REG_DEVID       0x00 // ID пристрою (0xE5)
#define REG_DATA_FORMAT  0x31 // Формат даних
#define REG_POWER_CTL    0x2D // Режим енергоспоживання
#define REG_BW_RATE      0x2C // Швидкість передачі (Data Rate)

/* Регістри налаштування чутливості та подій */
#define REG_TAP_THRESH   0x1D // Поріг для Single Tap
#define REG_OFFSET_X     0x1E // Калібрування осі X
#define REG_OFFSET_Y     0x1F // Калібрування осі Y
#define REG_OFFSET_Z     0x20 // Калібрування осі Z
#define REG_TAP_DUR      0x21 // Тривалість поштовху
#define REG_TAP_LATENT   0x22 // Затримка події
#define REG_TAP_WINDOW   0x23 // Вікно події
#define REG_ACT_THRESH   0x24 // Поріг активності
#define REG_INACT_THRESH 0x25 // Поріг бездіяльності
#define REG_INACT_TIME   0x26 // Час бездіяльності
#define REG_ACT_INACT_CTL 0x27 // Керування осями для Act/Inact
#define REG_FF_THRESH    0x28 // Поріг вільного падіння
#define REG_FF_TIME      0x29 // Час вільного падіння

/* Регістри переривань */
#define REG_INT_ENABLE 0x2E #define REG_INT_MAP 0x2F #define REG_INT_SOURCE 0x30

/* Регістри вихідних даних (X, Y, Z) */
#define REG_DATAX0       0x32
```

```
#define REG_DATAX1      0x33
#define REG_DATAY0      0x34
#define REG_DATAY1      0x35
#define REG_DATAZ0      0x36
#define REG_DATAZ1      0x37

/* Константи швидкості передачі даних (Output Data Rate) */
#define ODR_3200HZ      0x0F
#define ODR_1600HZ      0x0E
#define ODR_800HZ       0x0D
#define ODR_400HZ       0x0C
#define ODR_200HZ       0x0B
#define ODR_100HZ       0x0A
#define ODR_50HZ        0x09
#define ODR_25HZ        0x08
#define ODR_12_5HZ      0x07

/* Налаштування АЦП та периферії ATmega128 */
#define ADC_VREF_CFG    0x00 // ADLAR=0, 10-бітний режим
#define ADC_CH_PF7      0x07 // Вхідний канал PF7
#define VREF_SEL0       6
#define VREF_SEL1       7

#define PORT_BTNS    PORTF #define DDR_BTNS    DDRF  #define BTN_MENU_IDX    4

// Глобальні змінні
char lcd_out[33];           // Рядковий буфер для дисплея
unsigned char dev_id = 0xCC;
unsigned char coord_mode = 0; // Режим відносних координат
// Дані акселерометра
volatile int raw_x, raw_y, raw_z; volatile unsigned char sensor_buffer[6];
// Фізичні константи та коефіцієнти
const float MG_PER_LSB_OFFSET = 15.6; const float MG_PER_LSB_DATA = 3.9;
const float GRAVITY_CONST=9.81; float accel_x=0.0f, accel_y=0.0f, accel_z=0.0f;
float angle_pitch = 0.0, angle_roll = 0.0; float init_pitch = 0.0, init_roll = 0.0;
/* Перерахування для конфігурації акселерометра */
// Режими роздільної здатності
typedef enum {
   MODE_FULL_RES=8, MODE_2G=0, MODE_4G=1, MODE_8G=2, MODE_16G=3
} ADXL_Resolution;

// Типи переривань
typedef enum {
   IRQ_DATA_READY = 128, IRQ_SINGLE_TAP = 64, IRQ_DOUBLE_TAP = 32, RQ_ACTIVITY = 16,
   IRQ_INACTIVITY = 8, IRQ_FREE_FALL = 4, IRQ_WATERMARK = 2, IRQ_OVERRUN = 1
} ADXL_Interrupt;

// Стан функцій
typedef enum { FUNC_ON  = 1, FUNC_OFF = 2 } Feature_State;

// Призначення виводів переривань
typedef enum { MAP_TO_INT1 = 1, MAP_TO_INT2 = 2 } Interrupt_Mapping;
```

