

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Радіотехнічний факультет
Катедра прикладної радіоелектроніки

До захисту допущено:

Зав. катедри

_____ Андрій МОВЧАНЮК

«__» _____ 20__ р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньою програмою «Інтелектуальні технології радіоелектронної техніки», за спеціальністю 172 «Телекомунікації та радіотехніка»

на тему: АВТОМАТИЗАЦІЯ 3D ДРУКУ

Виконала:

студентка IV курсу, групи РЕ-22

Кравецька Валерія Марківна

Прізвище, ім'я, по батькові

Керівник: ст. викладач кат. РЕ Адаменко Володимир Олексійович

Посада, науковий ступінь, вчене звання

Прізвище, ім'я, по батькові

Рецензент: ст. викладач кат. РІ Захарченко О. С.

Посада, науковий ступінь, вчене звання,

Прізвище, ім'я, по батькові

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.

Студентка _____

Київ – 2026 року

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

Радіотехнічний факультет

Кафедра прикладної радіоелектроніки

Рівень вищої освіти – перший (бакалаврський)

Спеціальність 172 «Телекомунікації та радіотехніка»

Освітня програма «Інтелектуальні технології радіоелектронної техніки»

ЗАТВЕРДЖУЮ

Зав. кафедри

_____ Андрій МОВЧАНЮК

«___» _____ 20__ р.

ЗАВДАННЯ

на дипломний проєкт студентки

Кравецької Валерії Марківни

1. Тема проєкту «Автоматизація 3D друку»

2. Керівник проєкту ст.вик. катедри РЕ Адаменко Володимир Олексійович, затверджені наказом по університету від « 29 » травня 2026 р. №1989-с

Строк подання студентом проєкту 12 червня 2026 року

3. Вихідні дані до проєкту: 3D-принтер Bambu Lab A1, зарядна станція EcoFlow (з підтримкою Cloud API), електроживлення керуючого шлюзу: постійна напруга 5 В.

4. Зміст пояснювальної записки 1)Огляд існуючих рішень, 2) Розробка архітектури та апаратної складової системи 3) Розробка програмного забезпечення та алгоритмів автоматизації

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо): Схема структурна апаратно-програмного комплексу (А4), схема функціональна взаємодії модулів та мережевих інтерфейсів (А4), схема алгоритму функціонування програмного забезпечення обчислювального модуля (А4), демонстраційна презентація дипломного проєкту (12 слайдів), креслення (12 штук, А4/А3)

6. Дата видачі завдання «4» вересня 2025 р.

Календарний план

№з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	Аналіз існуючих рішень	02.03.2026 – 15.03.2026	
2.	Розробка механічної частини	16.03.2026 – 20.05.2026	
3.	Розробка структури системи та вибір компонентів	01.04.2026 – 15.04.2026	
4.	Розробка програмного забезпечення проєкту	16.04.2026 – 24.05.2026	
5.	Створення інтерфейсів для користувача	25.05.2026 – 31.05.2026	
6.	Підготовка матеріалів графічної частини	01.06.2026 – 05.06.2026	
7.	Оформлення текстової та графічної документації	06.06.2026 – 11.06.2026	
8.	Попередній огляд матеріалів на кафедрі	12 червня	

Студентка

(підпис)

Керівник проєкту

(підпис)

В.М. КРАВЕЦЬКА

(ініціали, прізвище)

В.О. АДАМЕНКО

(ініціали, прізвище)

АНОТАЦІЯ

Метою роботи є розробка та реалізація програмно-апаратного комплексу для автоматизації роботи 3D-ферми на базі принтера Bambu Lab A1 з використанням мікроконтролера ESP32 як центрального інтелектуального шлюзу. Система призначена для забезпечення повного циклу автономної роботи обладнання в режимі «24/7» шляхом автоматичного очищення робочого столу, динамічного формування черги друку з різних файлів та віддаленого моніторингу телеметрії. У роботі застосовано методи об'єктно-орієнтованого програмування на мові C++ для розробки внутрішнього програмного забезпечення мікроконтролера. Комунікація між компонентами побудована на базі мережевого протоколу MQTT (обмін телеметрією з принтером і Telegram-ботом у реальному часі) та протоколу HTTP (для взаємодії з хмарним EcoFlow API та локальним веб-інтерфейсом). Проектування корпусу модуля та важеля очищення столу виконано методами тривимірного комп'ютерного моделювання (CAD) з урахуванням специфіки наступного виготовлення деталей методом FDM-друку. Дипломний проект містить 59 сторінок пояснювальної записки, 31 рисунок, 4 таблиці, 7 додатків.

Ключові слова: 3D-ферма, Bambu Lab A1, ESP32, мікроконтролер, автоматизація, протокол MQTT, телеграм-бот, черга друку, очищення столу, CAD-моделювання, FDM-друк, електронний пристрій керування, Ecoflow API, IOT-шлюз.

ANNOTATION

The objective of the work is to develop and implement a hardware and software complex for automating a 3D printing farm based on the Bambu Lab A1 printer using an ESP32 microcontroller as a central intelligent gateway. The system is designed to provide a full cycle of autonomous equipment operation in "24/7" mode through automatic print bed cleaning, dynamic print queue management from various files, and remote telemetry monitoring. The work applies object-oriented programming methods in C++ for firmware development. Communication between components is built on the MQTT network protocol (real-time telemetry exchange with the printer and Telegram bot) and the HTTP protocol (for interaction with the cloud EcoFlow API and local web interface). The design of the module case and the bed cleaning lever was performed using three-dimensional computer-aided design (CAD) methods, considering the specifics of subsequent part manufacturing by FDM printing. The diploma project contains 59 pages of the explanatory note, 31 figures, 4 tables, 7 appendices.

Keywords: 3D printing farm, Bambu Lab A1, ESP32, microcontroller, automation, MQTT protocol, Telegram bot, print queue, bed cleaning, CAD modeling, FDM printing, electronic control device, EcoFlow API, IoT gateway.

Пояснювальна записка
до дипломного проєкту
на тему: «Автоматизація 3D-друку»

Київ — 2026 року

ЗМІСТ

Перелік умовних позначень та скорочень	4
Вступ.....	8
1 Огляд існуючих рішень	9
1.1 Програмні комплекси екосистеми Bambu Lab та хмарні IoT-платформи моніторингу	9
1.2 Аматорські та комерційні системи автоматичної зміни платформ	12
1.3 Порівняльний аналіз та обґрунтування розробки.....	15
2 Розробка механічної частини.....	16
2.1 Особливості проектування та адаптації для принтерів типу "bedslinger"	16
2.1.1 Кінематика принтерів з рухомим столом та обмеження FDM-друку	16
2.1.2 Обґрунтування та порівняльний аналіз матеріалів для виготовлення вузлів APBC.....	19
2.1.3 Концептуальне моделювання та конструктивні елементи APBC.....	21
3 Розробка електронної частини.....	24
3.1 Загальна архітектура системи.....	24
3.2 Обґрунтування вибору мікроконтролера та середовища розробки	25

					PE22.468332.001			
ЗМ.	Лис	№ докум.	Підпис	Дата	Автоматизація 3D друку	Лім.	Лист	Листів
Розробила	Кравецька В.М.						1	
Перевірів	Адаменко В.О.							
Н. Контр.								
Затвер-	Адаменко В.О.					PE-22, РТФ		

3.3 Специфіка системи Bambu Lab та налагодження локального зв'язку	29
3.4 Комунікаційний протокол MQTT та підключення принтерів	30
3.4.1 Особливості протоколу Bambu Lab MQTT	30
3.4.2 Управління з'єднанням та механізм pushall	31
3.5 Система черги друку	33
3.6 Модифікація керуючого G-коду 3D-принтера для інтеграції системи автоматичної зміни платформ	34
3.6.1 Зміни в параметрах робочого простору	34
3.6.2 Структура інтегрованого сервісного коду очищення платформи	36
3.7 Контроль енергопостачання через інтеграцію з EcoFlow	40
3.7.1 REST API та підпис запитів HMAC-SHA256	41
3.7.2 Обробка телеметрії та автоматичне реагування	42
3.8 Користувачькі інтерфейси	42
3.8.1 Локальна веб-панель	42
3.8.2 Режим початкового налаштування (AP Mode)	44
3.8.3 Telegram-бот	45
3.9 Збереження конфігурації	48
3.10 Конструктивне виконання захисного корпусу та ергономіка електронного модуля	48
Висновки	56

Перелік джерел посилання.....	58
Додаток А. Технічне Завдання	1
Додаток Б. Схема структурна комплексу автоматизації.....	1
Додаток В. Схема функціональна взаємодії задач FreeRTOS	1
Додаток Г. Схема алгоритму функціонування ПЗ.....	1
Додаток І. Конструкторська документація підсисте	2
Додаток Д. Робочі креслення деталей (Корпус)	11
Додаток Е. Лістинг Коду	1

					РЕ22.468332.001 ПЗ	Лис
						3
Зм.	Лис	№ докум.	Підпис	Дата		

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ЄСКД – Єдина система конструкторської документації

ПЗ – програмне забезпечення

ТЗ – технічне завдання

3MF (3D Manufacturing Format) – універсальний XML-формат 3D-даних та тривимірних моделей для адитивного виробництва

AP Mode (Access Point Mode) – режим роботи мікроконтролера як бездротової точки доступу Wi-Fi

APBC (Automated Build Plate Changer) – механічна система автоматичної зміни робочих платформ (друкарських пластин)

API (Application Programming Interface) – програмний інтерфейс взаємодії між додатками та сервісами

ASA (Acrylonitrile Styrene Acrylate) – атмосферостійкий термопластичний полімер для FDM 3D-друку

ABS (Acrylonitrile Butadiene Styrene) – ударостійкий технічний термопласт, що використовується для друку функціональних вузлів

Captive Portal – програмний сервіс веб-сторінки автентифікації при першому підключенні до точки доступу

CSV (Comma-Separated Values) – текстовий формат, призначений для представлення табличних даних у вигляді списку значень, розділених комою

DC (Direct Current) – постійний струм

DNS (Domain Name System) – система доменних імен (мережева служба для трансляції доменів в IP-адреси)

EEPROM (Electrically Erasable Programmable Read-Only Memory) – електрично прана програмована пам'ять лише для зчитування

					<i>PE22.468332.001 ПЗ</i>	Лис
						4
Зм.	Лис	№ докум.	Підпис	Дата		

ESP-IDF (Espressif IoT Development Framework) – офіційний фреймворк та операційне середовище розробника для мікроконтролерів серії ESP32

FDM (Fused Deposition Modeling) – технологія адитивного виробництва методом пошарового наплавлення пластикової нитки (філаменту)

FreeRTOS (Real-Time Operating System) – операційна система реального часу для вбудованих систем та мікроконтролерів

FTP / FTPS (File Transfer Protocol / Secure) – захищений протокол передачі файлів комп'ютерними мережами

G-code – універсальна стандартизована мова програмування пристроїв із ЧПК (числовим програмним керуванням)

GET / POST – базові методи HTTP-запитів для отримання та відправки даних між клієнтом і сервером

HMAC-SHA256 (Keyed-Hash Message Authentication Code) – криптографічний алгоритм обчислення коду автентифікації повідомлень на основі хеш-функції SHA-256 та секретного ключа

HTTP / HTTPS (Hypertext Transfer Protocol / Secure) – захищений протокол передачі гіпертексту мережами Інтернет

IoT (Internet of Things) – концепція мережі фізичних об'єктів («інтернет речей»), оснащених вбудованими технологіями для взаємодії

IP / IP-адреса (Internet Protocol) – унікальний числовий ідентифікатор пристрою в комп'ютерній мережі

JSON (JavaScript Object Notation) – текстовий формат обміну даними, що базується на принципі «ключ-значення»

LAN Only Mode – режим роботи обладнання Bambu Lab виключно всередині локальної мережі, без доступу до зовнішніх хмарних серверів

Long Polling – технологія постійного утримання мережевого HTTP-з'єднання клієнтом для отримання миттєвих оновлень від сервера

					PE22.468332.001 ПЗ	Лис
						5
Зм.	Лис	№ докум.	Підпис	Дата		

mbedTLS – відкрита апаратна криптографічна бібліотека для реалізації алгоритмів шифрування, захисту TLS/SSL та генерації підписів

MDNS (Multicast DNS) – мережевий протокол для вирішення доменних імен у локальних мережах без використання централізованого DNS-сервера

MQTT (Message Queuing Telemetry Transport) – спрощений мережевий протокол передачі даних на основі патерну «публікація-підписка»

Mutex (Mutual Exclusion) – механізм взаємного виключення в ОС FreeRTOS для синхронізації потоків та захисту спільних ресурсів від одночасного доступу

NTP (Network Time Protocol) – Мережевий протокол для синхронізації внутрішнього часу пристрою із серверами точного часу

NVS (Non-Volatile Storage) – спеціалізований розділ флеш-пам'яті мікроконтролера для довгострокового збереження налаштувань у форматі «ключ-значення»

PETG (Polyethylene Terephthalate Glycol) – зносостійкий сополімер гліколю, поширений матеріал для функціонального 3D-друку

PLA (Polylactic Acid) – біологічно розкладний термопластичний поліефір на основі відновлюваних ресурсів

PROGMEM – керуюча директива мови C++ для примусового збереження статичних даних (веб-сторінок) безпосередньо у флеш-пам'яті (Flash-ROM) замість оперативної пам'яті (RAM)

REST API (Representational State Transfer) – архітектурний стиль взаємодії програмних компонентів за допомогою стандартних HTTP-запитів

SOC (State of Charge) – поточний рівень заряду акумуляторної батареї джерела безперебійного живлення, виражений у відсотках

SoC (System on a Chip) – система на кристалі (інтегральна схема, що об'єднує процесор, пам'ять та периферію на одному напівпровідниковому чипі)

					PE22.468332.001 ПЗ	Лис
						6
Зм.	Лис	№ докум.	Підпис	Дата		

SRAM / RAM (Static Random-Access Memory) – статична оперативна пам'ять мікроконтролера з довільним доступом

SSL / TLS (Secure Sockets Layer / Transport Layer Security) – криптографічні мережеві протоколи для забезпечення захищеного зв'язку та шифрування даних

UART (Universal Asynchronous Receiver-Transmitter) – універсальний асинхронний приймачепередавач (послідовний інтерфейс для апаратного зв'язку)

WIFI_AP_STA — Комбінований режим роботи бездротового модуля ESP32, при якому він одночасно діє як клієнт локальної мережі (Station) і як точка доступу (Access Point)

					РЕ22.468332.001 ПЗ	Лис
						7
Зм.	Лис	№ докум.	Підпис	Дата		

ВСТУП

Сучасний етап розвитку адитивних технологій характеризується масовим розгортанням 3D-ферм для серійного виготовлення деталей, що є особливо критичним для забезпечення волонтерських та оборонних потреб в Україні. Світовою тенденцією у цій галузі є перехід до повної автоматизації виробничих ліній. Проте ключовою прогалиною, що суттєво обмежує ефективність та безперервність роботи парків 3D-принтерів споживчого класу, залишається надмірна частка ручної праці. На ринку бракує доступних, відкритих рішень для дистанційного керування чергами друку та автоматичної зміни робочих платформ (APBC), які б до того ж враховували специфіку вітчизняних реалій – потребу в постійному моніторингу резервного живлення.

Актуальність розробки зумовлена гострою необхідністю створення доступної IoT-системи, здатної перетворити поодинокі 3D-принтери (зокрема, популярну серію Bambu Lab A1) на повністю автономні конвеєрні одиниці з віддаленим керуванням. Підставою для виконання дипломного проєкту є затверджене завдання кафедри прикладної радіоелектроніки.

Проєкт безпосередньо пов'язаний із розв'язанням актуальних логістичних та технологічних задач великих волонтерських ініціатив (наприклад, спільноти «ДрукАрмія»), спрямованих на оптимізацію часу операторів та ліквідацію простоїв обладнання.

Метою роботи є розробка спеціалізованого апаратно-програмного комплексу на базі мікроконтролера ESP32 для автоматизації процесу друку, фізичної зміни робочих платформ та віддаленого моніторингу телеметрії. Галуззю застосування результатів проєкту є комерційні дрібносерійні підприємства, волонтерські виробничі хаби та освітні інженерні лабораторії.

На основі проведеного аналізу потреб волонтерських спільнот та технічних можливостей обладнання було сформовано вимоги до проєкту, які наведено у Додатку А.

					PE22.468332.001 ПЗ	Лис
						8
Зм.	Лис	№ докум.	Підпис	Дата		

1 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

1.1 Програмні комплекси екосистеми Bambu Lab та хмарні IoT-платформи моніторингу

На сьогоднішній день екосистема компанії Bambu Lab, яка включає стаціонарні програмні пакети для слайсингу тривимірних моделей (Bambu Studio, OrcaSlicer), а також мобільні застосунки для віддаленого контролю (Bambu Handy), є конвенційним еталоном у сфері індивідуального моніторингу та оперативного керування одиничними пристроями адитивного виробництва. Зазначене програмне забезпечення функціонує за рахунок інтеграції закритих пропрієтарних протоколів та дає змогу кінцевому користувачеві в реальному часі відстежувати базові телеметричні показники: поточну та цільову температуру термоблока екструдера (сопла), температуру нагрівальної платформи (стола), швидкість обертання кулерів охолодження моделі та термокамери, а також отримувати низькочастотний відеопотік (Media Streaming) з інтегрованої камери пристрою.

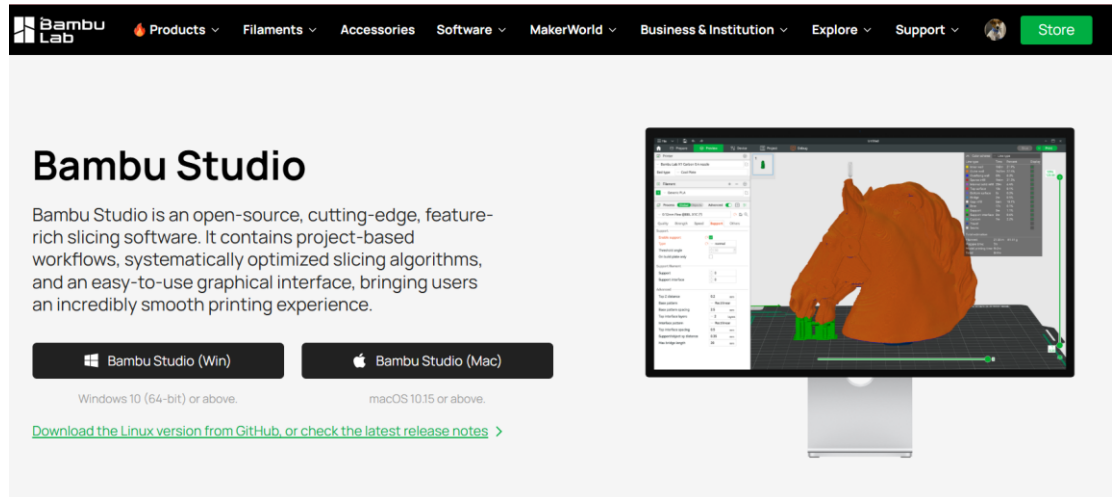


Рис. 1.1 – ПЗ Bambu Studio для ПК

					РЕ22.468332.001 ПЗ	Лис
						9
Зм.	Лис	№ докум.	Підпис	Дата		

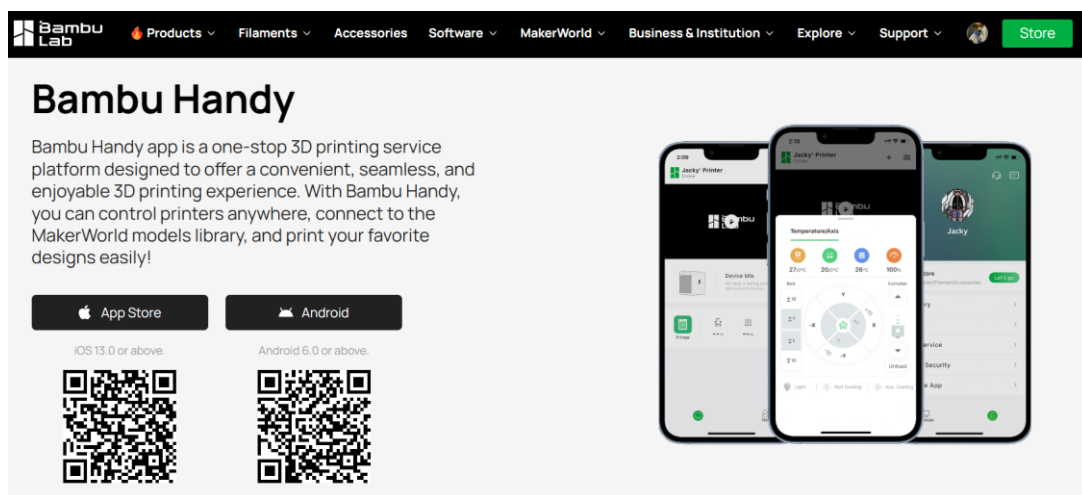


Рис. 1.2 – ПЗ Bambu Handy для мобільних пристроїв

Проте з інженерно-технічної точки зору даний програмний інструментарій має низку критичних архітектурних обмежень, які унеможливають його пряме використання для автоматизації промислових чи волонтерських 3D-ферм як єдиних виробничих кластерів.

Він орієнтується на поштучне (дискретне) виробництво. Логіка інтерфейсу та базовий програмний код екосистеми розраховані на концепцію «один користувач – один принтер». Система вимагає безпосереднього ручного підтвердження запуску кожного нового завдання друку оператором, що виключає можливість побудови безперервного технологічного циклу.