```
/* Зчитування одного байта з вказаного регістра акселерометра ADXL345 */
unsigned char get_adxl_reg_byte(unsigned char reg_addr) {
   unsigned char rx_byte; // Буфер для отриманого значення
   // Ініціалізація передачі та вибір регістра
   i2c_start();
   i2c_write(ADXL_I2C_WRITE);  // Надсилання адреси пристрою (режим запису)
   i2c_write(reg_addr);        // Вказуємо номер регістра
   // Повторний старт для перемикання в режим читання
   i2c_start();
   i2c_write(ADXL_I2C_READ);   // Надсилання адреси пристрою (режим читання)
   rx_byte = i2c_read(0);      // Читання байта без підтвердження (NACK)
   i2c_stop();       // Звільнення шини I2C
   delay_ms(20);    // Технологічна затримка для стабілізації
   return rx_byte; }

/* Функція для отримання 16-бітного значення (2 байти) з регістрів акселерометра */
int get_adxl_16bit_val(unsigned char start_reg) {
   unsigned char high_byte, low_byte;
   int result;
   // Ініціалізація зв'язку та вибір початкового регістра
   i2c_start();
   i2c_write(ADXL_I2C_WRITE);   // Адреса пристрою + біт запису
   i2c_write(start_reg);        // Встановлення покажчика на регістр
   // Перемикання в режим читання через повторний старт
   i2c_start();
   i2c_write(ADXL_I2C_READ);    // Адреса пристрою + біт читання
   // Читання даних:
   // Перший байт з ACK (підтвердження), другий з NACK (завершення)
   high_byte = i2c_read(1);
   low_byte  = i2c_read(0);
   i2c_stop();    // Завершення сесії I2C
   delay_ms(20);   // Пауза для стабілізації інтерфейсу
   // Об'єднання байтів у ціле число (MSB зсувається вліво на 8 біт)
   result = ((int)high_byte << 8) | low_byte;
   return result; }

/* Функція для відправки байта даних у конкретний регістр ADXL345 */
void set_adxl_reg_byte(unsigned char reg_addr, unsigned char val) {
   // Ініціалізація сеансу зв'язку по шині I2C
   i2c_start();
   // Передача адреси пристрою з бітом запису (Write)
   i2c_write(ADXL_I2C_WRITE);
   // Вказуємо цільовий регістр
   i2c_write(reg_addr);
   // Передаємо значення, яке потрібно записати
   i2c_write(val);
   // Завершення транзакції
   i2c_stop();
   // Необхідна пауза для завершення внутрішнього циклу запису датчика
   delay_ms(20); }


/* Налаштування робочого режиму акселерометра ADXL345 */
```

```
void setup_adxl_sensor(void) {
   // Крок 1: Переведення сенсора в режим очікування (Standby)
   // Це скидає внутрішні прапорці та готує пристрій до конфігурації
   set_adxl_reg_byte(REG_POWER_CTL, 0x00);
   // Технологічна пауза для перемикання режимів живлення
   delay_ms(10);
   // Крок 2: Активація режиму вимірювання (Measurement Mode)
   // Встановлюємо 3-й біт (Measure) у регістрі POWER_CTL
   set_adxl_reg_byte(REG_POWER_CTL, (1 << 3));
   // Очікування стабілізації вихідних даних
   delay_ms(15); }


/*Керування режимом вимірювання (Measurement Mode)
 * @param state: FUNC_ON — активувати вимірювання, FUNC_OFF — перейти в Standby */
void toggle_measure_mode(Feature_State state) {
   unsigned char current_reg_val;
   // 1. Отримуємо поточний стан регістра POWER_CTL
   current_reg_val = get_adxl_reg_byte(REG_POWER_CTL);
   // 2. Модифікуємо лише 3-й біт (Measure), зберігаючи інші налаштування
   if (state == FUNC_ON)
   { current_reg_val |= (1 << 3);  // Встановлюємо біт (Measure = 1) }
   else
   { current_reg_val &= ~(1 << 3); /* Скидаємо біт (Measure = 0) */ }
   // 3. Записуємо оновлене значення назад у сенсор
   set_adxl_reg_byte(REG_POWER_CTL, current_reg_val); }