Крім цього, існує функціональний розрив між програмними платформами. Можливості мобільної версії Bambu Handy та стаціонарного слайсера Bambu Studio суттєво відрізняються на рівні доступних інструментів адміністрування пристроїв. Це спричиняє десинхронізацію дій оператора, ускладнює оперативне керування парком машин та викликає незручності під час масового масштабування виробництва.

Також немає інструментарію для керування кластером. В офіційному програмному забезпеченні повністю відсутній інтегрований менеджер динамічних черг для розподілу завдань між декількома незалежними одиницями обладнання. Не передбачено засобів автоматичного групування принтерів за типами матеріалів, пріоритетністю замовлень чи поточним технічним станом.

					РЕ22.468332.001 ПЗ	Лис
						10
Зм.	Лис	№ докум.	Підпис	Дата		

Але головним недоліком є відсутність програмних API або відкритих алгоритмів взаємодії з апаратними системами очищення та автоматичної зміни робочих поверхонь (гнучких магнітних пластин із пружної сталі). Закрита архітектура прошивки принтера не дозволяє програмно ініціювати чи верифікувати успішність виконання механічних маніпуляцій після завершення друку.

Паралельно існують універсальні відкриті та комерційні IoT-платформи моніторингу, такі як OctoPrint (із розгалуженою екосистемою плагінів), 3DPrinterOS, Repetier-Server, а також професійні хмарні сервіси управління великими парками пристроїв. Вони надають розробникам доступ до ширшого інструментарію диспетчеризації, дозволяючи збирати аналітику, логувати помилки та централізовано надсилати файли на друк. Однак їхнє практичне розгортання в сучасних умовах (зокрема на волонтерських засадах) супроводжується серйозними мережевими та безпековими ускладненнями.

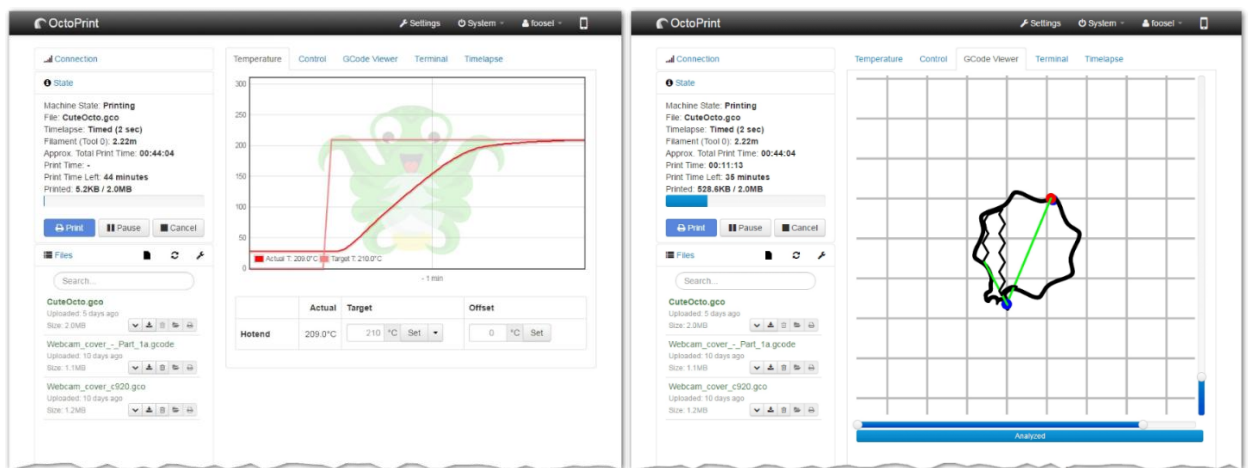


Рис. 1.3 – Інтерфейс OctoPrint

У першу чергу до них варто віднести потребу у складних мережових архітектурах. Для організації надійного віддаленого контролю за межами локальної мережі (LAN) оператор змушений конфігурувати статичні публічні IP-адреси, будувати захищені VPN-тунелі (на кшталт OpenVPN, WireGuard) або здійснювати небезпечне прокидання портів (Port Forwarding) на маршрутизаторах. Це створює високий поріг входження для користувачів без спеціальної технічної освіти.

					РЕ22.468332.001 ПЗ	Лис
						11
Зм.	Лис	№ докум.	Підпис	Дата		

Такий варіант також є вразливим до кібератак. Публікація керуючих веб-інтерфейсів у відкриту мережу Інтернет без належного адміністрування суттєво підвищує ризик несанкціонованого доступу злоумисників до керування нагрівальними елементами, що є критично небезпечним з точки зору пожежної безпеки виробничих приміщень.

Але, що найголовніше, у даному ПЗ не передбачено функціоналу керування автоматизованою фермою принтерів із наявною системою скидання пластини.

1.2 Аматорські та комерційні системи автоматичної зміни платформ

У сегменті DIY-модернізації (Do It Yourself) та дрібносерійного виробництва тривалий час ведуться інженерні пошуки рішень для автоматичного очищення робочих столів, загальновідомих як «Automated Plate Swap Systems» або «Automated Build Plate Changers» (APBC). Проведений аналіз дозволяє класифікувати ці розробки за принципом дії та архітектурою керування на три основні категорії, кожна з яких має суттєві конструктивні та експлуатаційні недоліки.

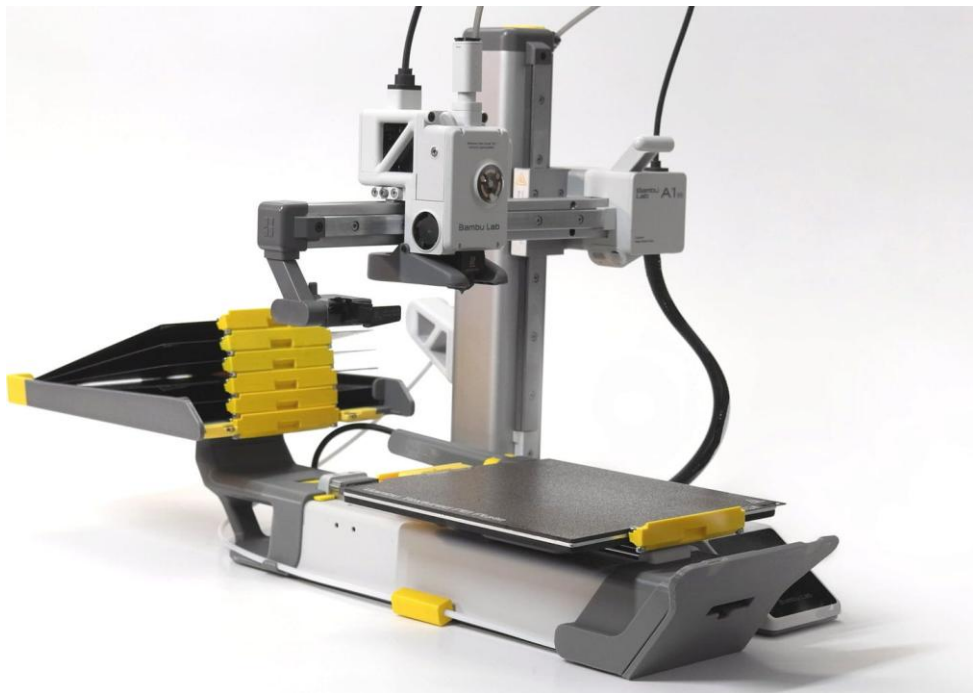


Рис. 1.4 – Swap Systems Swapmod Kit для Bambu Lab A1 Mini [12]

					<i>PE22.468332.001 ПЗ</i>	Лис
Зм.	Лис	№ докум.	Підпис	Дата		12

Пасивні системи на базі кінематики принтера. У таких системах (наприклад, поширені модифікації типу Swarpmo Kit (рис. 1.4)) скидання друкованої пластини або готового виробу здійснюється суто за рахунок механічної енергії крокових двигунів самого принтера. При завершенні друку друкуюча головка (екструдер) або сам нагрівальний стіл переміщуються у крайні координати, де спеціально закріплений упор або гак чіпляє край магнітної пластини, викликаючи її деформацію та зсув у приймальний лоток під дією пружності сталі чи сили тяжіння.

Іншим різновидом є пряме штовхання деталі зі столу радіатором або корпусом екструдера. Дані рішення характеризуються низькою надійністю та високим рівнем механічних ризиків: під час збирання великих або монолітних деталей із високою адгезією до покриття на конструктивні елементи друкуючої головки (напрямні, лінійні підшипники, ремені осей X/Y) діє значне нелінійне векторне навантаження, на яке кінематика принтера конструктивно не розрахована. Це призводить до прискореного люфту вузлів, деформації валів та зміщення геометрії (Layer Shift). Метод підходить виключно для поодиноких, геометрично високих деталей, які мають достатнє плече прикладання сили штовхання, і є абсолютно неефективним для масового друку низькопрофільних волонтерських або комерційних виробів.



Рис. 1.5 – Ферма автоматизована шляхом скидання виробів екструдером

Зм.	Лис	№ докум.	Підпис	Дата

PE22.468332.001 ПЗ

Лис
13

Існують також напівавтономні системи з додатковими актуаторами. Для зниження навантаження на кінематику принтера деякі розробники модернізують механізм, додаючи зовнішні виконавчі пристрої – сервоприводи, соленоїди або крокові двигуни, які приводять у дію штовхач. Керування цими актуаторами зазвичай намагаються реалізувати шляхом підключення до вільних пінів загального призначення (GPIO) на материнській платі 3D-принтера, або через сторонні керовані реле, підключені паралельно до силових ліній (наприклад, ліній живлення кулерів чи нагрівачів). Головним технологічним недоліком такого підходу є те, що сучасне обладнання (зокрема лінійки Bambu Lab A1/P1/X1) має повністю закриту архітектуру апаратного забезпечення та зашифровані прошивки. Будь-яке фізичне втручання в материнську плату, підпайка до внутрішніх шин даних чи модифікація силових ланцюгів призводять до неминучої втрати офіційної заводської гарантії на доволі дорогий пристрій, а також ризику виходу з ладу внутрішніх DC-DC перетворювачів напруги та вигорання портів центрального процесора принтера через відсутність гальванічної розв'язки.

Є і загальний програмний недолік існуючих рішень. Незалежно від механічного виконання, фундаментальним недоліком перерахованих аматорських та напівпрофесійних систем є необхідність ручної модифікації вихідного G-коду (.gcode) для кожного окремого завдання друку. Оператор змушений вручну або через специфічні профілі слайсера інтегрувати кастомні макроси та координатні скрипти в блок завершення друку. Це створює низку критичних проблем для виробничого процесу: високий поріг входження, адже кінцевий користувач або волонтер без глибоких знань синтаксису G-коду та архітектури конкретного принтера не здатний самотійно налаштувати систему без ризику влаштувати механічну аварію (зіткнення сопла зі столом чи елементами накопичувача), людський фактор та безальтернативність черги, оскільки найменша помилка в лінійній координаті руху або пропуск сервісної команди у файлі призводить до повного заклинювання механізму чи пошкодження магнітної платформи. Крім того, така схема унеможливорює побудову динамічної

					<i>PE22.468332.001 ПЗ</i>	Лис
						14
Зм.	Лис	№ докум.	Підпис	Дата		

черги: принтер здатний циклічно відтворювати лише один і той самий заздалегідь підготовлений файл, що повністю виключає гнучкість при зміні номенклатури виробів на фермі.

1.3 Порівняльний аналіз та обґрунтування розробки

Даний дипломний проєкт спрямований на подолання технологічного розриву між віддаленим програмним моніторингом обладнання для адитивного виробництва та повною фізичною автоматизацією конвеєрного циклу. Метою роботи є проєктування та практична реалізація інтегрованого апаратно-програмного комплексу, де зовнішній обчислювальний модуль на базі мікроконтролера ESP32 координує роботу швидкісного 3D-принтера Bambu Lab A1 та механічної підсистеми APBC (Automated Build Plate Changer) як єдиної замкненої екосистеми.

					РЕ22.468332.001 ПЗ	Лис
						15
Зм.	Лис	№ докум.	Підпис	Дата		

2 РОЗРОБКА МЕХАНІЧНОЇ ЧАСТИНИ

2.1 Особливості проектування та адаптації для принтерів типу «Bedslinger»

2.1.1 Кінематика принтерів з рухомим столом та обмеження FDM-друку

Механічна частина розробляється для принтерів з кінематикою типу «Bedslinger» (де стіл рухається по осі Y, а друкарська голова – по осях X та Z). Вони мають свою специфіку, яку необхідно враховувати при проектуванні автоматизації.



Рис. 2.1 – Зовнішній вигляд Bambulab A1 – популярного представника принтерів типу «Bedslinger»

По-перше, постійний рух стола створює значні інерційні навантаження, якщо вони стають надмірними, може виникати «ghosting» ефект. Будь-який додатковий механізм, змонтований безпосередньо на столі, збільшує його масу, що негативно позначається на якості друку (рис. 2.2) та зносі крокових

Зм.	Лис	№ докум.	Підпис	Дата

PE22.468332.001 ПЗ

Лис
16

двигунів. Тому система APBS проектувалася з максимальною оптимізацією ваги деталей та винесенням основних статичних вузлів так, аби вони не створювали додаткового навантаження.

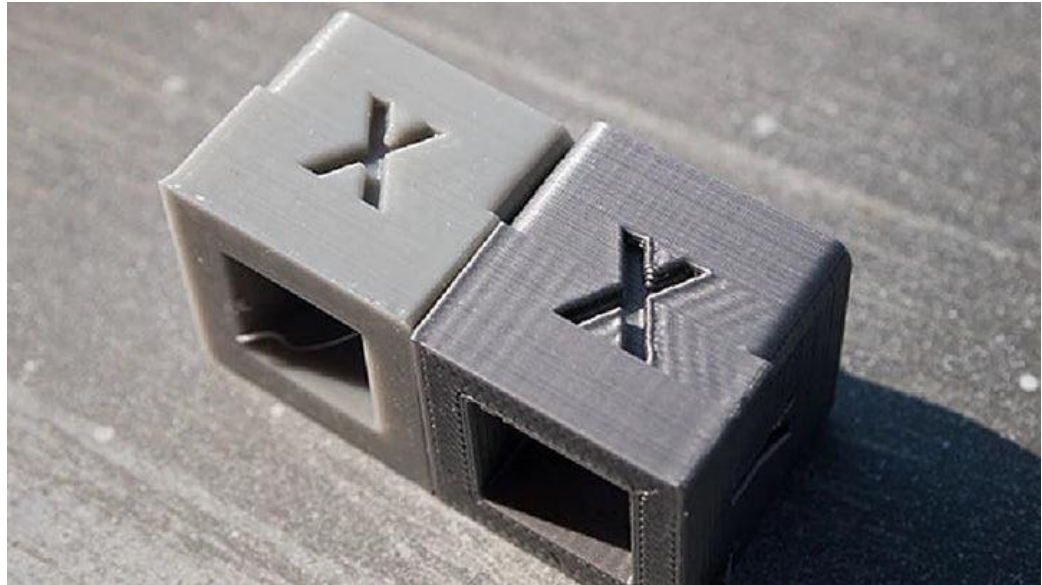


Рис. 2.2 – Вплив «ghosting» на деталі

По-друге, оскільки FDM-друк базується на пошаровому наплавленні гарячого пластику, геометрія надрукованих деталей має певні конструктивні обмеження. Перш за все, є необхідність урахування технологічних зазорів (допусків) на усадку пластику (для найпоширеніших PLA/PETG – в межах 0.1–0.3 мм). Крім цього, потрібно мінімізувати кути нахилу поверхонь, що потребують підтримок, задля забезпечення чистоти функціональних отворів та пазів. І, наостанок, варто враховувати орієнтацію деталей на друкарському столі під час виробництва таким чином, щоб вектори сил при роботі механізму не були спрямовані вздовж ліній міжшарового зчеплення (рис. 2.3), адже це запобігає розколюванню деталей.

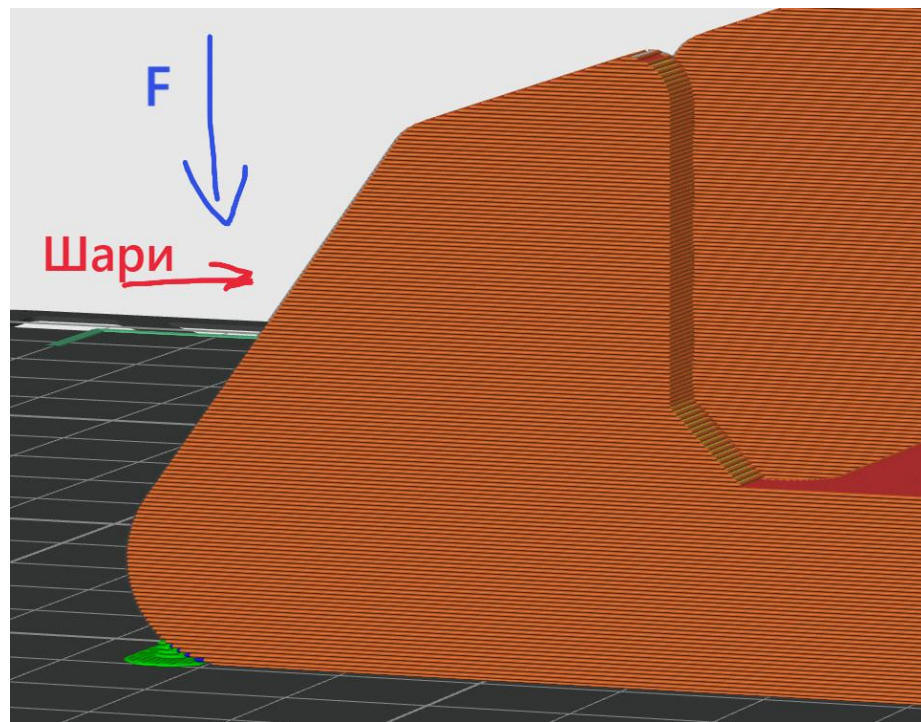


Рис. 2.3 – Візуалізація шарів при FDM-друці

Оскільки міцність зчеплення між окремими шарами філаменту є значно нижчою за міцність монолітного пластику, під час проектування геометрії зачіпного гака та напрямних системи APBC у САПР SolidWorks було враховано цей фактор. Всі силові елементи орієнтовані на робочій платформі так, щоб зусилля від пружної деформації магнітної пластини під час її скидання розподілялися вздовж осей X-Y (на зсув або стиснення), що повністю запобігає передчасному руйнуванню вузлів пристрою.

Окремої уваги заслуговує питання точності відтворення посадкових отворів та направляючих пазів при FDM-друці. Через термічне розширення та усадку пластику в процесі охолодження реальні розміри надрукованих отворів, як правило, є меншими за номінальні на 0,1–0,2 мм. З огляду на це, у CAD-моделях усіх сполучних елементів системи APBC розміри отворів під різьбові вставки М3 закладено з компенсаційним допуском відносно номінального діаметра.

Загальна маса надрукованих елементів системи APBC у зборі без кріпильних виробів на стандартних налаштуваннях (із заповненням 10-15%) становить приблизно 450-500 г, з них безпосередньо на рухомій платформі принтера

					<i>PE22.468332.001 ПЗ</i>	Лис
Зм.	Лис	№ докум.	Підпис	Дата		18

закріплено лише передній гак та нижню опорну частину касети сукупною масою 18-23 г. Це значення є суттєво меншим за граничнодопустиме додаткове навантаження на стіл принтера типу «Bedslinger», яке, відповідно до технічної документації Bambu Lab A1, не повинно перевищувати 250 г для збереження номінальної якості друку.

2.1.2 Обґрунтування та порівняльний аналіз матеріалів для виготовлення вузлів APBC

Оскільки однією з ключових вимог технічного завдання є можливість відтворення більшості структурних елементів системи APBC безпосередньо на базовому FDM-обладнанні кінцевого користувача, критично важливим етапом проектування є правильний вибір конструкційних термопластиків.

На компоненти розроблюваного пристрою під час експлуатації одночасно діють кілька несприятливих факторів:

1. Термічне навантаження: частина деталей (зокрема елементи фіксації пластини) знаходяться у безпосередній близькості до нагрівального стола 3D-принтера, температура якого під час друку найпоширенішими матеріалами в залежності від типу пластини становить 30°C-80°C (а для інженерних пластиків може досягати 100°C).
2. Динамічні та інерційні навантаження: постійний реверсивний рух стола по осі Y створює вібрації, а процес механічного зачеплення та скидання пружної сталевий пластини супроводжується короткочасними ударними навантаженнями.
3. Економічна доцільність (собівартість): для масового впровадження системи на волонтерських фермах (на кшталт «ДрукАрмії») ціна матеріалу має бути мінімальною, щоб не збільшувати фінансове навантаження на меценатів та друкерів.

З метою визначення оптимального балансу між міцністю, термостійкістю та фінансовою доступністю було проведено порівняльний аналіз поширених філаментів (таблиця 2.1).