/* Налаштування системи переривань акселерометра
 * @param event: тип події (DataReady, Tap, Activity тощо)
 * @param state: статус (FUNC_ON / FUNC_OFF)
 * @param output_pin: вибір фізичної ніжки (MAP_TO_INT1 / MAP_TO_INT2) */
void configure_interrupts(ADXL_Interrupt event, Feature_State state, Interrupt_Mapping
output_pin) {
   unsigned char enable_reg, map_reg;
   // --- 1. Керування активацією переривання (REG_INT_ENABLE) ---
   enable_reg = get_adxl_reg_byte(REG_INT_ENABLE);
   if (state == FUNC_ON) enable_reg |= event;  // Вмикаємо біт переривання
   else enable_reg &= ~event; // Вимикаємо біт переривання
   set_adxl_reg_byte(REG_INT_ENABLE, enable_reg);
   // --- 2. Маршрутизація на фізичний вивід (REG_INT_MAP) ---
   // У регістрі MAP: 0 = INT1, 1 = INT2
   map_reg = get_adxl_reg_byte(REG_INT_MAP);
   if (output_pin == MAP_TO_INT1)
      map_reg &= ~event; // Скидаємо біт для спрямування на INT1
   else map_reg |= event;  // Встановлюємо біт для спрямування на INT2
   set_adxl_reg_byte(REG_INT_MAP, map_reg); }


/* Перевірка статусу конкретного переривання
 * @param event: подія для перевірки (наприклад, FREE_FALL, SINGLE_TAP)
 * @return: 1 — переривання активне, 0 — неактивне */
unsigned char check_interrupt_event(ADXL_Interrupt event) {
   unsigned char status_reg;
```

```
    // Зчитуємо вміст регістра джерел переривань
    status_reg = get_adxl_reg_byte(REG_INT_SOURCE);
    // Перевіряємо, чи встановлено відповідний біт події
    // Оскільки елементи enum ADXL_Interrupt вже є масками (1, 2, 4, 8...),
    // використовуємо побітове "І" (&) без додаткових зсувів.
    if (status_reg & event)
    { return 1; // Подія відбулася
    }
    return 0; /* Подія відсутня */ }

/* Налаштування чутливості детектора поштовхів (Tap Detection)
 * @param threshold_val: значення порогу (крок 62.5 mg/LSB)
 * Приклад: значення 16 відповідає приблизно 1g (16 * 62.5 = 1000 mg) */
void update_tap_sensitivity(unsigned char threshold_val) {
    // Використовуємо уніфіковану функцію запису в регістр
    // REG_TAP_THRESH ми визначили раніше як 0x1D
    set_adxl_reg_byte(REG_TAP_THRESH, threshold_val); }


/* Налаштування максимальної тривалості поштовху (DUR)
 * @param time_limit: значення ліміту часу (крок 625 мкс/LSB)
 * Приклад: значення 16 відповідає 10 мс (16 * 0.625 = 10 мс) */
void set_tap_time_limit(unsigned char time_limit) {
    // Використовуємо уніфіковану функцію запису для доступу до регістра DUR.
    // REG_TAP_DUR ми визначили раніше як 0x21.
    set_adxl_reg_byte(REG_TAP_DUR, time_limit); }


/* Налаштування часового інтервалу затримки (Latent)
 * @param delay_val: значення затримки (крок 1.25 мс/LSB)
 * Це мінімальний час очікування після першого поштовху,
 * перш ніж почнеться пошук другого (Double Tap). */
void set_tap_latency_window(unsigned char delay_val) {
    // Використовуємо уніфіковану функцію запису для доступу до регістра LATENT.
    // REG_TAP_LATENT ми визначили раніше як 0x22.
    set_adxl_reg_byte(REG_TAP_LATENT, delay_val); }


/* Налаштування часового вікна для реєстрації другого поштовху
 * @param window_val: значення інтервалу (крок 1.25 мс/LSB)
 * Це період часу після закінчення Latency, протягом якого
 * має відбутися другий поштовх для активації Double Tap. */
void set_double_tap_window(unsigned char window_val) {
    // Використовуємо уніфіковану функцію запису для доступу до регістра WINDOW.
    // REG_TAP_WINDOW ми раніше визначили як 0x23.
    set_adxl_reg_byte(REG_TAP_WINDOW, window_val); }


/* Налаштування порогу детекції активності (Activity Threshold)
 * @param act_val: значення порогу (крок 62.5 mg/LSB)
 * Приклад: при act_val = 8, поріг складе 0.5g (8 * 62.5 = 500 mg)
 * Дана функція визначає рівень прискорення, що вважається початком руху. */
void set_activity_trigger_level(unsigned char act_val) {
    // Використовуємо уніфіковану функцію запису для доступу до регістра THRESH_ACT.
    // REG_ACT_THRESH ми визначили раніше як 0x24.
```

```
   set_adxl_reg_byte(REG_ACT_THRESH, act_val); }


/* Встановлення рівня чутливості для визначення стану спокою (Inactivity)
 * @param inact_val: рівень порогу (дискретність 62.5 mg/LSB)
 * Якщо прискорення по обраних осях нижче цього значення протягом
 * часу TIME_INACT, генерується переривання неактивності. */
void set_inactivity_limit(unsigned char inact_val) {
   // Використовуємо уніфіковану функцію запису для доступу до регістра THRESH_INACT.
   // REG_INACT_THRESH ми визначили раніше як 0x25.
   set_adxl_reg_byte(REG_INACT_THRESH, inact_val); }


/* Налаштування часового інтервалу для детекції спокою
 * @param seconds_val: тривалість у секундах (1 сек/LSB)
 * Визначає час, протягом якого прискорення має залишатися
 * нижче порогу THRESH_INACT для активації переривання. */
void set_inactivity_timer(unsigned char seconds_val) {
   // Використовуємо нашу базову функцію запису для доступу до регістра TIME_INACT.
   // REG_INACT_TIME ми визначили раніше як 0x26.
   set_adxl_reg_byte(REG_INACT_TIME, seconds_val); }


/* Зчитування даних прискорення та обчислення середнього значення
 * @param samples: кількість вимірювань для фільтрації (усереднення) */
void update_accel_averages(unsigned char samples)
{
   unsigned char i, j;
   long sum_x = 0, sum_y = 0, sum_z = 0;
   for (i = 0; i < samples; i++)
   {
      // 1. Початок транзакції I2C для пакетного читання 6 байт (X, Y, Z)
      i2c_start();
      i2c_write(ADXL_I2C_WRITE);
      i2c_write(REG_DATAX0); // Починаємо з першого регістру даних
      i2c_start();
      i2c_write(ADXL_I2C_READ);
      // 2. Послідовне читання 6 регістрів (DATAX0...DATAZ1)
      for (j = 0; j < 6; j++)
      {
         // Для перших 5 байт надсилаємо ACK (1), для останнього — NACK (0)
         sensor_buffer[j] = i2c_read(j < 5);
      }
      i2c_stop();
      delay_ms(15); // Пауза між вибірками
      // 3. Збірка 16-бітних значень та додавання до суми
      sum_x += (int)((sensor_buffer[1] << 8) | sensor_buffer[0]);
      sum_y += (int)((sensor_buffer[3] << 8) | sensor_buffer[2]);
      sum_z += (int)((sensor_buffer[5] << 8) | sensor_buffer[4]);
   }
   // 4. Обчислення фінального середнього арифметичного
   raw_x = (int)(sum_x / samples);
   raw_y = (int)(sum_y / samples);
   raw_z = (int)(sum_z / samples); }
```

```
/* Генерація короткого звукового сигналу (біп)
 * Використовує 5-й пін порту B для керування випромінювачем */
void trigger_audio_alert(void) {
   // Налаштовуємо PB5 як вихід (логічна одиниця вмикає звук)
   DDRB |= (1 << 5);
   // Тривалість активного стану сигналу
   delay_ms(200);
   // Перемикаємо PB5 у стан входу (вимкнення звуку через Hi-Z або нуль)
   DDRB &= ~(1 << 5);
   // Пауза після сигналу для розділення звуків
   delay_ms(200);  }