					<i>PE22.468332.001 ПЗ</i>	Лис
						19
Зм.	Лис	№ докум.	Підпис	Дата		

Матеріал	Температура деформації (°C)	Коефіцієнт лінійної усадки (%)	Ударна міцність	Стійкість до зносу	Відносна вартість (за 1 кг)
PLA	~55–60	0.1–0.3	Низька	Середня	Низька (±500 грн)
PLA+	~58–62	0.1–0.3	Середня	Середня	Низька (±600 грн)
PETG	~75–80	0.2–0.4	Висока	Висока	Низька (±450 грн)
ABS / ASA	~95–100	0.6–1.5	Дуже висока	Дуже висока	Середня / Висока (±850 грн)

Таблиця 2.1 – Порівняльний аналіз фізико-механічних та економічних властивостей матеріалів

На основі проведеного аналізу для виготовлення елементів системи APBC було прийнято рішення щодо комбінованого використання матеріалів залежно від їхнього цільового призначення, бюджету, зручності друку та зони навантаження:

Матеріал PLA+ обрано для друку більшості масштабних корпусних деталей, вертикальних тримачів касети та зовнішніх елементів приймального лотка. Ці вузли конструктивно винесені за межі зони активного теплового випромінювання нагрівального стола принтера. Вибір PLA+ зумовлений його високою міжшаровою адгезією, мінімальною усадкою (що критично для запобігання деформацій великих деталей під час виготовлення) та підвищеною жорсткістю, яка забезпечує стабільність геометрії всієї рами заміниacza пластин. До того ж, працюючи з цим матеріалом можна відмітити зручність прибирання

					PE22.468332.001 ПЗ	Лис
						20
Зм.	Лис	№ докум.	Підпис	Дата		

підтримок, які потрібні в кількох місцях цих деталей. Економічний фактор PLA+ дозволяє суттєво знизити загальну собівартість конструкції.

Матеріал PETG є обов'язковим і рекомендованим для виготовлення елементів безпосередньої фіксації. Оскільки ці компоненти найближче контактують із найгарячішою частиною 3D-принтера – магнітною платформою стола, температура якої може досягати 80°C–100°C, використання PETG повністю запобігає термічній деформації та розм'якшенню робочих вузлів під час тривалих циклів безперервного друку, зберігаючи при цьому доступну вартість виробу.

Параметри друку для деталей із PLA+ були встановлені такі: температура сопла 220°C, температура стола 60°C, товщина шару 0,2 мм, заповнення 10-15% із патерном «Gyroid» для рівномірного розподілу навантажень у всіх напрямках. Деталі з PETG друкувались при температурі сопла 240°C, стола – 80°C, товщина шару 0,2 мм, заповнення 10-15% із заповненням «Gyroid».

2.1.3 Концептуальне моделювання та конструктивні елементи APBC

Основний концепт полягає у використанні «магазину» для чистих гнучких магнітних пластин та механізму їх поштучної подачі на стіл. Для зручнішої побудови і замірів крім використання штангенциркуля з власним обладнанням також було використано існуючу STEP модель Bambu Lab A1 [5], яка демонструє всі можливі положення стола та друкуючої голівки: крайнє положення стола і екструдера. Геометрія елементів зачеплення, напрямних стійок та вузлів кріплення до рами принтера типу «Bedslinger» була розроблена в САПР SolidWorks. Робочі креслення деталей підсистеми механіки APBC представлені у Додатку Е.

					<i>PE22.468332.001 ПЗ</i>	Лис
						21
Зм.	Лис	№ докум.	Підпис	Дата		

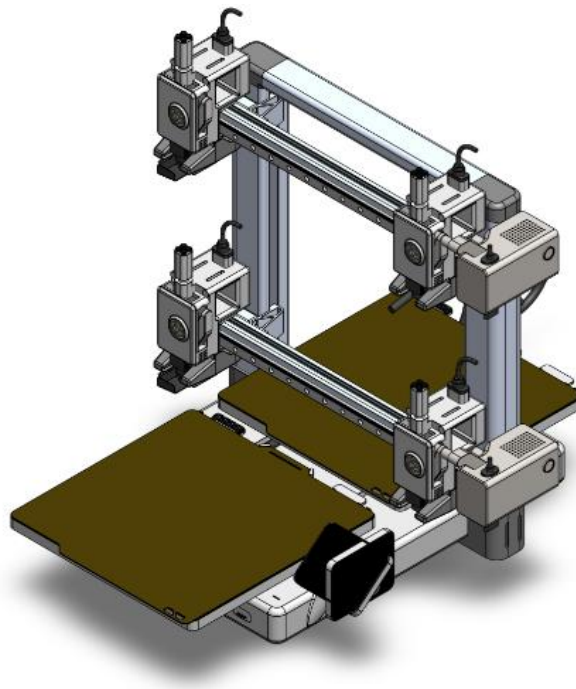


Рис. 2.4 – STEP-модель Bambulab A1 у всіх крайніх положеннях [14]

Ключові вузли конструкції включають «магазин» подачі чистих пластин, вертикальний тримач, що утримує стек пластин, передній гак, який підчеплює пластини і, впираючись у корпус, трохи їх піддіває, крайній гак і фіксація під них. Великі елементи було додатково розділено з врахуванням необхідності скріплення задля того, щоб вмістити на пластині і практично повністю нівелювати необхідність підтримок.

Задля того, щоб зачепити елементи до самого принтера, на їх нижніх частинах окрім повтору контурів додано штирі (рис. 2.5), які заходять у пази для гвинтів на корпусі. Таким чином вони надійно фіксуються без втручання в корпус приладу.

Зм.	Лис	№ докум.	Підпис	Дата

PE22.468332.001 ПЗ

Лис
22

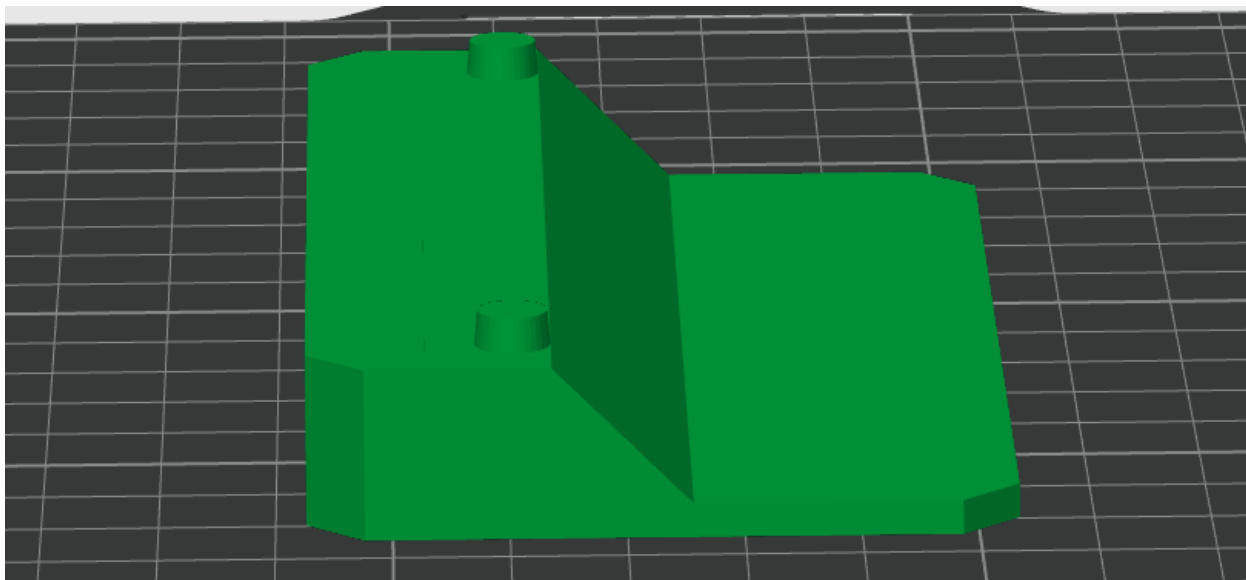


Рис. 2.5 – Штирі для закріплення

Ще одним критичним аспектом стала фізика взаємодії робочої поверхні зі столом принтера. Оригінальні друкарські пластини виготовлені з гнучкої пружної сталі і надійно утримуються на столі за допомогою сильного магнітного поля. Через високу силу магнітного тяжіння просте горизонтальне стягування пластини є неможливим – такий підхід призвів би до критичного перевантаження та пошкодження механізмів принтера. З огляду на це, кінематика зачіпного гака була спроектована з урахуванням необхідності локального відриву спочатку вгору, а вже потім уперед.

Заміну робочої поверхні інтегровано безпосередньо у логіку руху принтера: у момент завершення друку стіл по осі Y зміщується в крайнє положення, де за рахунок точного позиціонування та спеціально розрахованого кута зачеплення механічний гак підчіплює відпрацьовану пластину з готовою деталлю, спрямовуючи її у приймальний лоток. Наступна чиста пластина зі стеку під дією сили тяжіння та напрямних автоматично займає робоче положення на магнітній платформі стола. Такий підхід дозволив мінімізувати вагу рухомих частин і повністю уникнути ефекту «ghosting» під час друку.

Зм.	Лис	№ докум.	Підпис	Дата

PE22.468332.001 ПЗ

Лис
23

3 РОЗРОБКА ЕЛЕКТРОННОЇ ЧАСТИНИ

3.1 Загальна архітектура системи

Апаратно-програмний комплекс управління фермою 3D-принтерів Bambu Lab побудовано на базі мікроконтролера ESP32. Система реалізує розподілену архітектуру з кількома незалежними потоками виконання.

Архітектура передбачає три незалежні шари взаємодії:

1. Шар управління принтерами – комунікація через протокол MQTT/TLS локальною мережею.
2. Шар моніторингу резервного живлення – взаємодія з джерелами безперебійного живлення через REST API EcoFlow.
3. Шар дистанційного керування – управління фермою через Telegram Bot API.

Додатково реалізовано вбудований HTTP-сервер для локального веб-інтерфейсу та DNS-сервер для режиму первинного налаштування (Captive Portal).

Ключовою особливістю програмної архітектури є використання операційної системи реального часу FreeRTOS, інтегрованої в середовище розробки. Три окремі задачі (tasks) виконуються паралельно на двох ядрах процесора: TaskMQTT закріплено за ядром 1, тоді як TaskTelegram та TaskEcoFlow виконуються на ядрі 0. Такий розподіл дозволяє уникнути взаємних блокувань (deadlocks) під час мережевих операцій і забезпечує стабільне та швидке реагування на події принтера із затримкою не більше 50 мс. Схему роботи системи наведено в Додатку Б.

Для запобігання взаємним блокуванням (deadlocks) мережеві та прикладні задачі FreeRTOS розподілені між двома обчислювальними ядрами мікроконтролера відповідно до функціональної схеми взаємодії задач (Додаток В).

					<i>PE22.468332.001 ПЗ</i>	Лис
						24
Зм.	Лис	№ докум.	Підпис	Дата		

3.2 Обґрунтування вибору мікроконтролера та середовища розробки

У якості центрального обчислювального модуля системи було обрано плату розробника на базі системи на кристалі (SoC) ESP32, а саме готовий модуль ESP-WROOM-32.



Рис. 3.1 – Зовнішній вигляд ESP32-WROOM-32

Вибір ESP32 зумовлений його високою обчислювальною потужністю (двоядерний процесор із тактовою частотою до 240 МГц) та наявністю вбудованого апаратного модуля Wi-Fi (802.11 b/g/n), що є критично необхідним для взаємодії з принтерами у локальній мережі. Обсяг оперативної пам'яті (SRAM 520 КБ) дозволяє ефективно працювати з великими об'єктами формату JSON, які генерують 3D-принтери під час віддачі телеметрії, а також одночасно підтримувати роботу локального веб-сервера та SSL-шифрування.

Розробка програмного забезпечення для центрального комунікаційного модуля апаратно-програмного комплексу автоматизації адитивного виробництва базується на використанні фреймворку Arduino Core для мікроконтролера ESP32. З метою реалізації складних мережевих протоколів, криптографічних алгоритмів шифрування, асинхронного парсингу об'єктів телеметрії та взаємодії з енергонезалежною пам'яттю системи, було залучено комплекс стандартних та сторонніх відкритих бібліотек мови C++. Кожен програмний компонент обирався з урахуванням апаратних обмежень обсягу оперативної пам'яті (SRAM) та необхідності забезпечення високої швидкості виконання потоків в операційній системі реального часу FreeRTOS.

					<i>PE22.468332.001 ПЗ</i>	Лис
						25
Зм.	Лис	№ докум.	Підпис	Дата		

Повний вихідний код розробленого програмного забезпечення центрального комунікаційного модуля для мікроконтролера ESP32, написаний мовою C++ з використанням бібліотек Arduino Core та FreeRTOS, наведено у Додатку Ж.

Основою для побудови бездротового зв'язку є низькорівнева бібліотека WiFi.h, яка постачається у складі офіційного пакету підтримки платформ Espressif. Даний програмний модуль забезпечує повне керування вбудованим апаратним радіоблоком мікроконтролера, підтримуючи роботу пристрою як у режимі клієнта локальної мережі (Station mode), так і в режимі автономної точки доступу (Access Point mode). Для первинного конфігурування системи бездротова мережа ініціалізується у комбінованому режимі WIFI_AP_STA, що дозволяє пристрою розгортати сервісну сторінку та паралельно виконувати спроби аутентифікації у маршрутизаторі підприємства.

Для організації безпечної передачі даних через відкриті комп'ютерні мережі та створення захищених сокетів інтегровано бібліотеку WiFiClientSecure.h. Вона виступає критично важливим інфраструктурним прошарком, оскільки дозволяє встановлювати шифровані з'єднання за протоколами SSL/TLS. У розробленому коді окремі persistent-екземпляри даного клієнта використовуються для паралельного підключення до локальних MQTT-брокерів принтерів Bambu Lab на захищеному порту 8883 та для зв'язку з віддаленими хмарними серверами Telegram Bot API. Зважаючи на специфіку локальної роботи у LAN Only Mode, об'єкти конфігуруються за допомогою директиви setInsecure(), що дозволяє обходити процедуру жорсткої перевірки ланцюжка корневих сертифікатів (CA Certificate) та суттєво розвантажує обчислювальні ядра процесора.

Диспетчеризація завдань і обмін асинхронними повідомленнями з 3D-принтерами реалізовано за допомогою сторонньої бібліотеки PubSubClient.h, яка є індустріальним стандартом для впровадження архітектури клієнт-сервер за протоколом MQTT версії 3.1.1. Програмний модуль адаптовано під умови високого мережевого навантаження ферми друку шляхом примусового

					<i>PE22.468332.001 ПЗ</i>	Лис
						26
Зм.	Лис	№ докум.	Підпис	Дата		

розширення внутрішнього буфера приймання та передачі пакетів до 20480 байт за допомогою функції `setBufferSize()`. Це дозволяє утримувати та стабільно обробляти великі масиви даних телеметрії. Завдяки вбудованому механізму зворотного виклику (callback-функції) бібліотека забезпечує миттєву асинхронну обробку вхідних топіків звітності `report`, не блокуючи при цьому виконання інших паралельних обчислювальних процесів у системі.

Для десеріалізації та аналізу текстових повідомлень, що надходять від принтерів та серверів, застосовано оптимізовану бібліотеку `ArduinoJson.h`. Даний інструмент дозволяє працювати з об'єктами обміну JavaScript Object Notation безпосередньо у динамічній пам'яті (`heap`). Особливістю реалізації є використання об'єкта класу `JsonDocument` разом із методом динамічної потокової фільтрації `DeserializationOption::Filter()`. Програмний фільтр дозволяє виокремлювати із загального масиву даних лише цільові інженерні змінні, такі як поточна температура сопла, стан виконання коду ЧПК та відсоток готовності виробу. Такий підхід знижує пікове використання оперативної пам'яті мікроконтролера в три рази, гарантуючи стабільність пристрою при тривалій безперервній експлуатації.

Розгортання локального користувацького інтерфейсу та Captive-порталу реалізовано за допомогою зв'язки бібліотек `WebServer.h` та `DNSServer.h`. Модуль `WebServer` функціонує на стандартному порті 80 та забезпечує маршрутизацію вхідних HTTP-запитів, зв'язуючи визначені веб-ендпоінти (такі як `/status`, `/add_printer`, `/start_queue`) з відповідними обробниками C++. Для мінімізації навантаження на `heap`, розмітка сторінки завантажується безпосередньо з Flash-пам'яті програм за допомогою макросу `PROGMEM` та бібліотеки `pgmspace.h`. У свою чергу, утиліта `DNSServer` у режимі первинного налаштування перехоплює всі транзитні запити клієнтів на порту 53 та перенаправляє їх на локальну IP-адресу точки доступу, реалізуючи безбар'єрне підключення з будь-якого мобільного пристрою оператора.

Взаємодія з віддаленими хмарними серверами та API EcoFlow для надсилання запитів і зчитування квот споживання енергії виконується за

					<i>PE22.468332.001 ПЗ</i>	Лис
						27
Зм.	Лис	№ докум.	Підпис	Дата		

допомогою вбудованого модуля HTTPClient.h. Програмний компонент реалізує архітектурний стиль REST API та керує заголовками пакетів. Забезпечення високого рівня безпеки та автентифікації запитів до серверів EcoFlow Developer Portal реалізовано через системну бібліотеку нижнього рівня mbedtls/md.h, яка інтегрована в операційну систему ESP-IDF. Використання апаратних криптографічних засобів сімейства mbedTLS дозволило без залучення стороннього програмного забезпечення обчислювати складні цифрові підписи за алгоритмом HMAC-SHA256, що є обов'язковою умовою для авторизації сесій моніторингу резервного енергопостачання ферми.

Для забезпечення енергонезалежності системи та довгострокового збереження налаштувань підключень, API-ключів та бази файлів черги друку, використано бібліотеку Preferences.h. На відміну від застарілих методів емуляції EEPROM, цей програмний інтерфейс працює безпосередньо з сучасним розділом NVS (Non-Volatile Storage) флеш-пам'яті мікроконтролера. Бібліотека структурує дані за принципом «ключ-значення» всередині ізольованих просторів імен, автоматично контролює типи змінних та виконує перезапис виключно у разі зміни конфігурації. Це захищає напівпровідникову структуру чіпа від передчасного зносу комірок пам'яті.

Синхронізація системного часу електронного модуля з міжнародними стандартами часу UTC+3 реалізована за допомогою стандартних системних бібліотек time.h та sys/time.h через взаємодію з віддаленими NTP-серверами точного часу. Отримання валідної часової мітки (timestamp) виконується функцією gettimeofday() одразу після стабілізації Wi-Fi з'єднання. Наявність точного часу є критично необхідною для функціонування математичного апарату криптографічного підпису записів, оскільки сервери EcoFlow автоматично відхиляють будь-які транзакції, якщо розбіжність між внутрішнім часом мікроконтролера та еталонним часом супутника перевищує 30 секунд.

Для полегшення доступу до вебінтерфейсу в локальних мережах без використання складних цифрових IP-адрес інтегровано бібліотеку ESPmDNS.h. Програмний модуль реалізує підтримку мережевого протоколу Multicast DNS,

					<i>PE22.468332.001 ПЗ</i>	Лис
						28
Зм.	Лис	№ докум.	Підпис	Дата		

що дозволяє присвоїти контролеру постійне локальне доменне ім'я bambu.local. Це завершує формування гнучкого, надійного та стійкого до помилок програмного каркаса, який ефективно розподіляє обмежені апаратні ресурси SoC ESP32 для виконання завдань автоматизації виробництва.

3.3 Специфіка системи Bambu Lab та налагодження локального зв'язку

Сучасні 3D-принтери Bambu Lab (зокрема серія A1) мають закриту хмарну екосистему, проте підтримують локальний режим роботи (LAN Only Mode). У цьому режимі принтер самостійно виконує роль MQTT-брокера, що працює на стандартному захищеному порту 8883.

Авторизація на брокері відбувається з використанням заводського серійного номера пристрою (як ідентифікатора та основи дерева топіків) та локального коду доступу (Access Code), який генерується мікропрограмою принтера. Для переходу в локальний режим необхідно безпосередньо на екрані керування принтером провести відповідні налаштування.

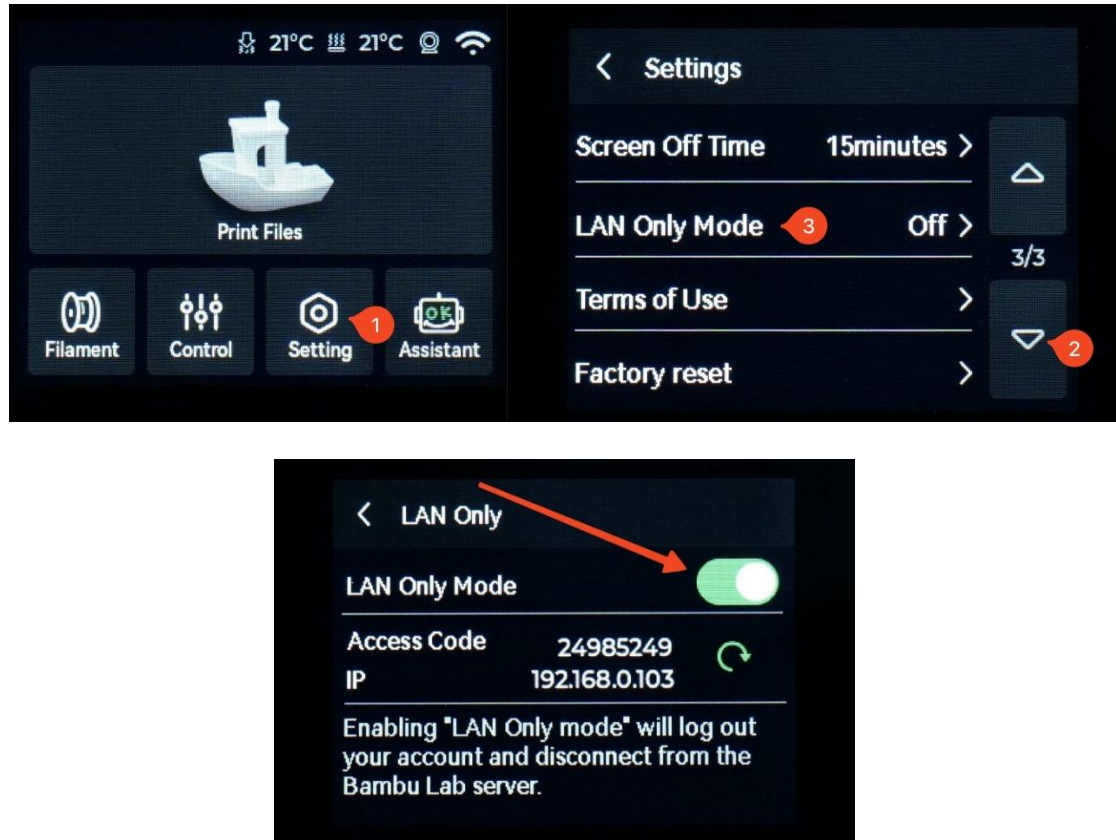


Рис. 3.2 – Налаштування локального режиму роботи [10]

					PE22.468332.001 ПЗ	Лис
Зм.	Лис	№ докум.	Підпис	Дата		29

Цих налаштувань достатньо, щоб отримувати інформацію від принтера. Однак, щоб надавати йому завдання (зокрема, запуску файлів на друк) закритою системою передбачено додаткову вимогу: необхідність активації режиму розробника (Developer Mode).

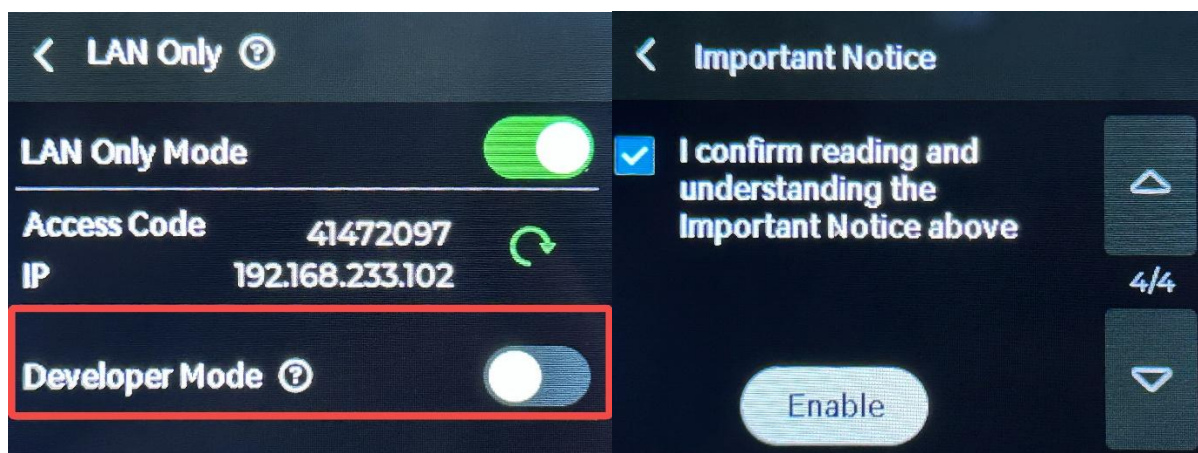


Рис. 3.3 – Перехід до режиму розробника [9]

3.4 Комунікаційний протокол MQTT та підключення принтерів

3.4.1 Особливості протоколу Bambu Lab MQTT

Принтери Bambu Lab використовують власну реалізацію протоколу MQTT версії 3.1.1 поверх TLS. Кожен принтер виступає брокером і водночас підписником власних топіків. Автентифікація здійснюється за схемою логін/пароль, де логін фіксований –«bblp», а паролем є восьмизначний Access Code принтера.

Топіки організовані за такою схемою:

device/{serial}/report – повідомлення від принтера до контролера

device/{serial}/request – команди від контролера до принтера, де {serial} – серійний номер конкретного пристрою.

Оскільки всі принтери підключені до одного обробника подій (callback-функції MQTT), диспетчеризація вхідних повідомлень виконується шляхом аналізу префіксу топіка.

					PE22.468332.001 ПЗ	Лис
						30
Зм.	Лис	№ докум.	Підпис	Дата		

```
//MQTT
void mqttCallback(char* topic, byte* payload, unsigned int length) {
    int prIdx = -1;
    for (int i = 0; i < printerCount; i++) {
        if (String(topic).startsWith("device/" + printerCfg[i].serial)) { prIdx = i; break; }
    }
    if (prIdx < 0) return;
}
```

Рис. 3.4 – Аналіз префіксу топіка

Зважаючи на обмеження оперативної пам'яті ESP32, при обробці об'ємних JSON-відповідей від принтера (розмір яких може сягати 20 КБ) застосовується метод потокової фільтрації на рівні десеріалізації за допомогою бібліотеки ArduinoJson. Це знижує пікове споживання пам'яті (heap) приблизно втричі порівняно з повним розбором документа.

```
JsonDocument filter;
filter["print"]["nozzle_temper"]      = true;
filter["print"]["bed_temper"]         = true;
filter["print"]["mc_percent"]         = true;
filter["print"]["mc_remaining_time"]  = true;
filter["print"]["subtask_name"]       = true;
filter["print"]["gcode_state"]        = true;
filter["print"]["layer_num"]          = true;
filter["print"]["total_layer_num"]    = true;
filter["print"]["spd_lvl"]            = true;
```

Рис. 3.5 – Метод потокової фільтрації

3.4.2 Управління з'єднанням та механізм pushall

За замовчуванням протокол Bambu Lab не надсилає повний стан принтера при підключенні. Для ініціалізації телеметрії необхідно надіслати команду pushall. Система реалізує дворівневий механізм синхронізації: перший pushall надсилається через 2 секунди після встановлення з'єднання (для стабілізації TLS-сесії), подальші запити формуються кожні 60 секунд.

					PE22.468332.001 ПЗ	Лис
						31
Зм.	Лис	№ докум.	Підпис	Дата		

Критично важливою для стабільності системи є логіка використання прапорця `mqttReady`. При першому підключенні або відновленні з'єднання цей прапорець ініціалізується як `false`. Перший отриманий звіт про стан принтера фіксується як «базовий» без генерування сповіщень у Telegram. Це запобігає появі хибних тривог (наприклад, повідомлень «принтер розпочав друк» у разі короткочасного розриву Wi-Fi з'єднання).

```
if (!st.mqttReady) {
    st.prevState = ns;
    st.state     = ns;
    st.mqttReady = true;
    Serial.printf("[MQTT] Printer %d initial state: %s\n", prIdx, ns.c_str());
    return;
}
```

Рис. 3.6 – Налаштування для стабільності системи

Команда запуску файлу на друк має найскладнішу структуру в даному протоколі. Аналіз реверс-інженерних даних показав, що мікропрограма принтера очікує специфічний JSON-об'єкт із дублюванням шляху до файлу у полях `file` та `url` (для зворотної сумісності з різними версіями мікропрограм):

```
p["file"] = "file:///sdcard/Auto/" + filename;
p["url"] = "file:///sdcard/Auto/" + filename;

p["bed_type"] = "auto";
p["bed_levelling"] = true;
p["flow_cali"] = false;
p["vibration_cali"] = false;
p["layer_inspect"] = false;
p["use_ams"] = false;
```

Рис. 3.7 – Команда запуску друку

Особливу увагу приділено полю `param`, яке має вказувати на відносний шлях до G-коду всередині 3MF-архіву («Metadata/plate_1.gcode»). Передача

					PE22.468332.001 ПЗ	Лис
						32
Зм.	Лис	№ докум.	Підпис	Дата		

прямой назви файлу в цьому полі призводить до помилки парсингу на стороні принтера.

3.5 Система черги друку

Логіка автоматизованої черги друку реалізована у вигляді масиву структур фіксованого розміру (до 30 позицій). Система підтримує два незалежні канали надходження завдань: локальний (через веб-інтерфейс) та віддалений (через Telegram-бота). Усі завдання зводяться до єдиної структури виконання, параметри якої наведені у таблиці 3.1.

Поле	Тип	Призначення	Значення за замовчуванням
items[]	String[30]	Імена файлів	—
total	int	Кількість завдань	0
index	int	Поточна позиція	0
active	bool	Черга активна	false
waiting	bool	Очікування між завданнями	false
printerIdx	int	Індекс цільового принтера	0

Таблиця 3.1 – Структура виконання

Між послідовними завданнями черги апаратно витримується пауза тривалістю 12 секунд (WAIT_BETWEEN_MS). Цей інтервал критично необхідний для того, щоб внутрішній контролер принтера після завершення попереднього циклу друку встиг програмно скинути статус FINISH та звільнити буфер для прийому нової команди project_file.

Алгоритм черги інтегровано з системою резервного живлення: процес автоматично призупиняється при виникненні критичного рівня заряду

					РЕ22.468332.001 ПЗ	Лис
Зм.	Лис	№ докум.	Підпис	Дата		33

зарядної станції, який користувач встановлює власноруч (коли прапорець `esco.critical` активний). Логіку роботи автоматизованого конвеєра, яка координує перевірку стану принтера та запуск наступного файлу з черги, виконано у вигляді циклічного алгоритму, наведеного у Додатку Г.

3.6 Модифікація керуючого G-коду 3D-принтера для інтеграції системи автоматичної зміни платформ

Для забезпечення повної автономності роботи розробленого апаратно-програмного комплексу та виключення потреби в ручному знятті надрукованих виробів, було проведено модифікацію стандартних пресетів керування 3D-принтера Bambu Lab A1.

Традиційний підхід до автоматизації передбачає надсилення сервісних команд ззовні. Проте в ході дослідження було виявлено, що стандартне відключення крокових двигунів (команда M18) після завершення друку призводить до втрати валідності координат за віссю Z. Як наслідок, будь-які подальші механічні маніпуляції з платформою викликали аварійне зіткнення каретки з рамою принтера.

Для розв'язання цієї проблеми алгоритм автоматичного очищення робочої зони та завантаження нової пластини було інтегровано безпосередньо у внутрішнє програмне забезпечення слайсера Bambu Studio (вкладка Machine end G-code). Це дозволяє виконувати сервісні переміщення осей без знеструмлення крокових двигунів, зберігаючи абсолютне позиціонування сопла у просторі.

3.6.1 Зміни в параметрах робочого простору

Оскільки механічна система автоматичної зміни платформ (APBC) потребує встановлення додаткових фізичних вузлів на раму та стіл принтера, у середовищі нарізки моделей було скориговано базову геометрію простору друку (Printable space).

					PE22.468332.001 ПЗ	Лис
						34
Зм.	Лис	№ докум.	Підпис	Дата		

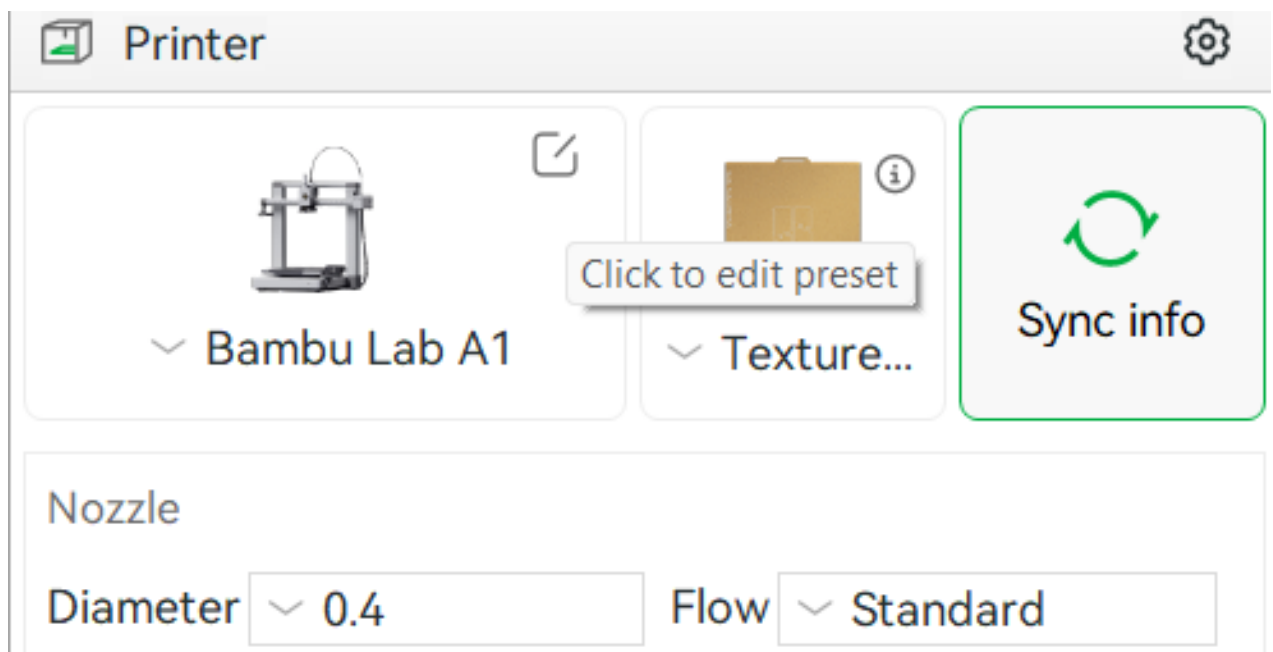


Рис. 3.8 – Розташування необхідного меню налаштувань

Максимальна висота друку (Printable height) була зменшена із заводських 256 мм до 253 мм. Це створює безпечний буферний простір об'ємом 3 мм, унеможливаючи випадкове зіткнення надрукованої деталі з рамою принтера під час процесу скидання пластини.

Зона виключення столу (Bed exclude area) задана матрицею координат: 0x0, 256x0, 256x15, 0x15. Дане налаштування програмно блокує смугу завширшки 15 мм уздовж переднього краю платформи, захищаючи встановлені там механічні елементи від контакту з соплом.

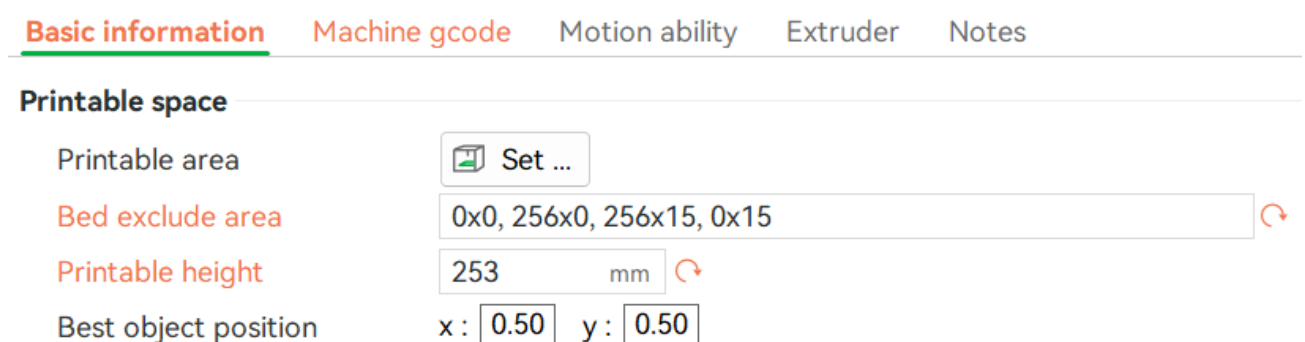


Рис. 3.9 – Зміна доступної площини друку та зон виключення

3.6.2 Структура інтегрованого сервісного коду очищення платформи

Для забезпечення повної автоматизації технологічного циклу комплексу APBC було здійснено глибоку модернізацію стартового (Machine start G-code) та фінішного (Machine end G-code) скриптів керування 3D-принтера Bambu Lab A1. Це дозволило реалізувати замкнену двоконтурну систему програмно-апаратного контролю: алгоритмічний захист від початку екструзії за відсутності друкарської поверхні на першому етапі та запуск фізичного циклу конвеєрної зміни робочих платформ на другому.

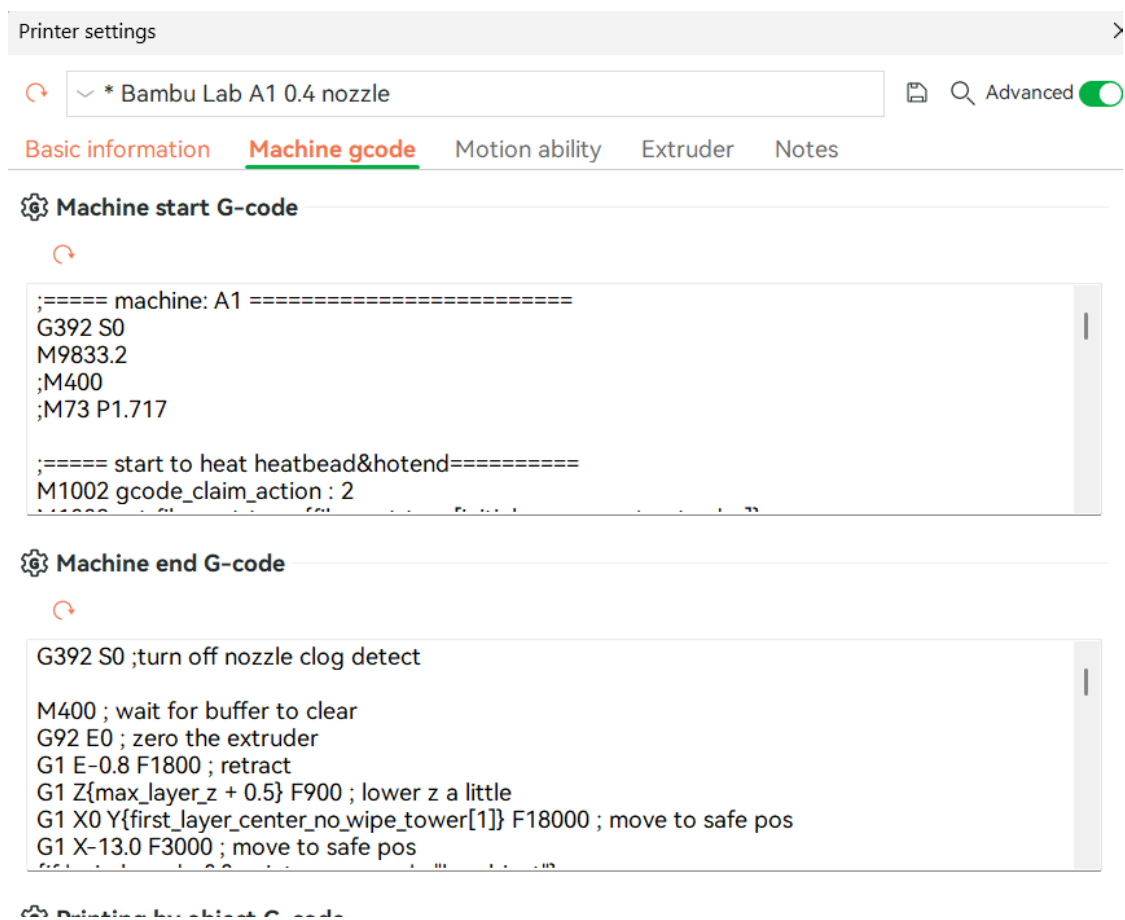


Рис. 3.10 – Зміна сервісного коду

У заводському варіанті стартового коду принтер здійснює первинні процедури калібрування без попередньої перевірки фактичного стану робочої поверхні. У розроблений кастомний пресет інтегровано систему умовних переходів на основі аналізу системних прапорці.

					РЕ22.468332.001 ПЗ	Лис
Зм.	Лис	№ докум.	Підпис	Дата		36

M1002 judge_flag build_plate_detect_flag: команда ініціює звернення до внутрішнього системного регістра мікропрограми принтера, який відповідає за прапорець активації функції детекції робочої пластини.

Блок умовного логічного переходу M622 S1 ... M623: у разі істинності аналізованого прапорця, керуюча логіка викликає спеціалізовану апаратну команду G39.4. Виконавчий пристрій за допомогою вбудованого в екструдер високоточного тензOMETричного датчика виконує фізичне сканування нагрівальної платформи для верифікації наявності гнучкої металевої пластини.

Якщо виконавчі механізми системи APBC з технічних причин (наприклад, заклинювання касети або вичерпання запасу розхідних елементів у накопичувачі) не змогли подати нову пластину, вбудована система контролю фіксує відсутність металевої поверхні. У цьому разі мікропрограма здійснює аварійну зупинку процесу друку (Abort), запобігаючи деструктивному нанесенню розплавленого полімеру безпосередньо на делікатну магнітну основу нагрівального столу.

Додатково в стартовому коді було модифіковано координати сервісних ліній очищення сопла. Замість стандартних заводських значень крайньої передньої межі робочого столу ($Y = -0.5$ та $Y = 1.0$) усі траєкторії примусово перенесені у позитивну безпечну зону – строго на координати $Y = 13$ та $Y = 15$. Це дозволило програмно ізолювати зону завширшки 15 мм уздовж переднього краю столу, унеможливлюючи зіткнення рухомого екструдера із встановленими там механічними фіксаторами та зачепами системи зміни платформ.