/*Зчитування результату перетворення АЦП для вказаного каналу
 * @param channel: номер вхідного каналу (0-7)
 * @return: 10-бітне цифрове значення напруги */
unsigned int get_adc_sample(unsigned char channel) {
   // 1. Налаштування мультіплексора: вибір каналу та опорної напруги (Vref)
   // Використовуємо маску, щоб не зачепити інші налаштування ADMUX
   ADMUX = (channel & 0x07) | (ADC_VREF_TYPE & 0xC0);
   // Коротке очікування для стабілізації вхідної напруги на конденсаторі S&H
   delay_us(15);
   // 2. Запуск циклу перетворення (встановлення біта ADSC - ADC Start Conversion)
   ADCSRA |= (1 << ADSC);
   // 3. Очікування завершення (прапорець ADIF - ADC Interrupt Flag встановлюється в 1)
   // Цикл працює, поки біт ADIF рівний 0
   while (!(ADCSRA & (1 << ADIF)));
   // 4. Скидання прапорця ADIF шляхом запису логічної одиниці в нього
   ADCSRA |= (1 << ADIF);
   // Повертаємо повний 10-бітний результат з реєстру ADCW (ADCL + ADCH)
   return ADCW;
}

/* Глобальні змінні для моніторингу та інтерфейсу */
// Параметри для виведення даних на дисплей
float display_val;              // Значення для відображення на LCD (колишня indication)
const float MATH_PI = 3.14159f;    // Математична константа Pi (колишня tinf)
// Параметри системи живлення та АЦП
unsigned int raw_adc_code;        // Отриманий цифровий код з АЦП
const float VOLT_THRESHOLD_MIN = 11.0f; // Критичний поріг напруги акумулятора
// Змінні для керування меню та кнопками
unsigned char menu_index;         // Індекс поточного пункту меню (колишня k)
unsigned char last_button_state = 0xFF; // Попередній стан пінів порту (для антибрязкоту)
/**
 * Перевірка стану кнопки на порту F
 * @param bit_mask: номер піна (0-7), до якого підключена кнопка
 * @return: повертає true (1), якщо кнопку натиснуто (логічний нуль на вході)
 */
unsigned char is_button_pressed(unsigned char bit_mask) {
   // Оскільки використовується внутрішня підтяжка (Pull-up),
   // натиснута кнопка замикає пін на землю, даючи низький рівень (0).
   // Оператор '!' інвертує результат для повернення логічної одиниці.
```

```
    if (!(PINF & (1 << bit_mask)))
    { return 1; /* Кнопка активна */   }
    return 0; /* Кнопка не натиснута */ }