					РЕ22.468332.001 ПЗ	Лис
						37
Зм.	Лис	№ докум.	Підпис	Дата		

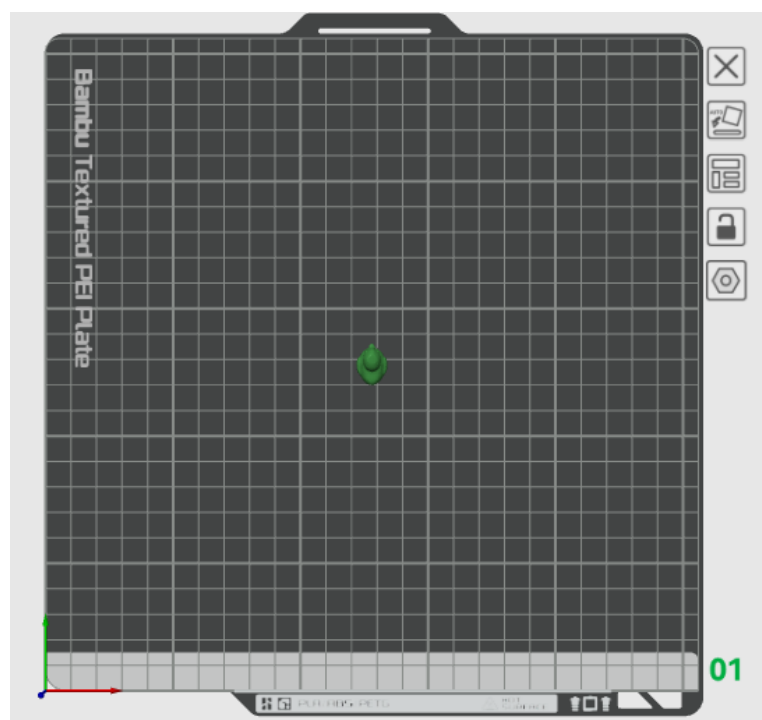
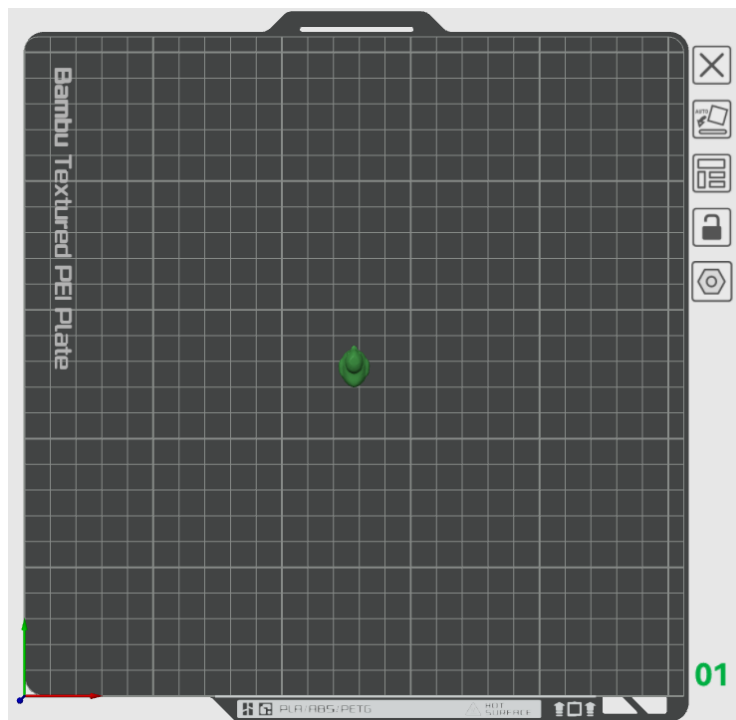


Рис. 3.11 – Модифіковані координати сервісної зони

Також у стартовому блоці після завершення механічних тестів (Mech mode fast check) було заблоковано виконання заводської інструкції примусового повторного позиціонування G28 X. Оскільки повторний транзитний рух у крайню ліву точку є надлишковим і створює додатковий ризик зачепу елементів касети.

Зм.	Лис	№ докум.	Підпис	Дата

PE22.468332.001 ПЗ

Лис
38

Найбільш масштабні зміни архітектури G-коду внесені у фінішний скрипт завершення друку. У базовій конфігурації слайсера після нанесення останнього шару моделі принтер виконує відведення сопла, позиціонує стіл за координатою $Y = 180$ та повністю деактивує живлення обмоток крокових двигунів командою M18. У кастомній модифікації проміжний виклик команди M18 повністю видалено, а виконавчий цикл автоматичного очищення столу ; --- AUTO-CLEAR START --- інтегровано безпосередньо всередину фінішного скрипту, коли обмотки крокових двигунів перебувають під напругою утримання.

Модифікований алгоритм забезпечує виконання таких інженерних вимог:

1. Збереження валідності просторових координат (Homing): утримання фіксованого магнітного поля в обмотках крокових двигунів протягом усього процесу скидання та завантаження платформ є критично необхідним. Передчасне вимкнення двигунів призвело б до мимовільного просідання осі Z під дією сили тяжіння, що викликало б втрату точок відліку та десинхронізацію кінематики. Завдяки модифікації система зберігає абсолютну прив'язку до робочого простору.
2. Кінематична безпека виконавчими осями: Переміщення столу та екструдера для скидання деталі здійснюється за чітко розрахованою траєкторією.

Команда повного знеструмлення крокових двигунів M18 X Y Z виконується виключно на фінальному кроці алгоритму – після того, як нова чиста пластина з касети надійно позиціонована на магнітній основі столу, а каретка екструдера опущена та переведена у безпечну нейтральну координату G1 Z200 F8000. Такий підхід повністю нівелює ризик пропуску кроків, заклинювання кінематичних вузлів чи механічного пошкодження компонентів розробленого апаратно-програмного комплексу.

					PE22.468332.001 ПЗ	Лис
						39
Зм.	Лис	№ докум.	Підпис	Дата		

Функціональний блок	Заводський G-код Bambu Lab	Модифікований G-код системи APBC	Технічне обґрунтування зміни
Координати зондування	Y-0.5, Y1.0	Y13, Y15	Створення зони виключення (15 мм) для захисту навісних механізмів від зіткнень.
Управління живленням ШД	Примусова активація M18 після друку моделей	Видалення проміжної M18, фіксація обмоток до кінця роботи конвеєра	Збереження абсолютної системи координат і валідності осі Z під час зміни платформ.
Алгоритм скидання деталей	Відсутній (ручне зняття оператором)	Інтегровано сервісний цикл ; --- AUTO-CLEAR START ---	Повна автоматизація процесу очищення робочої зони та завантаження нової пластини.
Калібрування осей XY	Примусове виконання G28 X після механічних тестів	Команду G28 X заблоковано (;G28 X)	Оптимізація часу та траєкторії руху екструдера, виключення ризику зачепу касети.

Таблиця 3.2 – Модифікації керуючого G-коду

3.7 Контроль енергопостачання через інтеграцію з EcoFlow

Оскільки ферма 3D-принтерів може експлуатуватися в умовах нестабільного енергопостачання, систему доповнено модулем моніторингу

					PE22.468332.001 ПЗ	Лис
						40
Зм.	Лис	№ докум.	Підпис	Дата		

резервного живлення станцій EcoFlow. Для реалізації цієї функції була проведена реєстрація на порталі для розробників EcoFlow (EcoFlow Developer Portal) з метою отримання унікальних ключів доступу (AppKey та SecretKey).

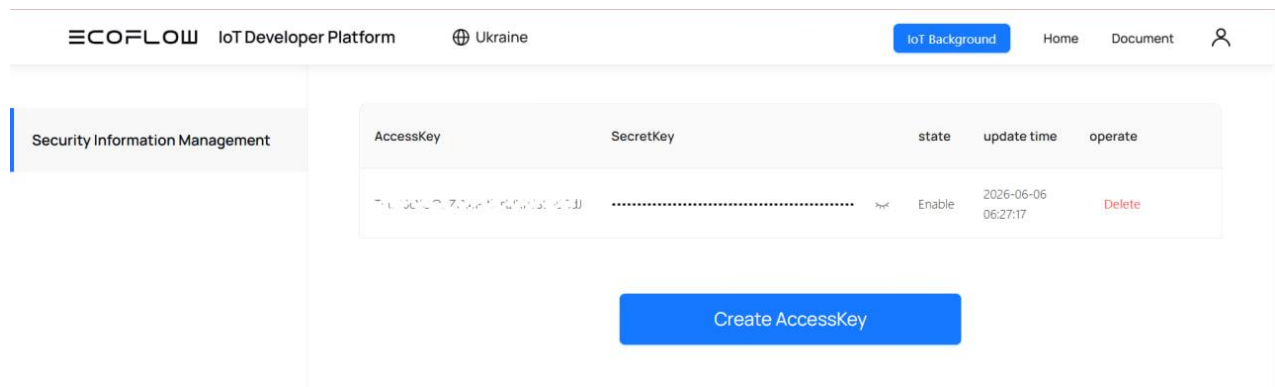


Рис. 3.12 – EcoFlow Developer Portal

3.7.1 REST API та підпис запитів HMAC-SHA256

EcoFlow надає відкритий REST API для отримання телеметрії зарядних станцій. Усі запити до API повинні бути підписані з використанням алгоритму HMAC-SHA256. Рядок для підпису формується з трьох обов'язкових параметрів у лексикографічному порядку:

`accessKey={appKey}&nonce={6-значне число}×tamp={мс Unix}`

Реалізація підпису виконана безпосередньо через бібліотеку mbedTLS, яка входить до складу ESP-IDF, без залучення сторонніх бібліотек:

```
String ecoflowSign(const String& params, const String& secret) {
    byte hmac[32]; char hex[65];
    mbedtls_md_context_t ctx;
    mbedtls_md_init(&ctx);
    mbedtls_md_setup(&ctx, mbedtls_md_info_from_type(MBEDTLS_MD_SHA256), 1);
    mbedtls_md_hmac_starts(&ctx, (const uint8_t*)secret.c_str(), secret.length());
    mbedtls_md_hmac_update(&ctx, (const uint8_t*)params.c_str(), params.length());
    mbedtls_md_hmac_finish(&ctx, hmac);
    mbedtls_md_free(&ctx);
    for (int i = 0; i < 32; i++) sprintf(&hex[i*2], "%02x", hmac[i]);
    return String(hex);
}
```

Рис. 3.13 – Створення рядка підпису за алгоритмом HMAC-SHA256

Часова мітка отримується через функцію `gettimeofday()` після синхронізації з NTP-серверами `pool.ntp.org` та `time.nist.gov` при старті системи. Синхронізація NTP є обов'язковою умовою коректної роботи підпису – відхилення більше 30 секунд призводить до відхилення запиту сервером EcoFlow.

3.7.2 Обробка телеметрії та автоматичне реагування

Система опитує API EcoFlow кожні 60 секунд і отримує такі параметри стану батареї: рівень заряду (SOC), вихідна потужність (споживання ферми), вхідна потужність (зарядження від мережі або сонячних панелей), розрахунковий час автономної роботи.

На основі цих даних реалізовано два сценарії автоматичного сповіщення в Telegram. По-перше, виявлення відключення мережевого живлення – якщо вхідна потужність змінилася з ненульового значення до нуля, генерується термінове повідомлення з поточним рівнем заряду та споживанням. По-друге, виявлення відновлення живлення – зворотна ситуація. Логіка виявлення базується на порівнянні поточного та попереднього значень вхідної потужності через поле `eco.prevInPower`.

Критичний поріг спрацьовує лише при одночасному виконанні двох умов: рівень заряду не перевищує налаштований поріг (за замовчуванням 20%) і відсутнє вхідне живлення. Це запобігає хибному спрацьовуванню під час зарядження при низькому рівні SOC.

3.8 Користувацькі інтерфейси

Оскільки кінцевий продукт розрахований на звичайного користувача, для забезпечення безбар'єрного керування фермою розроблено два рівні клієнтських інтерфейсів, які дозволяють проводити необхідний контроль та налаштування як на місці з ПК, так і на відстані з мобільного пристрою.

3.8.1 Локальна веб-панель

Веб-інтерфейс реалізований у вигляді єдиного HTML-файлу, що зберігається в секції Flash-пам'яті ESP32 за допомогою макросу `PROGMEM`. Вся

					PE22.468332.001 ПЗ	Лис
						42
Зм.	Лис	№ докум.	Підпис	Дата		

клієнтська логіка (JavaScript), стилі (CSS) та розмітка (HTML) об'єднані в один документ розміром близько 12 КБ. Такий підхід не потребує файлової системи SPIFFS/LittleFS і забезпечує миттєву відповідь без читання зі сховища.

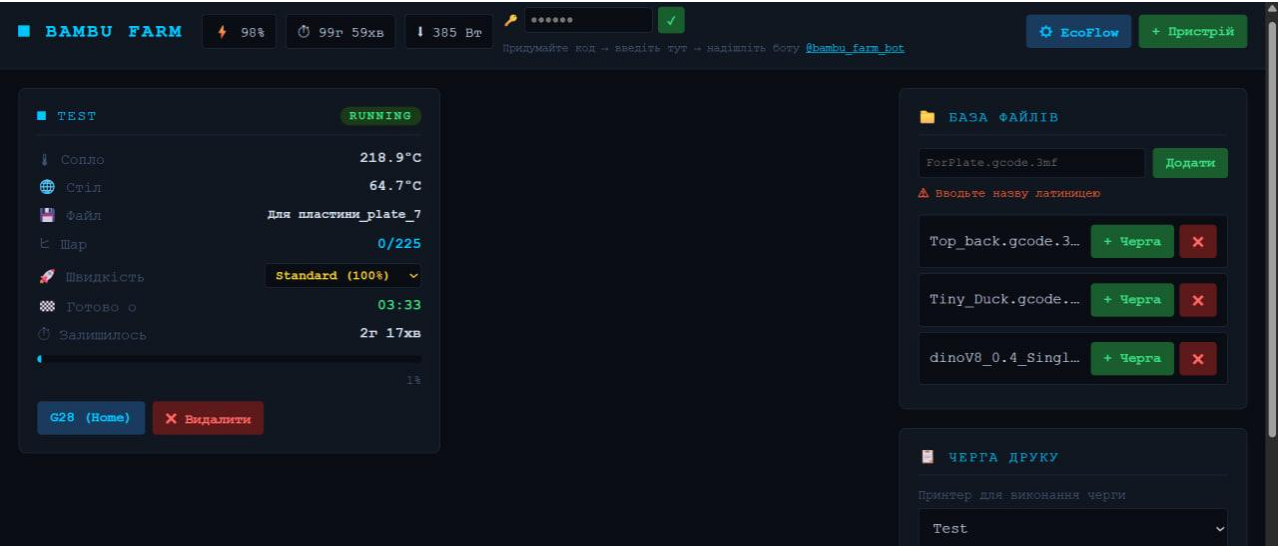


Рис. 3.14 – Загальний вигляд веб-інтерфейсу під час роботи

Інтерфейс відображає стан принтерів, що оновлюється кожні 2,5 секунди через GET-запит до ендпоінта /status. Відповідь формується як компактний JSON без додаткових бібліотек, шляхом послідовного з'єднання кількох рядків. Це дозволяє уникнути алокації великих JSON-документів у heap та знижує затримку відповіді.

Ендпоінт	Параметри	Дія
GET /status	—	JSON стану всіх пристроїв
GET /add_printer	name, ip, pass, sn	Додати принтер
GET /del_printer	id	Видалити принтер
GET /start_queue	pi, q (CSV)	Запустити чергу
GET /set_tg_pass	p	Встановити код доступу TG
GET /set_speed	i, lvl	Змінити швидкість принтера
GET /save_eco	sn, key, sec, pause	Зберегти налаштування EcoFlow
GET /home	i	Команда G28 (Home)
GET /files	—	Список файлів (JSON)
GET /add_file	name	Додати файл до бази
GET /del_file	id	Видалити файл

Таблиця 3.3 – Ендпоінти, які надає HTTP-сервер

3.8.2 Режим початкового налаштування (AP Mode)

При першому запуску або за відсутності збережених облікових даних Wi-Fi система автоматично переходить у режим точки доступу (AP Mode). ESP32 розгортає Wi-Fi мережу «BambuFarm-Setup» і запускає DNS-сервер, що перенаправляє всі запити на локальну сторінку налаштування (Captive Portal). Це дозволяє підключитися до пристрою зі смартфона без знання IP-адреси.



Рис. 3.15 – Вигляд веб-інтерфейсу при першому підключенні до мережі ESP

Після введення облікових даних мережі ESP32 паралельно підключається до зазначеної мережі у режимі WIFI_AP_STA і повертає отриману IP-адресу через ендпоінт /wifi_ip. При успішному підключенні користувач натискає кнопку підтвердження, після чого пристрій перезавантажується та переходить у штатний режим роботи.

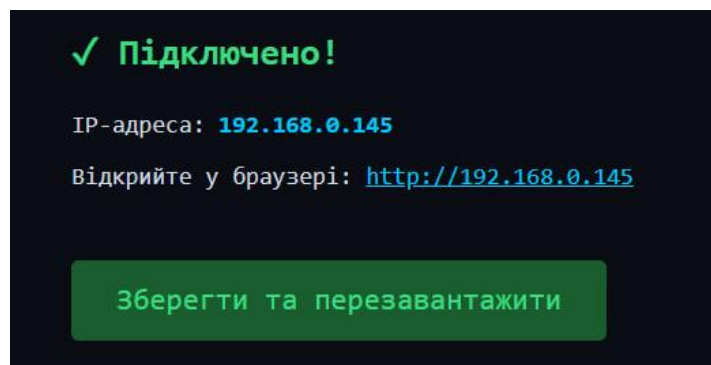


Рис. 3.16 – Підказка для користувача після введення даних для входу у локальну мережу

3.8.3 Telegram-бот

Інструментом глобального віддаленого моніторингу за межами локальної мережі виступає Telegram-бот. На відміну від класичного веб-доступу, інтеграція з Telegram API (метод Long Polling) не вимагає від користувача наявності статичної публічної IP-адреси («білої» IP) чи налаштування прокидання портів (Port Forwarding) на роутері. Бот дозволяє отримувати push-сповіщення про завершення друку, критичні помилки або розрядку станції EcoFlow безпосередньо на мобільний пристрій оператора.

					PE22.468332.001 ПЗ	Лис
						45
Зм.	Лис	№ докум.	Підпис	Дата		

Система використовує дворівневу модель безпеки. На першому рівні – статичний токен Telegram Bot API, вшитий у мікропрограму розробником. На другому рівні – динамічний код доступу, що встановлюється кінцевим користувачем через веб-інтерфейс. Токен ідентифікує бота, код доступу – авторизує конкретний обліковий запис Telegram.

Після успішної авторизації система прив'язує Telegram Chat ID до конфігурації і зберігає його в EEPROM (Preferences). Усі наступні повідомлення перевіряються за збігом Chat ID. Зміна коду доступу автоматично скидає прив'язку (обнуляє tgChatId), що вимагає повторної авторизації – це захищає від несанкціонованого доступу при компрометації коду.

Система використовує механізм long polling для отримання оновлень від Telegram Bot API. Запит getUpdates з параметром timeout=3 утримує з'єднання до 3 секунд, якщо нових повідомлень немає. Параметр offset (lastUpdateId + 1) забезпечує отримання лише непрочитаних оновлень і запобігає повторній обробці.

Для уникнення конкурентного доступу до єдиного TLS-з'єднання tgClient між задачею Telegram (polling) та іншими задачами (надсилання сповіщень) використовується mutex FreeRTOS. Таймаут захоплення mutex становить 5 секунд, після чого операція скасовується з логуванням помилки – це запобігає взаємному блокуванню задач.

Оброблені оновлення зберігаються в буфері tgUpdQueue і обробляються після звільнення mutex. Такий підхід мінімізує час утримання mutex і дозволяє задачі MQTT надсилати сповіщення навіть під час активного polling.

Інтерфейс Telegram-бота побудований на основі inline-клавіатур – інтерактивних кнопок, прикріплених до повідомлень. Кожна кнопка містить поле callback_data з командою та, за необхідності, індексом принтера або файлу. Система розпізнає такі типи команд:

status – відображення поточного стану всіх принтерів та EcoFlow

list_files – перелік файлів з бази даних для додавання до черги

					<i>PE22.468332.001 ПЗ</i>	Лис
						46
Зм.	Лис	№ докум.	Підпис	Дата		

my_queue – перегляд та управління особистою чергою бота

addq_{n} – додавання n-го файлу до черги бота

runq_{n} – запуск черги бота на n-й принтер

rmq_{n} – видалення n-ї позиції з черги

pause_{n} / resume_{n} – пауза або відновлення друку на n-му принтері

clear_q – очищення всіх черг

Черга Telegram-бота (tgBotQueue) є окремою від основної веб-черги (queue). Це дозволяє одночасно формувати чергу через веб-інтерфейс і через бота. При запуску через команду runq_ вміст черги бота копіюється в основну чергу, після чого очищається.