void main(void)
{
    // --- 1. Низькорівневе налаштування портів (IO Setup) ---
    // Всі порти в початковий стан (входи, підтяжка вимкнена)
    PORTA = 0x00; DDRA = 0x00; PORTB = 0x00; DDRB = 0x00;
    PORTC = 0x00; DDRC = 0x00; PORTD = 0x00; DDRD = 0x00;
    PORTE = 0x00; DDRE = 0x00; PORTF = 0x00; DDRF = 0x00;
    PORTG = 0x00; DDRG = 0x00;
    // --- 2. Конфігурація периферії (Таймери та АЦП) ---
    // Таймер 1: Генерація сигналу (PWM режим 14 - ICR1 як TOP)
  TCCR1A=(1<<COM1A1)|(1<<WGM11); TCCR1B=(1<<WGM13)|(1<<WGM12)|(1<<CS10);
    ICR1 = 0x07CF;   // Частота сигналу
    OCR1A = 0x03FF;  // Шпаруватість (Duty Cycle)
    // Налаштування АЦП: AREF=5V, Prescaler=32 (125kHz при 4MHz)
   ADMUX=(ADC_VREF_TYPE&0xFF); ADCSRA=(1<<ADEN)|(1<<ADPS2)|(1<<ADPS0);
    // --- 3. Ініціалізація інтерфейсу та датчиків ---
    lcd_init(); lcd_clear(); i2c_init();
    // Налаштування кнопок (Порт F)
    DDRF &= ~(1 << MENU_ENTER_BTN);
    PORTF |= (1 << MENU_ENTER_BTN); // Увімкнення Pull-up
    setup_adxl_sensor(); // Наша нова функція ініціалізації
    // Конфігурація параметрів акселерометра
    set_adxl_reg_byte(REG_DATA_FORMAT, 0x0B); // +/-16g, Full Res
    update_tap_sensitivity(50);
    set_tap_time_limit(15); set_activity_trigger_level(75);
    set_inactivity_limit(75); set_inactivity_timer(10);
    // Перевірка зв'язку з датчиком
    unsigned char sensor_id = get_adxl_reg_byte(REG_DEVID);
    trigger_audio_alert(); // Звуковий сигнал при старті
    if (sensor_id == 0xE5) {
       sprintf(buffer, "ID: 0x%02X OK", sensor_id); lcd_puts(buffer);
    } else { lcd_putsf("ADXL ERROR!"); }
    delay_ms(1000);
    lcd_clear(); lcd_putsf("  STABILIZING  "); lcd_gotoxy(0, 1);
    lcd_putsf("    SYSTEM     "); delay_ms(1500);
    // --- 4. Головний цикл керування (Control Loop) ---
    while (1) {
       // Отримання та обробка даних
       update_accel_averages(3);
       // Розрахунок фізичних величин (прискорення та кути)
       float x_g = (raw_x * gainX) * (3.9 / 10.0);
       float y_g = (raw_y * gainY) * (3.9 / 10.0);
       float z_g = (raw_z * gainZ) * (3.9 / 10.0);
       float rad_to_deg = 180.0 / MATH_PI;
  float current_pitch = atan2(x_g, sqrt(y_g * y_g + z_g * z_g))*rad_to_deg - dpitch;
  float current_roll  = atan2(y_g, sqrt(x_g * x_g + z_g * z_g)) * rad_to_deg - droll;
       // Візуалізація на LCD
       lcd_gotoxy(0, 0); lcd_putsf("PITCH   ROLL");
     sprintf(buffer, "%+5.1f%c  %+5.1f%c", current_pitch, 0xDF, current_roll, 0xDF);
```

```
  lcd_gotoxy(0, 1); lcd_puts(buffer);
  // --- 5. Логіка стабілізації (Двигуни) ---
  // Керування по осі Roll (Крен) - Порт D
  if (current_roll == 0) {
     DDRD |= (1 << 5); PORTD |= (1 << 5); // Стан "Норма"
     PORTD &= ~((1 << 6) | (1 << 7));
  } else { PORTD &= ~(1 << 5);
     if (current_roll < 0) {
        DDRD |= (1 << 6); PORTD |= (1 << 6); // Вліво
        PORTD &= ~(1 << 7);
     } else { DDRD |= (1 << 7); PORTD |= (1 << 7); // Вправо
        PORTD &= ~(1 << 6); } }
  // Керування по осі Pitch (Тангаж) - Порти G та F
if (current_pitch == 0) { DDRG |= (1 << 2); PORTG |= (1 << 2); // Стан "Норма"
     PORTF &= ~((1 << 0) | (1 << 5));
  } else { PORTG &= ~(1 << 2);
     if (current_pitch < 0) { DDRF |= (1 << 0); PORTF |= (1 << 0); // Вгору
        PORTF &= ~(1 << 5);
     } else { DDRF |= (1 << 5); PORTF |= (1 << 5); // Вниз
        PORTF &= ~(1 << 0); } }
  // --- 6. Системний моніторинг (Батарея та Кнопки) ---
  raw_adc_code = get_adc_sample(ADC_INPUT);
  float battery_volt = (raw_adc_code / 0.512) / 100.0;
  if (battery_volt <= VOLT_THRESHOLD_MIN) {
     lcd_gotoxy(15, 1); lcd_putsf("!"); // Попередження про розряд
     trigger_audio_alert(); }
  // Обробка кнопки калібрування (Zero-level)
  if (is_button_pressed(MENU_ENTER_BTN)) {
     if (flagk == 0) {
        dpitch = current_pitch; droll = current_roll; flagk = 1;
        lcd_gotoxy(15, 0); lcd_putsf("*");
     } else {
        dpitch = 0; droll = 0; flagk = 0;
        lcd_gotoxy(15, 0); lcd_putsf(" "); }
     delay_ms(800); // Захист від брязкоту
  } } }
```