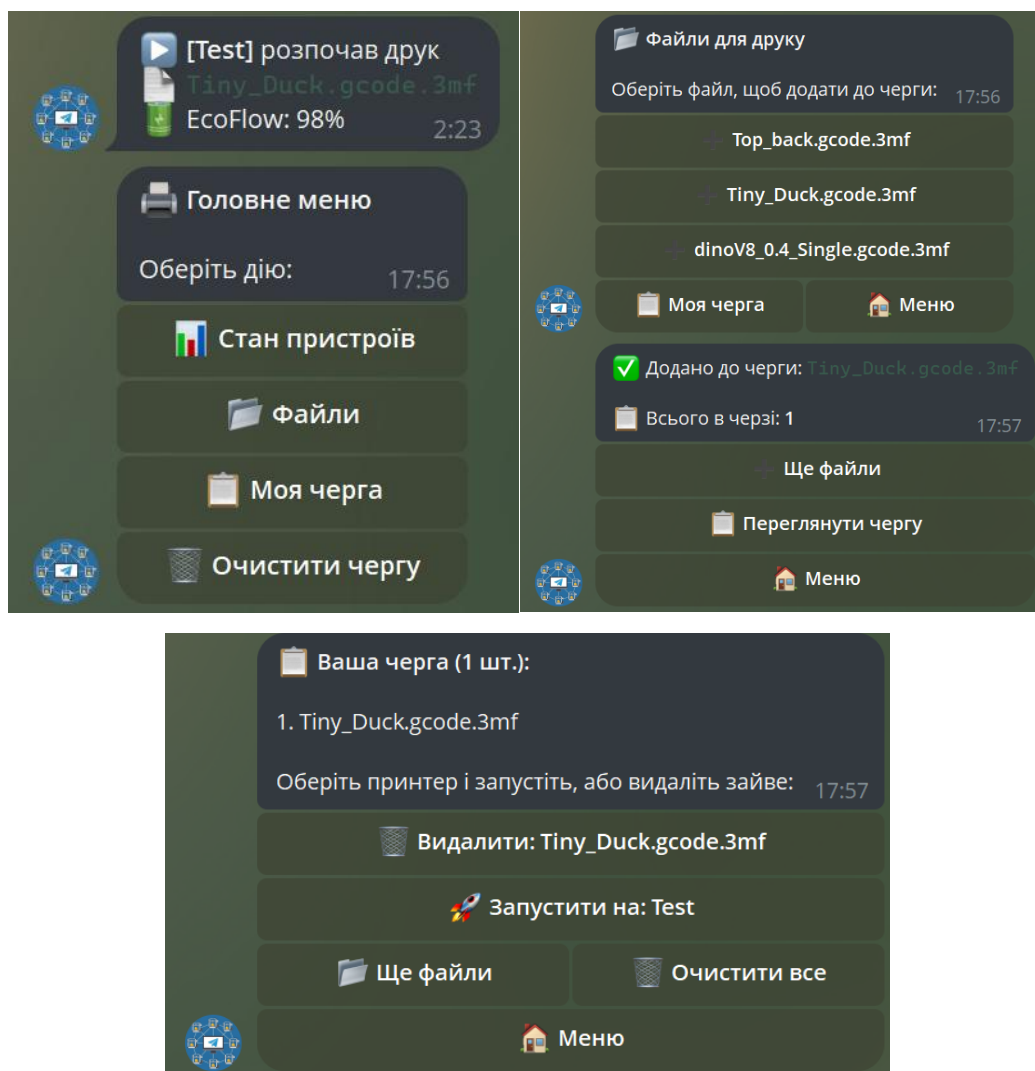


Рис. 3.17 – Доступні для користувача можливості

3.9 Збереження конфігурації

Запис системних параметрів здійснюється у розділ NVS (Non-Volatile Storage) Flash-пам'яті ESP32 з використанням бібліотеки Preferences. Для уникнення колізій даних конфігурація сегментована на два незалежні простори імен:

«cfg» – налаштування підключень принтерів, токенів Telegram та ключів EcoFlow;

«files» – база імен файлів для друку.

Дані структуровано за принципом «ключ-значення» з використанням індексних префіксів (наприклад, p0_name, p1_ip), що забезпечує стабільну роботу до чотирьох паралельно підключених принтерів. Фізичний запис в NVS-пам'ять ініціюється виключно у разі виявлення змін конфігурації, що нівелює проблему зносу комірок Flash-пам'яті мікроконтролера.

3.10 Конструктивне виконання захисного корпусу та ергономіка електронного модуля

Для забезпечення надійної експлуатації розробленого комунікаційного пристрою в умовах реального виробничого середовища 3D-ферми було спроектовано захисний кастомний корпус. Моделювання тривимірної геометрії корпусу виконувалось у системі автоматизованого проектування (САПР) SolidWorks. Конструкція розроблялась із урахуванням вимог технічної естетики, ергономіки, промислового дизайну, а також захисту від електростатичних розрядів (ESD) й випадкових механічних дій оператора.

Захисний корпус має двокомпонентну збірну структуру, що складається з базової основи (нижньої кришки), де фіксується плата розширення з мікроконтролером ESP32, та верхньої захисної панелі. Сполучення частин корпусу між собою реалізовано за допомогою невеликих пазів по боках. Матеріалом для виготовлення корпусу було обрано термостабільний та ударостійкий

					<i>PE22.468332.001 ПЗ</i>	Лис
						48
Зм.	Лис	№ докум.	Підпис	Дата		

кополімер PETG, що наносився методом пошарового наплавлення (FDM) на принтерах Bambu Lab.

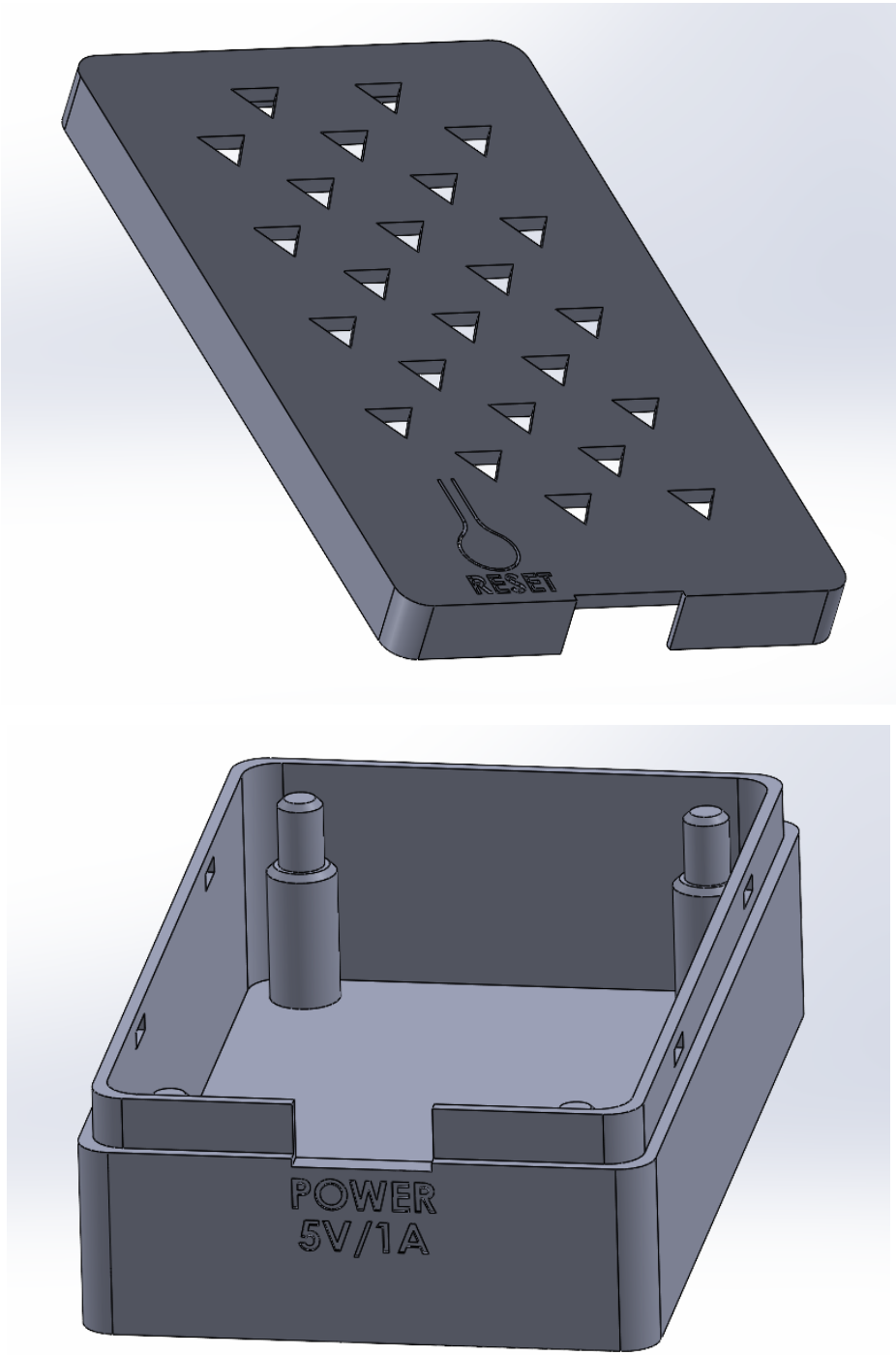


Рис. 3.18 – Зовнішній вигляд корпусу

З метою мінімізації впливу людського фактора на стабільність роботи автоматики ферми, архітектура корпусу передбачає суворе обмеження фізичного доступу користувача до внутрішніх компонентів системи. Креслення основи та верхньої панелі з вентиляційною перфорацією наведені у Додатку Е.

Конструкція повністю ізолює системну шину, контактні майданчики та GPIO-піни плати ESP32 всередині монолітного захисного об'єму. Таке інженерне рішення повністю виключає ризик випадкового короткого замикання або пошкодження делікатних напівпровідникових структур статичною електрикою з рук обслуговуючого персоналу ферми (доступу до пінів немає).

Для взаємодії оператора з пристроєм на зовнішні грані корпусу виведено лише два критично важливі інтерфейси:

1. Роз'єм інтерфейсу живлення Type-C: вікно під роз'єм спроектоване з технологічним допуском по периметру, що забезпечує безперешкодне підключення кабелів живлення з будь-якою товщиною пластикового штекера та гарантує щільну фіксацію кабелю без зламу контактного гнізда плати.
2. Елемент примусового апаратного скидання процесора EN (Reset): для запобігання випадковому перериванню поточної автоматизованої черги друку через ненавмисне доторкання, кнопку спроектовано так, щоб натискання було можливе виключно за допомогою спеціалізованого тонкого інструмента (сервісної скріпки або штифта), що реалізує тактильний захист високого рівня та унеможливорює випадкові збої в роботі програми.

Усі доступні для користувача функціональні зони на корпусі мають стійкі візуальні підказки, виконані в середовищі САПР методом втисненого дебосингу на глибину 0.2 мм (що відповідає висоті одного шару друку та виключає вивалювання внутрішніх елементів літер при виготовленні). Порт живлення забезпечений маркуванням номінальних параметрів постійного струму «Power 5V = 1A», а кнопка скидання чітким міжнародним ідентифікатором «RESET».

Для запобігання термічному перевантаженню SoC ESP32 в моменти пікових обчислень (під час активної роботи Wi-Fi модуля та розрахунку криптографічних SSL-сертифікатів для зв'язку з серверами Telegram), у верхній грані

					<i>PE22.468332.001 ПЗ</i>	Лис
						50
Зм.	Лис	№ докум.	Підпис	Дата		

двокомпонентного корпусу передбачено геометричний масив пасивних вентиляційних отворів (перфорацію). Це оптимізує внутрішні конвекційні потоки повітря та забезпечує стабільний температурний режим електронних компонентів у межах норми протягом усього цілодобового циклу роботи ферми.

					РЕ22.468332.001 ПЗ	Лис
						51
Зм.	Лис	№ докум.	Підпис	Дата		

4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ТА ВЕРИФІКАЦІЯ РОБОТИ ПРОТОТИПУ

4.1 Опис експериментального стенду та методика тестування

Для проведення експериментальних досліджень, верифікації розроблених алгоритмів та оцінки надійності механічних вузлів автоматизації було розгорнуто діючий прототип комплексу. Структура експериментального стенду містить такі апаратно-програмні компоненти:

1. 3D-принтер Bambu Lab A1 з активованим локальним режимом роботи (LAN Only Mode та Developer Mode).
2. Модуль ESP32-WROOM-32), у розробленому двокомпонентному захисному корпусі із пластику PETG.
3. Зарядна станція EcoFlow RIVER 2 Pro, підключена до локальної мережі.
4. Підсистема APBC, що складається з передньої частини (статичний і рухомий гак та кріплення для них, задньої частини («магазин» для пластин і кріплення для нього з фіксатором), фіксаторів для гнучких пластин і кріплення для стола.

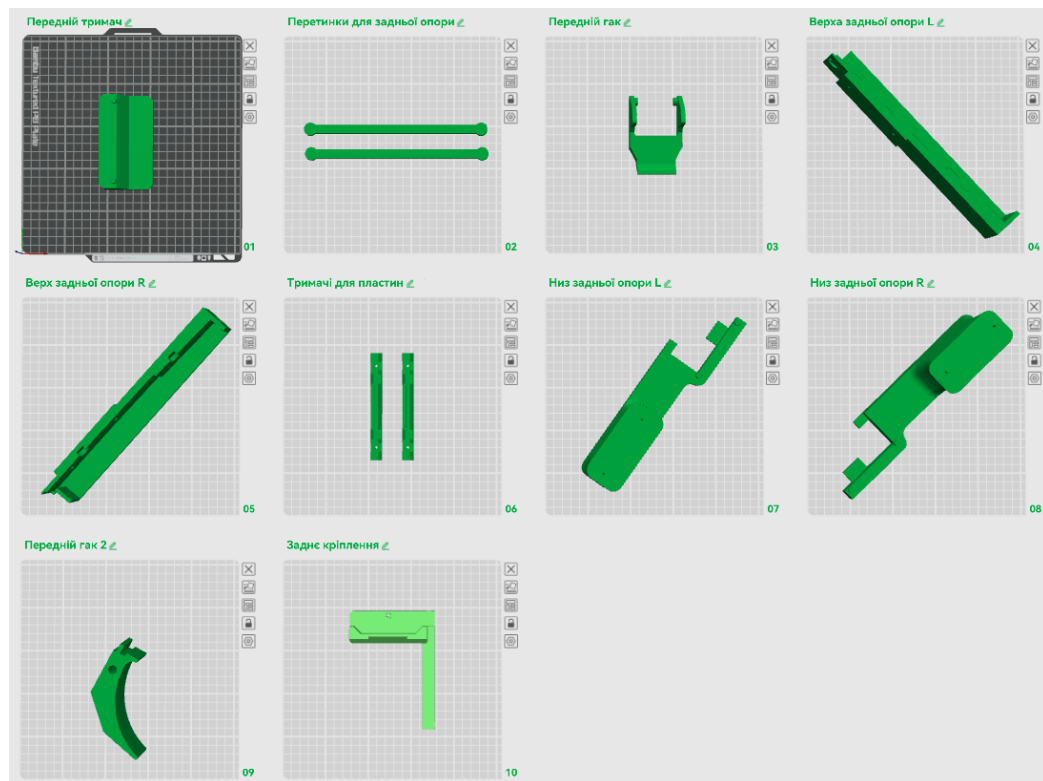


Рис. 4.1 – Повний набір деталей у слайсері Bambulab

					РЕ22.468332.001 ПЗ	Лис
Зм.	Лис	№ докум.	Підпис	Дата		52

Методика тестування передбачала циклічне відтворення черги друку, що складається з двох послідовних завдань (файлів тестових моделей у форматі .gcode.3mf). У ході експерименту фіксувалися такі параметри: стабільність утримання TLS 1.2 сесій з локальним MQTT-брокером принтера та хмарию Telegram Bot API та точність позиціонування осей при використанні модифікованого фінішного G-коду.

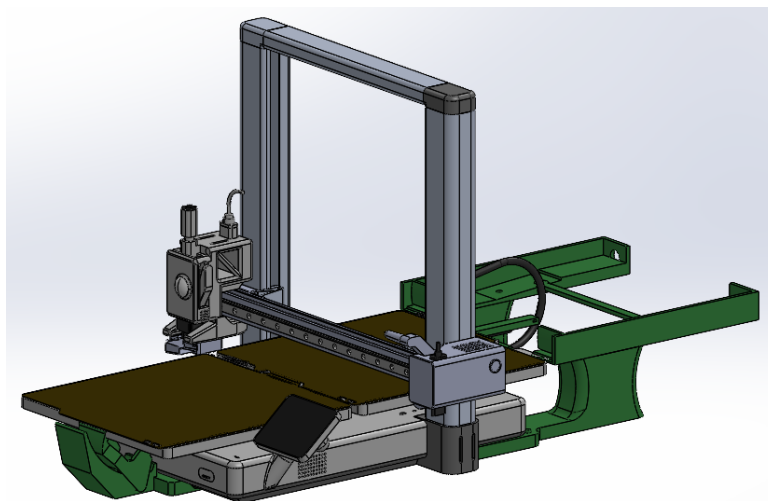


Рис. 4.2 – Трьохвимірна CAD-модель експериментального стенду у зборі в САПР SolidWorks



Рис. 4.3 – Фотографія розгорнутого діючого прототипу системи автоматизації 3D-друку

					<i>PE22.468332.001 ПЗ</i>	Лис
Зм.	Лис	№ докум.	Підпис	Дата		53

4.2 Аналіз часових характеристик та верифікація циклу автоматичної зміни платформ (APBC)

Критично важливим показником ефективності розробленого комплексу є час, що витрачається на обслуговування верстата між циклами друку. У традиційній схемі при ручному керуванні простій обладнання в нічний час або під час відсутності оператора може становити від кількох годин до кількох діб у неробочі дні.

Експериментально зафіксований повний часовий цикл автоматичного очищення столу та завантаження нової платформи складається з таких фаз:

0–2 с: Завершення нанесення фінального шару моделі, повернення екструдера в крайню ліву позицію.

12-14 с: Плавне підняття інструментальної каретки по осі Z на безпечну висоту та швидкісне переміщення нагрівального столу по осі Y у крайню задню координату (команда повного вильоту столу вперед).

20-25 с: Поетапне скидання поточної пластини та заміна на наступну з «магазину»

5-6 с: Повернення столу в початкові координати та надсилання мікроконтролером ESP32 асинхронного сповіщення оператору в Telegram-чат.

Таким чином, повна тривалість фізичного конвеєрного циклу зміни робочої поверхні становить близько хвилини, що є цілком задовільним показником і зводить технічний простій обладнання до мінімуму.

4.3 Дослідження стабільності багатопотокової архітектури FreeRTOS та мережевої взаємодії

Програмне забезпечення мікроконтролера ESP32 функціонує під керуванням операційної системи реального часу FreeRTOS. Стабільність роботи системи перевірялася в умовах штучно створеного підвищеного мережевого навантаження (одночасне опитування телеметрії, парсинг JSON-буферів розміром до 20 КБ та підтримка сесії Long Polling із таймаутом у 3 секунди).

					<i>PE22.468332.001 ПЗ</i>	Лис
Зм.	Лис	№ докум.	Підпис	Дата		54

Завдяки жорсткому закріпленню задачі TaskMQTT за обчислювальним ядром 1, а задач TaskTelegram та TaskEcoFlow за ядром 0, було досягнуто повної ізоляції критичних потоків. Затримка обробки вхідних топіків звітності report від MQTT-брокера принтера в callback-функції не перевищила 50 мс

Використання методу потокової фільтрації DeserializationOption::Filter() за допомогою бібліотеки ArduinoJson дозволило виокремлювати лише цільові інженерні змінні (температуру сопла nozzle_temper, стан ЧПК gcode_state та відсоток готовності виробу). Експериментальні заміри динамічної пам'яті (heap) показали, що такий підхід знизив пікове споживання оперативної пам'яті SRAM мікроконтролера в три рази (з 18 КБ до 6.2 КБ на один принтер), що повністю нівелює ризик виникнення помилок переповнення пам'яті (Memory Overflow) при безперервній експлуатації комплексу.

					РЕ22.468332.001 ПЗ	Лис
						55
Зм.	Лис	№ докум.	Підпис	Дата		

ВИСНОВКИ

У дипломному проєкті вирішено актуальну науково-технічну задачу комплексної автоматизації процесів безперервного FDM-друку для серійного виготовлення деталей у межах сучасних 3D-ферм та волонтерських ініціатив шляхом розробки автономного IoT-комплексу керування на базі мікроконтролера ESP32.

Проведено огляд існуючих технічних та програмних рішень у галузі адитивного виробництва, який показав відсутність на ринку відкритих та інтегрованих систем автоматизації черг друку для 3D-принтерів споживчого класу, які б також враховували енергетичний аспект. Обґрунтовано необхідність модернізації масового пристрою Bambu Lab A1 з відкритою кінематикою "Bedslinger" для ліквідації простоїв обладнання без втручання в його заводську апаратну частину.

Розроблено механічну підсистему автоматичної зміни платформ (APBC), яка конструктивно складається з передньої частини (кріплень, рухомого та статичного зачіпних гаків) та задньої частини (вертикального касетного «магазину» для зберігання стопки пластин із кріпленням і фіксатором). Усі кастомні конструкційні компоненти успішно спроектовано в САПР SolidWorks та адаптовано під FDM-друк термопластами PLA+ та PETG, що дозволило мінімізувати вартість виготовлення вузла.

Загальна маса надрукованих елементів підсистеми APBC становить 450–500 г при щільності внутрішнього заповнення моделей на рівні 10–15%, що забезпечує достатню жорсткість конструкції при мінімальному навантаженні на механіку верстата. Інтеграція комплексу змінює загальну геометрію робочого простору, додаючи 5 см з переднього боку та 30 см із заднього боку відносно крайніх точок принтера. При цьому збільшення габариту в задній частині на 30 см повністю перекриває та задовольняє стандартну вимогу виробника щодо безпечного технологічного відступу (15–20 см), необхідного для вільного вигину шлейфа живлення нагрівальної платформи та

					<i>PE22.468332.001 ПЗ</i>	Лис
						56
Зм.	Лис	№ докум.	Підпис	Дата		

безперешкодного ходу столу по осі Y, пересування першого безперешкодно забезпечено під «магазинном».

Спроектовано електронну частину та програмне забезпечення центрального інтелектуального шлюзу на базі мікроконтролера ESP32-WROOM-32. Використання багатопотокової архітектури ОС FreeRTOS із жорстким розподілом задач між двома ядрами (ядро 1 — MQTT-телеметрія, ядро 0 — хмарні сервіси) забезпечило затримку обробки топіків принтера не більше 50 мс. Застосування алгоритмів потокової фільтрації JSON-буферів знизило пікове споживання оперативної пам'яті SRAM мікроконтролера в 3 рази (з 18 КБ до 6,2 КБ), забезпечивши стабільність системи у режимі роботи 24/7.

Створено діючий експериментальний зразок комплексу, комплексне тестування якого повністю підтвердило працездатність розробленої структури. Впровадження системи дозволило повністю ліквідувати часові простой обладнання за рахунок автоматизації черги зсуву. Експериментально зафіксовано, що повна тривалість фізичного конвеєрного циклу зміни робочої поверхні становить близько хвилини, а безперервність циклу обмежена виключно кількістю пластин та ємністю накопичувача.

					РЕ22.468332.001 ПЗ	Лис
						57
Зм.	Лис	№ докум.	Підпис	Дата		

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Державний стандарт України Документація. Звіти у сфері науки і техніки. Офіційний вебпортал парламенту України. URL: <https://zakon.rada.gov.ua/rada/show/n0001217-96#Text> (дата звернення: 25.03.2026).
2. Additive manufacturing technologies / I. Gibson et al. SpringerLink. URL: <https://link.springer.com/book/10.1007/978-3-030-56127-7> (date of access: 12.04.2026).
3. BenoitBlanchon. ArduinoJson: Efficient JSON serialization for embedded C++. ArduinoJson. URL: <https://arduinojson.org/> (date of access: 13.05.2026).
4. Bots: an introduction for developers. Telegram APIs. URL: <https://core.telegram.org/bots> (date of access: 04.06.2026).
5. Documentation. Eclipse Mosquitto. URL: <https://mosquitto.org/documentation/> (date of access: 25.03.2026).
6. Ecoflow developer. Ecoflow Developer. URL: <https://developer.ecoflow.com/us> (date of access: 23.05.2026).
7. Espressif Centralized Documentation Platform (CDP). URL: https://documentation.espressif.com/esp32_datasheet_en.pdf (дата звернення: 25.03.2026).
8. FreeRTOS documentation. FreeRTOS. URL: <https://www.freertos.org/Documentation/00-Overview> (date of access: 03.06.2026).
9. How to enable developer mode on bambu lab printers. Bambu Lab Wiki. URL: <https://wiki.bambulab.com/en/knowledge-sharing/enable-developer-mode> (date of access: 08.04.2026).
10. How to enable LAN Mode on Bambu Lab printers. Bambu Lab Wiki. URL: <https://wiki.bambulab.com/en/knowledge-sharing/enable-lan-mode> (date of access: 08.04.2026).

					<i>PE22.468332.001 ПЗ</i>	Лис
						58
Зм.	Лис	№ докум.	Підпис	Дата		

11. Nordquist T. MQTT explorer. MQTT Explorer. URL: <https://mqtt-explorer.com/> (date of access: 05.06.2026).
12. Swapmod kit A1 mini. 3DJake. URL: <https://www.3djake.uk/swap-systems/swapmod-kit> (date of access: 25.03.2026).
13. Telegram bot API. Telegram APIs. URL: <https://core.telegram.org/bots/api> (date of access: 29.05.2026).
14. Vajojajo. Bambulab A1 CAD step model. Maker World. URL: <https://makerworld.com/uk/models/770920-bambulab-a1-cad-step-model#profileId-706764> (date of access: 03.04.2026).

					PE22.468332.001 ПЗ	Лис
						59
Зм.	Лис	№ докум.	Підпис	Дата		

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЄКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	A4	PE22.468332.001 ТЗ	Завдання на дипломний проєкт	6	
2	A4	PE21.468332.001 ПЗ	Пояснювальна записка	59	
3	A4	PE22.468332.001 С1	Схема структурна комплексу автоматизації	1	
4	A4	PE22.468332.003 Е5	Схема функціональна взаємодії задач FreeRTOS	1	
5	A4	PE21.468332.001ПЕ	Схема алгоритму функціонування ПЗ	1	
6	A4	PE22.468332.001...006	Робочі креслення деталей підсистеми APBC	6	
7	A3	PE22.468332.008...010	Робочі креслення деталей підсистеми APBC	4	
8	A4	PE22.468332.001...002	Робочі креслення деталей корпусу ESP32	2	
9	A4	PE22.468332.000	Текст програми (Лістинг коду)	25	

				PE22.468332.001		
	ПІБ	Підп.	Дата			
Розробн.	Кравецька В.М,			Автоматизація 3D Друку	Лист	Листів
Керівн.	Адаменко В.О.				1	1
					КПІ ім. Ігоря Сікорського Каф.ПРЕ, гр. РЕ-22	
Н/контр.						
Зав.каф.	Мовчанюк А.В.					

ДОДАТОК А. ТЕХНІЧНЕ ЗАВДАННЯ

«ЗАТВЕРДЖУЮ»

Зав. катедри

_____ Андрій МОВЧАНЮК

« ____ » _____ 2026 р.

Автоматизація 3D друку

ТЕХНІЧНЕ ЗАВДАННЯ

PE22.468332.001 ТЗ

Київ – 2026 року

1 ТЕХНІЧНЕ ЗАВДАННЯ

1.1 Назва, Підстава для виконання

Система автоматизації 3D друку

Підставою для виконання є завдання кафедри ПРЕ, що затверджено наказом факультету №1089-с від 29.05.2026р.

1.2 Мета виконання і призначення розробки

Метою є проектування та реалізація інтегрованої системи, що складається з механічної та електронної частин, призначеної для автоматизації процесу безперервного 3D-друку та віддаленого контролю кластера принтерів без потреби в ручній модифікації робочих G-code файлів. Особлива увага приділяється забезпеченню низького порогу входження для кінцевого користувача (відсутність потреби у спеціальній технічній освіті) та мінімізації собівартості виробництва.

Призначенням є масштабування адитивного виробництва в режимі автономної роботи «24/7» як для комерційного сектору, так і для волонтерських спільнот (зокрема руху «ДрукАрмія») з метою оптимізації виготовлення великих партій однотипних виробів для оборонних та цивільних потреб.

1.3 Технічні вимоги

1.3.1 Вимоги до функціональності та складу системи

Електронна складова керуючого модуля має бути побудована на базі енергоефективної системи на кристалі (SoC) архітектури ESP32 із вбудованим апаратним радіомодулем Wi-Fi (802.11 b/g/n) та підтримкою операційної системи реального часу FreeRTOS.

Апаратний модуль керування повинен функціонувати повністю автономно та ізольовано від материнської плати 3D-принтера (Bambu Lab A1) для збереження заводської гарантії пристрою на друкарське обладнання та виключення деструктивного втручання в його електричні схеми.

					<i>PE22.468332.001 T3</i>	Лис
						2
Зм.	Лис	№ докум.	Підпис	Дата		

Обмін даними, передача керуючих директив та зчитування телеметрії в реальному часі мають здійснюватися за асинхронним мережевим протоколом MQTT версії 3.1.1 поверх захищеного шару шифрування TLS на стандартному порті 8883 через локальний брокер принтера.

Програмне забезпечення системи повинно автоматично детектувати подію завершення друку поточного виробу, ініціювати виконання модифікованого сервісного коду завершення роботи машини (Machine end G-code) для переміщення столу по осі Y та скидання відпрацьованої гнучкої платформи механічним гаком у приймальний лоток, а також забезпечувати автоматичну подачу наступної чистої магнітної пластини з накопичувача під дією сили тяжіння.

Для захисту навісних механічних елементів автоматики від зіткнень із друкарською головою, програмна логіка стартового G-коду має передбачати примусовий зсув координат ліній очищення сопла з заводських значень у безпечну позитивну зону столу — строго на координати Y=13 та Y=15. Фінішний скрипт повинен утримувати обмотки крокових двигунів під напругою (фіксоване магнітне поле) впродовж усього процесу зміни платформ для збереження валідності просторових координат осей та абсолютної точки відліку (Homing).

Має бути реалізовано віддалений інтерфейс дистанційного керування чергою завдань та онлайн-моніторингу робочих параметрів комплексу (температура сопла, нагрівального столу, поточний відсоток виконання, назва активного файлу) на базі Telegram Bot API. Взаємодія має відбуватися за методом Long Polling без потреби в отриманні статичних публічних IP-адрес («білих» IP), налаштуванні маршрутизації портів (Port Forwarding) чи конфігурації сторонніх VPN-тунелів.

Програмне забезпечення комплексу повинно підтримувати пряму інтеграцію з серверами EcoFlow Developer Portal через HTTP REST API для оперативного контролю параметрів автономного енергопостачання ферми друку (рівень заряду SOC, вхідна та вихідна потужність). Система має автоматично зупиняти чергу друку та переводити принтери в режим паузи при зниженні

					PE22.468332.001 T3	Лис
						3
Зм.	Лис	№ докум.	Підпис	Дата		

заряду станції до критичного порогу (20%) за умови відсутності напруги в промисловій мережі.

1.3.2 Вимоги до конструкції та кінематики

Конструкція накопичувача («магазину») змінних пластин має забезпечувати мінімальну додаткову масу на рухому платформу принтера типу «Bedslinger» (вісь Y) для запобігання виникненню інерційних спотворень геометрії виробів (ефекту «ghosting») та передчасного зносу підшипників і крокових двигунів.

Кути нахилу функціональних пазів, зачепів, технологічних отворів та кріплень у CAD-моделі мають бути оптимізовані для безпідтримочного або легкоочисного виробництва за технологією FDM 3D-друку без погіршення шорсткості.

1.3.3 Вимоги до надійності

Забезпечення безвідмовної механічної роботи вузлів подачі та зачеплення на зсув та стиснення за умов циклічних навантажень.

Автоматичне асинхронне відновлення MQTT-з'єднання з брокерами принтерів та сесії Long Polling з серверами Telegram при короткочасних збоях чи перевантаженнях бездротової мережі Wi-Fi.

Енергонезалежність критичних даних: збереження конфігурації пристроїв, токенів та поточної бази файлів у NVS-розділі флеш-пам'яті за допомогою бібліотеки Preferences для захисту від раптового знеструмлення контролера.

1.4 Вимоги до сировини, матеріалів і ПКВ (покупних комплектуючих виробів)

1.4.1 Вимоги до конструкційних матеріалів (філаментів)

Для друку масштабних нетермонавантажених корпусних деталей рами, тримачів касети та елементів приймального лотка має використовуватися

					PE22.468332.001 T3	Лис
						4
Зм.	Лис	№ докум.	Підпис	Дата		

термопласт, який забезпечує високу жорсткість, мінімальну лінійну усадку (в межах 0.1–0.3%) та високу міжшарову адгезію.

Для виготовлення деталей, що знаходяться у безпосередній близькості до нагрівального стола принтера (елементи фіксації, передній зачіпний гак, нижня опора), обов'язковим є використання матеріалу із температурою деформації не нижче $\sim 75^{\circ}\text{C}$ – 80°C для запобігання термічному розм'якшенню при нагріві платформи до 80°C – 100°C .

1.4.2 Вимоги до ПКВ (компонентної бази)

Мікроконтролер: відлагоджувальна плата типу ESP32 NodeMCU (чи аналог з підтримкою Wi-Fi 802.11 b/g/n).

Робочі платформи: додаткові змінні гнучкі пластини з пружної сталі (Spring Steel Sheets) з PEI-покриттям, сумісні за геометрією з Bambu Lab A1.

Кріпильні елементи: металеві метричні гвинти та гайки типу M3/M4 (або латунні різьбові вставки для вплавлення) для створення розбірних жорстких з'єднань, стійких до вібрацій стола.

1.5 Вимоги до консервації, пакування та маркування

1.5.1 Консервація та захист електроніки

Друкована плата (PCB) керуючого електронного модуля після завершення монтажу компонентів має бути покрита захисним ізоляційним акриловим або силіконовим лаком для запобігання окисленню контактних майданчиків та доріжок під дією вологи.

Електронні компоненти комплексу (плата розширення, МК ESP32) повинні бути повністю захищені від виробничого пилу, статичної електрики (ESD-захист) та випадкових механічних пошкоджень шляхом розміщення всередині захисного двокомпонентного пластикового корпусу з інтегрованою пасивною вентиляційною перфорацією для оптимізації конвекційного охолодження SoC процесора.

					PE22.468332.001 T3	Лис
						5
Зм.	Лис	№ докум.	Підпис	Дата		

1.5.2 Пакування

Транспортування елементів системи у розібраному стані має здійснюватися в ударостійкій тарі (картонні коробки з ложементами зі спіненого поліетилену) для захисту пластикових FDM-вузлів від сколювання вздовж ліній міжшарового зчеплення.

1.5.3 Маркування

Маркування отвору апаратного скидання має містити міжнародний ідентифікатор «RESET», а вікно порту Type-C – індикацію параметрів живлення постійного струму «5V = 1A». Маркування виконується методом втисненогодебосингу на глибину 0.2 мм.

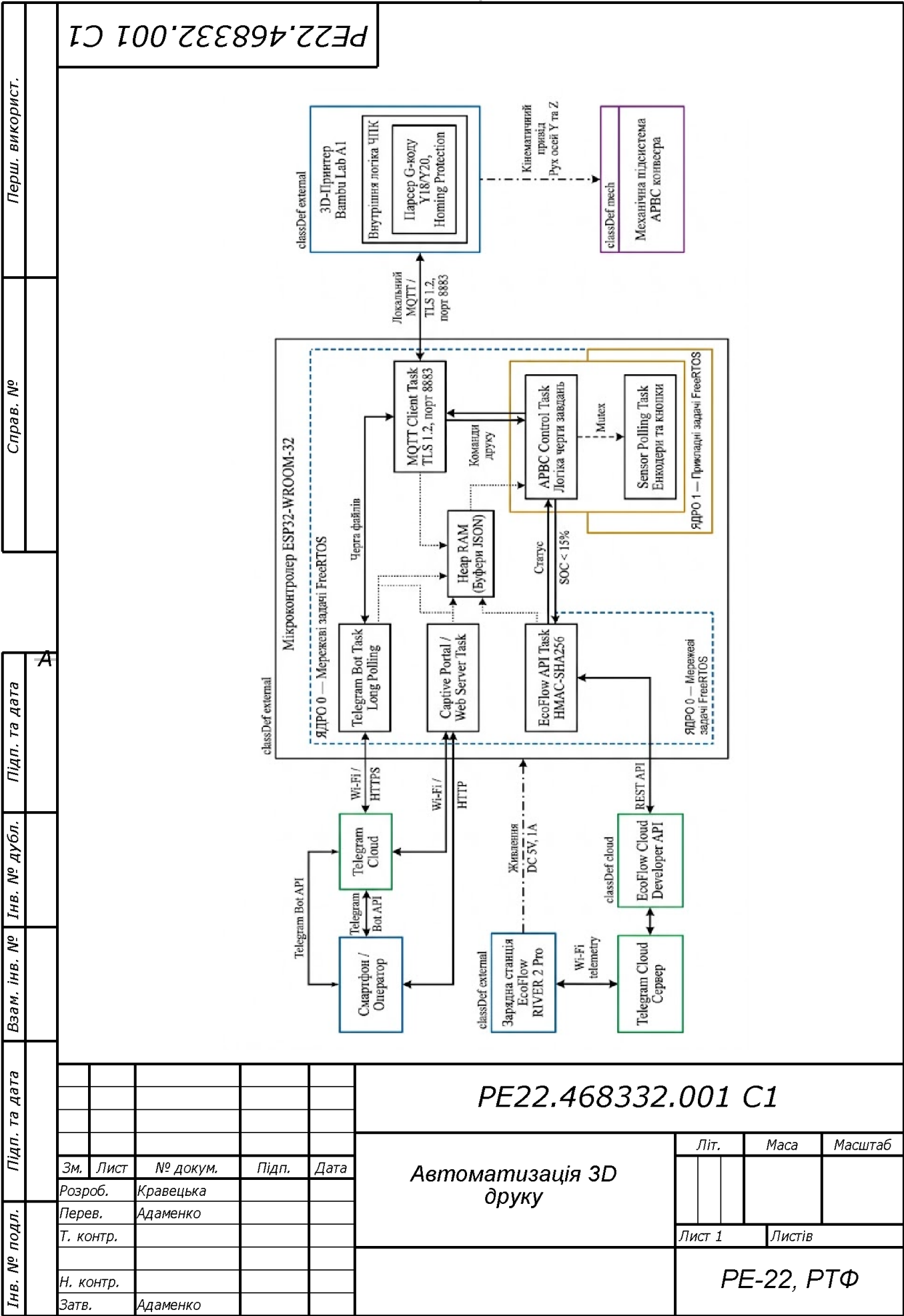
1.6 Вимоги до розроблюваної документації

Комплект конструкторської, технологічної та програмної документації бакалаврського дипломного проєкту має бути виконаний у суворій відповідності до вимог стандартів Єдиної системи конструкторської документації (ЕСКД) та чинних нормативних інструкцій ДСТУ. Повний вихідний код програми керування мікроконтролером має бути оформлений моноширинним шрифтом у вигляді обов'язкового додатка до пояснювальної записки.

					PE22.468332.001 T3	Лис
						6
Зм.	Лис	№ докум.	Підпис	Дата		

ДОДАТОК Б. СХЕМА СТРУКТУРНА КОМПЛЕКСУ
АВТОМАТИЗАЦІЇ

1

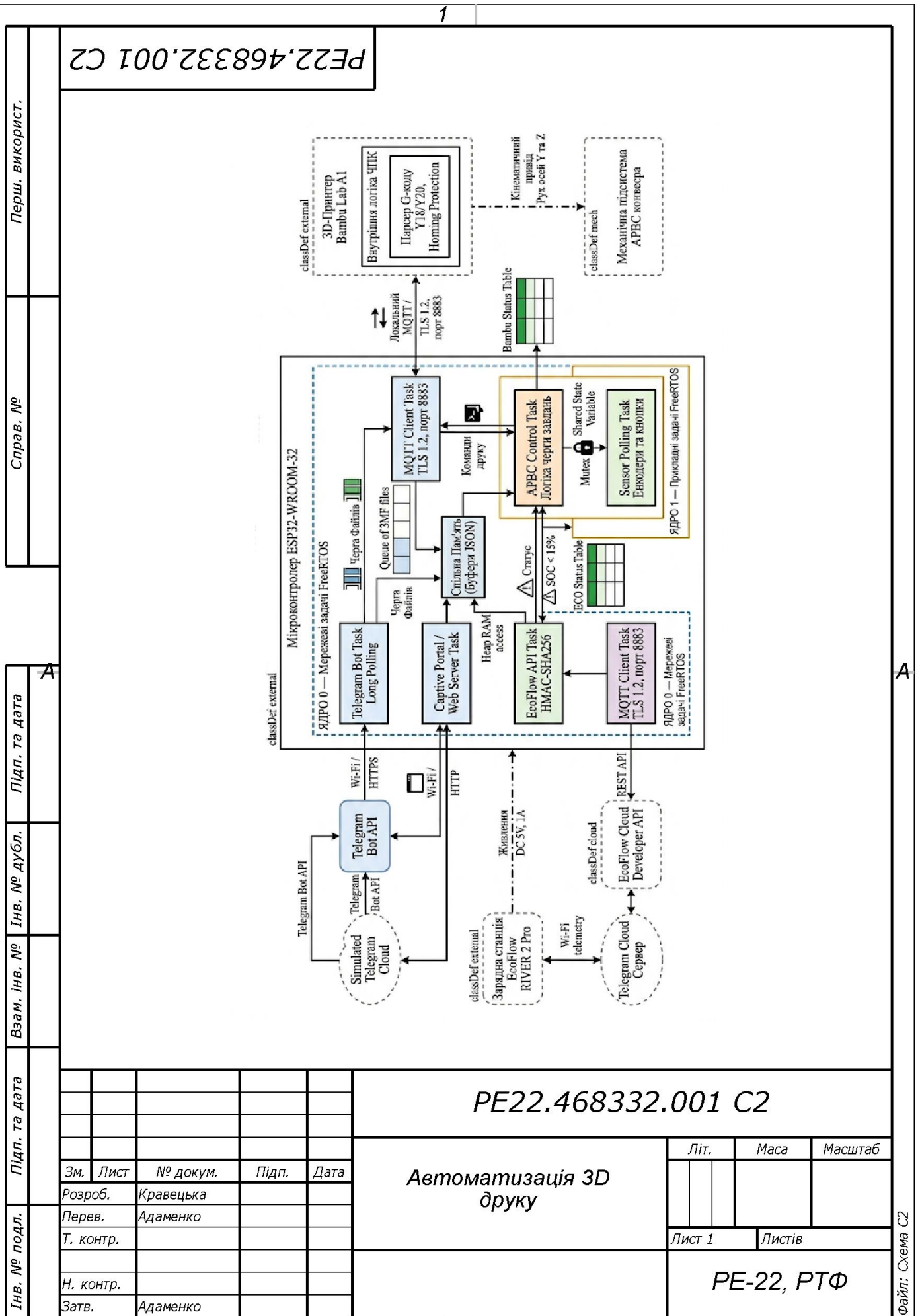


1

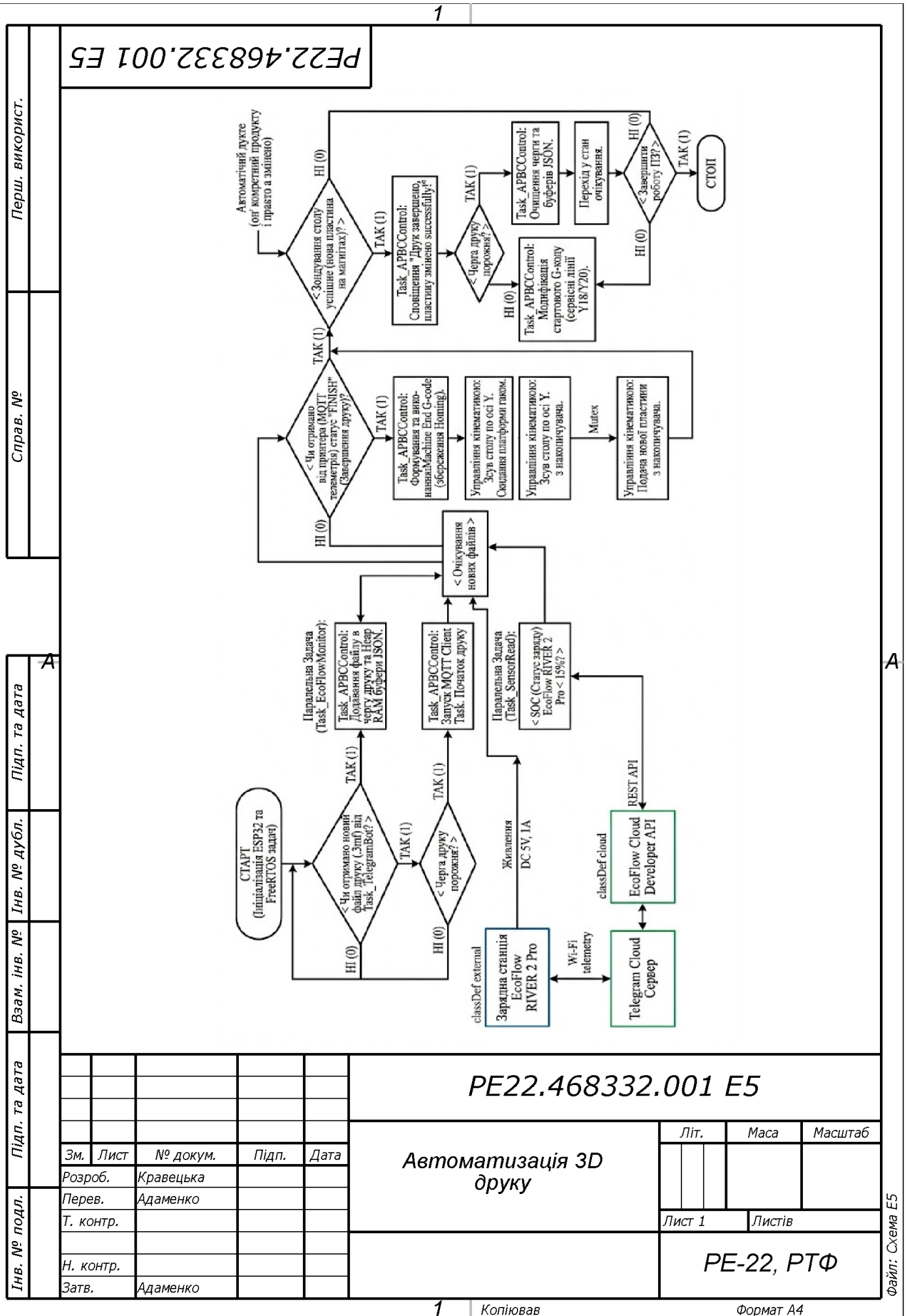
Копіював

Формат A4

ДОДАТОК В. СХЕМА ФУНКЦІОНАЛЬНА ВЗАЄМОДІЇ ЗАДАЧ FREERTOS



ДОДАТОК Г. СХЕМА АЛГОРИТМУ ФУНКЦІОНУВАННЯ ПЗ



Перш. використ.		Справ. №		РЕ22.468332.001	
Підп. та дата		Інв. № дубл.		Взам. інв. №	
Підп. та дата		Інв. № подл.		РЕ22.468332.001	
Зм.		Лист		№ докум.	
Розроб.		Кравецька		Підп.	
Перев.		Адаменко		Дата	
Т. контр.					
Н. контр.					
Затв.		Адаменко			
Тримач для пластини		Літ.		Маса	
		Лист 1		Листів	
		РЕ-22, РТФ			

1

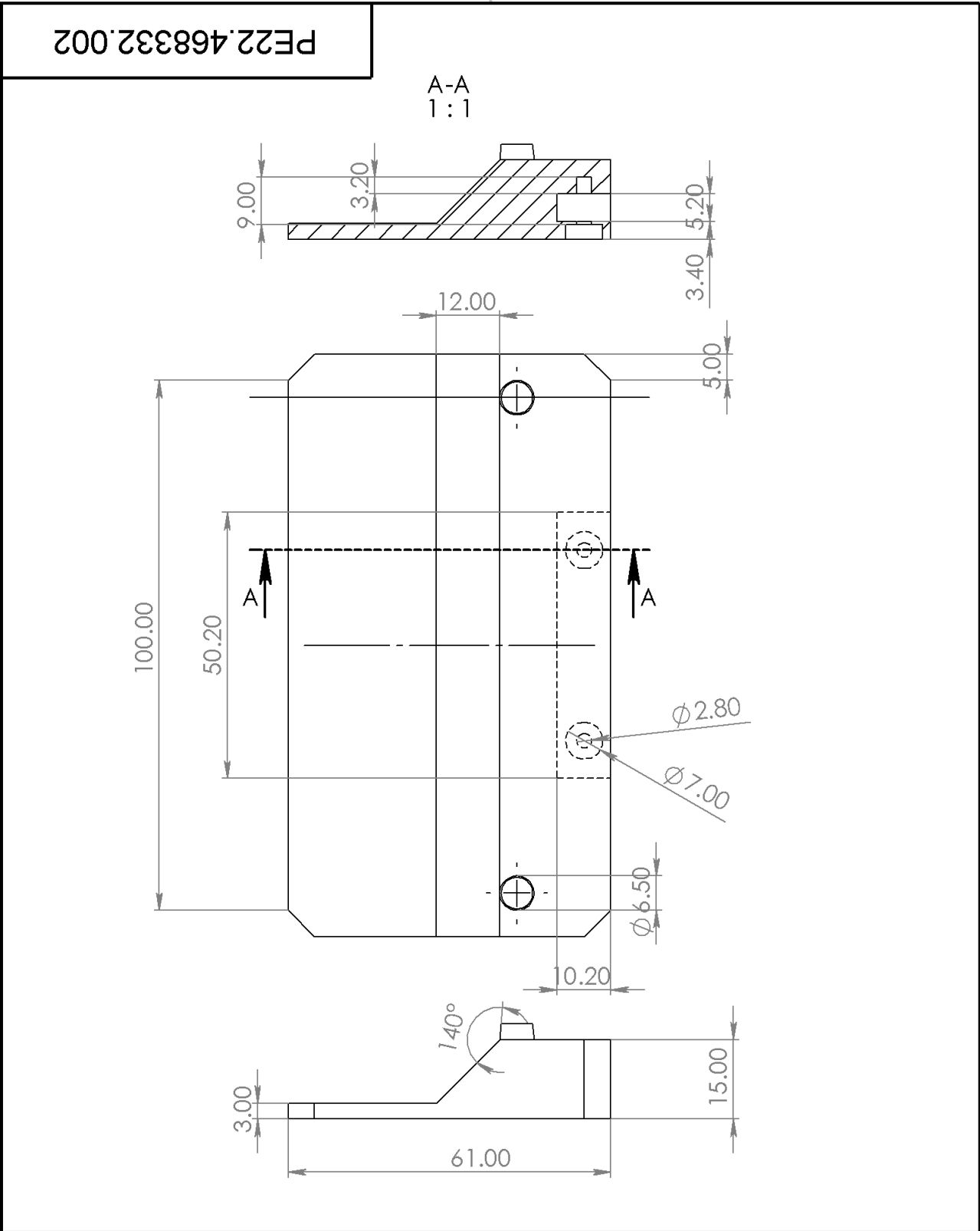
Копіював

Формат А4

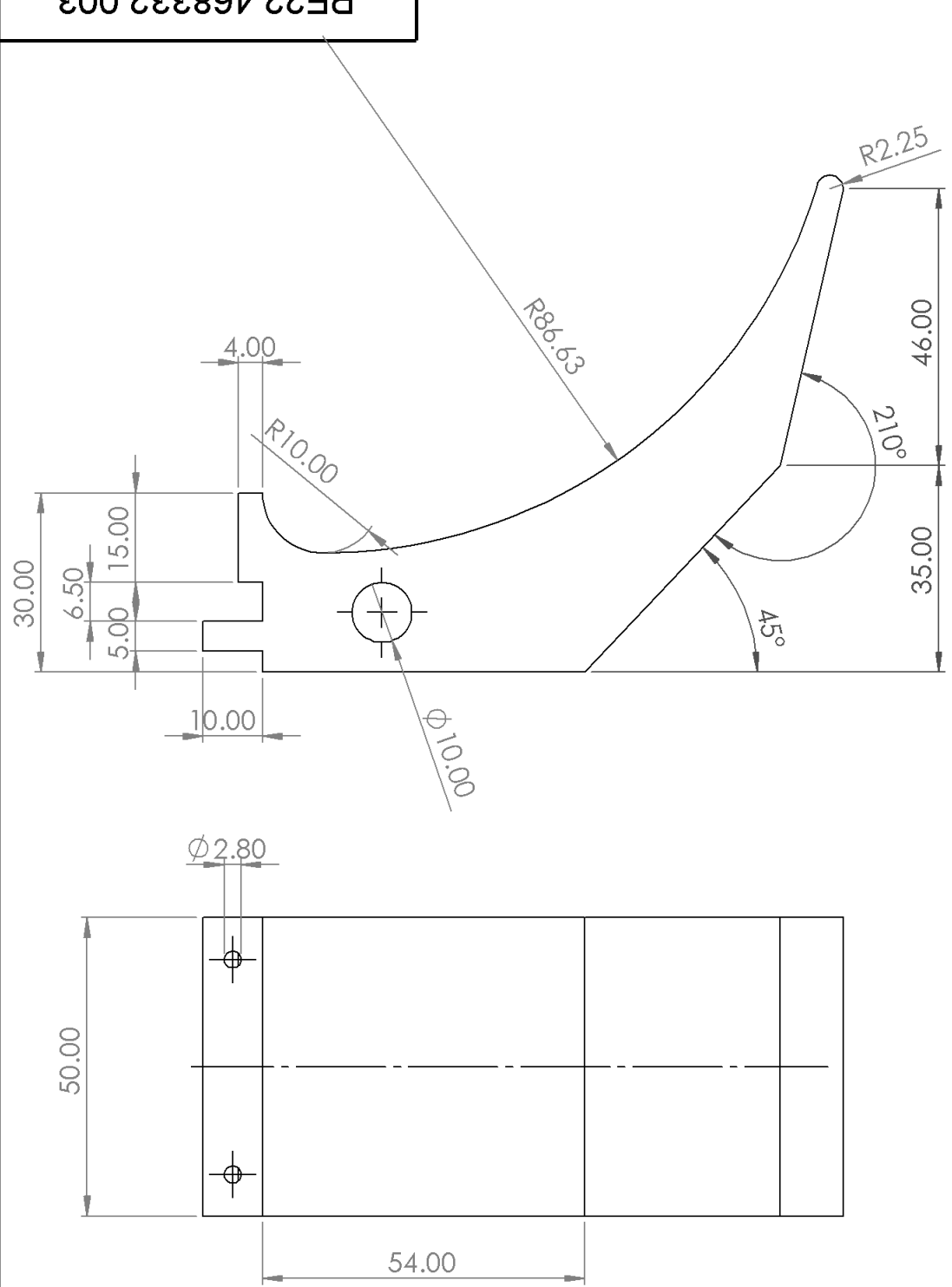
Файл: Для пластини

Перш. використ.	
Справ. №	

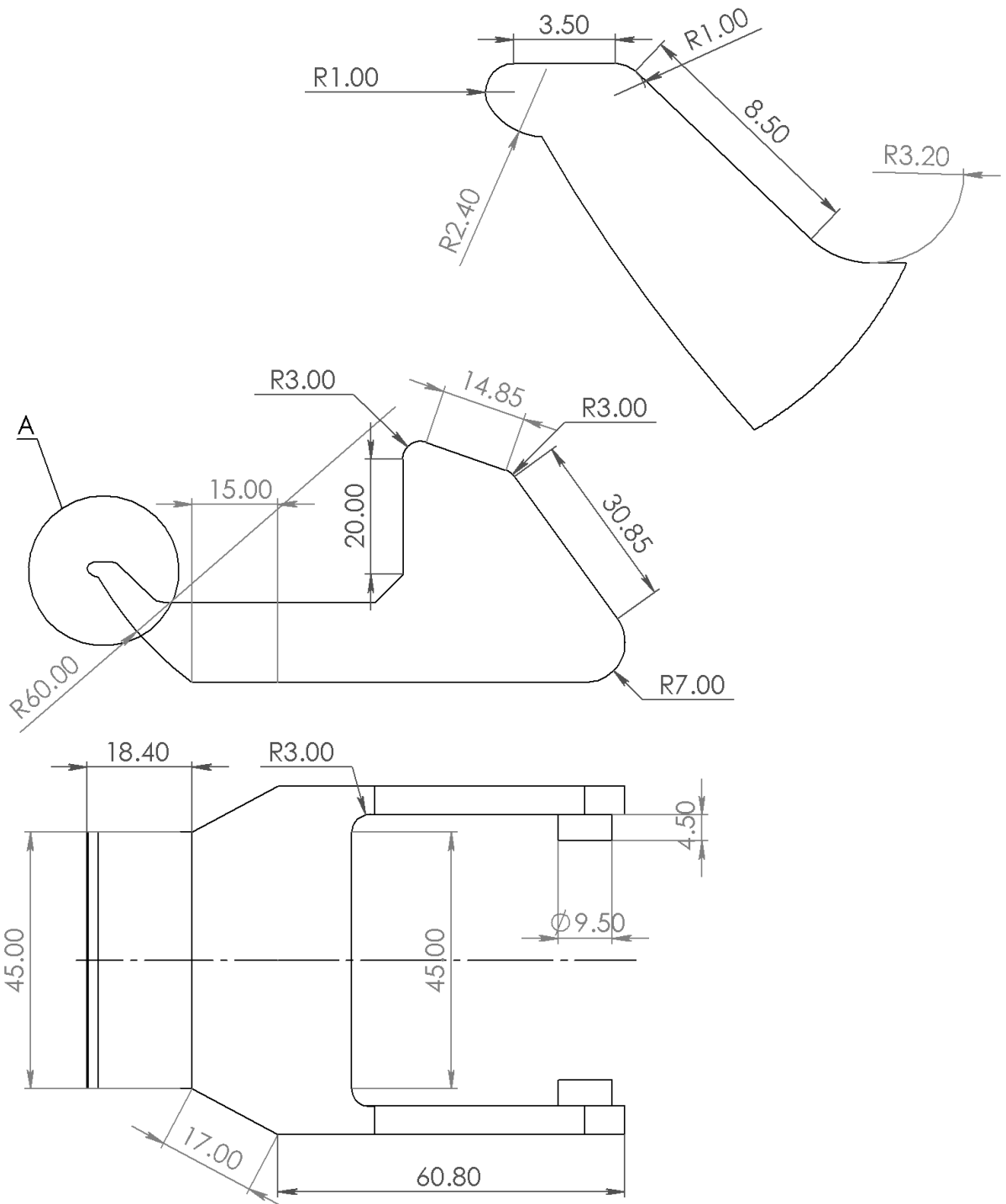
Підп. та дата	
Інв. № дубл.	
Взам. інв. №	
Підп. та дата	
Інв. № подл.	



PE22.468332.002				
Зм.	Лист	№ докум.	Підп.	Дата
Розроб.		Кравецька		
Перев.		Адаменко		
Т. контр.				
Н. контр.				
Затв.		Адаменко		
Передній фіксатор				
Літ.		Маса	Масштаб	
Лист 2		Листів		
PE-22, РТФ				

Перш. використ.		Справ. №		Підп. та дата		Інв. № дубл.		Взам. інв. №		Підп. та дата		Інв. № подл.							
PE22.468332.003												PE22.468332.003							
Зм.		Лист		№ докум.		Підп.		Дата		<i>Передній великий гак</i>		Літ.		Маса		Масштаб			
Розроб.		Кравецька																	
Перев.		Адаменко																	
Т. контр.																			
Н. контр.																			
Затв.		Адаменко										Лист 3		Листів					
										PE-22, РТФ									

PE22.468332.004

A
5:1

A

PE22.468332.004

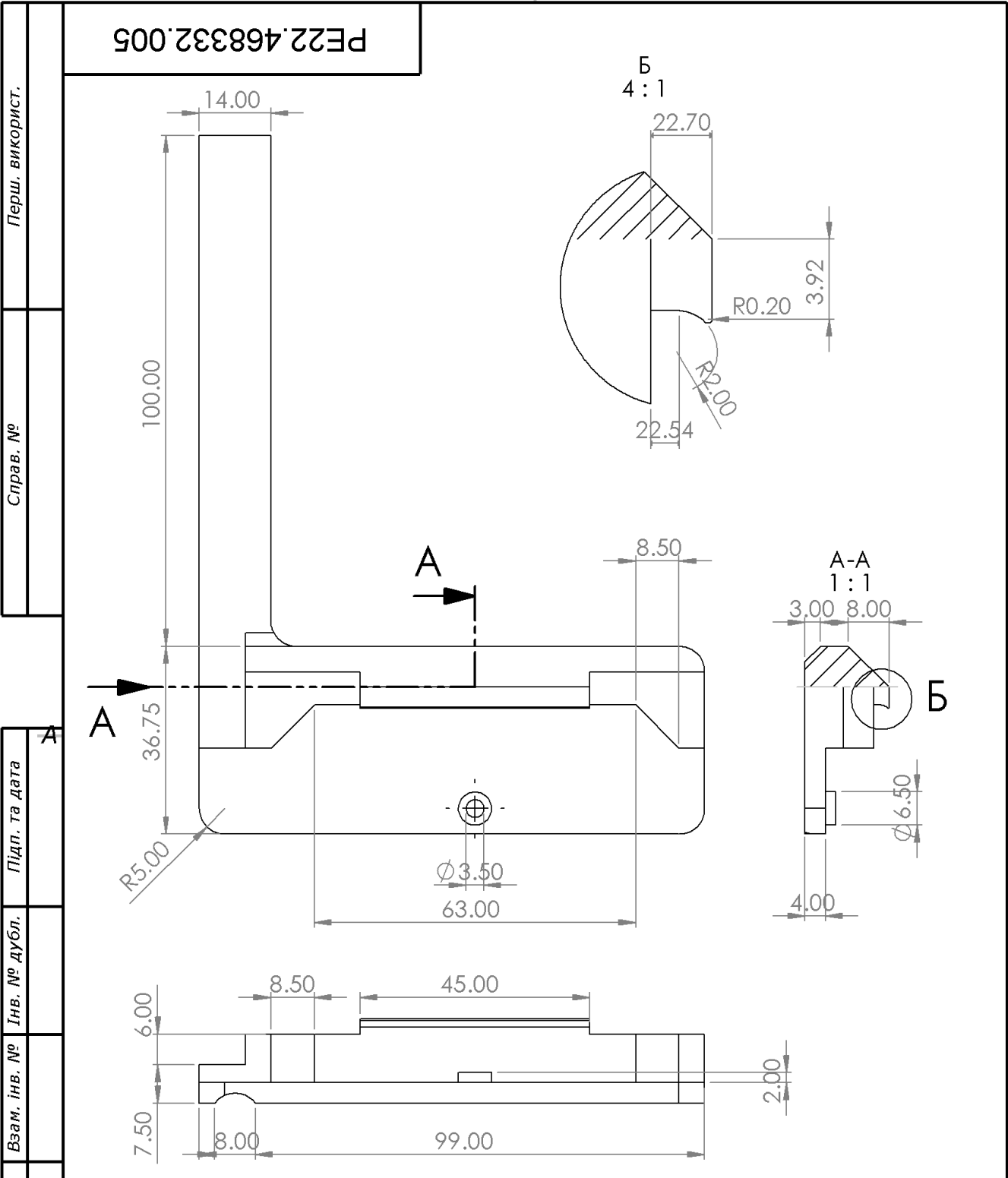
Зм.	Лист	№ докум.	Підп.	Дата
Розроб.		Кравецька		
Перев.		Адаменко		
Т. контр.				
Н. контр.				
Затв.		Адаменко		

Передній малий зак

Літ.	Маса	Масштаб
Лист 4	Листів	

PE-22, РТФ

Файл: Малий зак



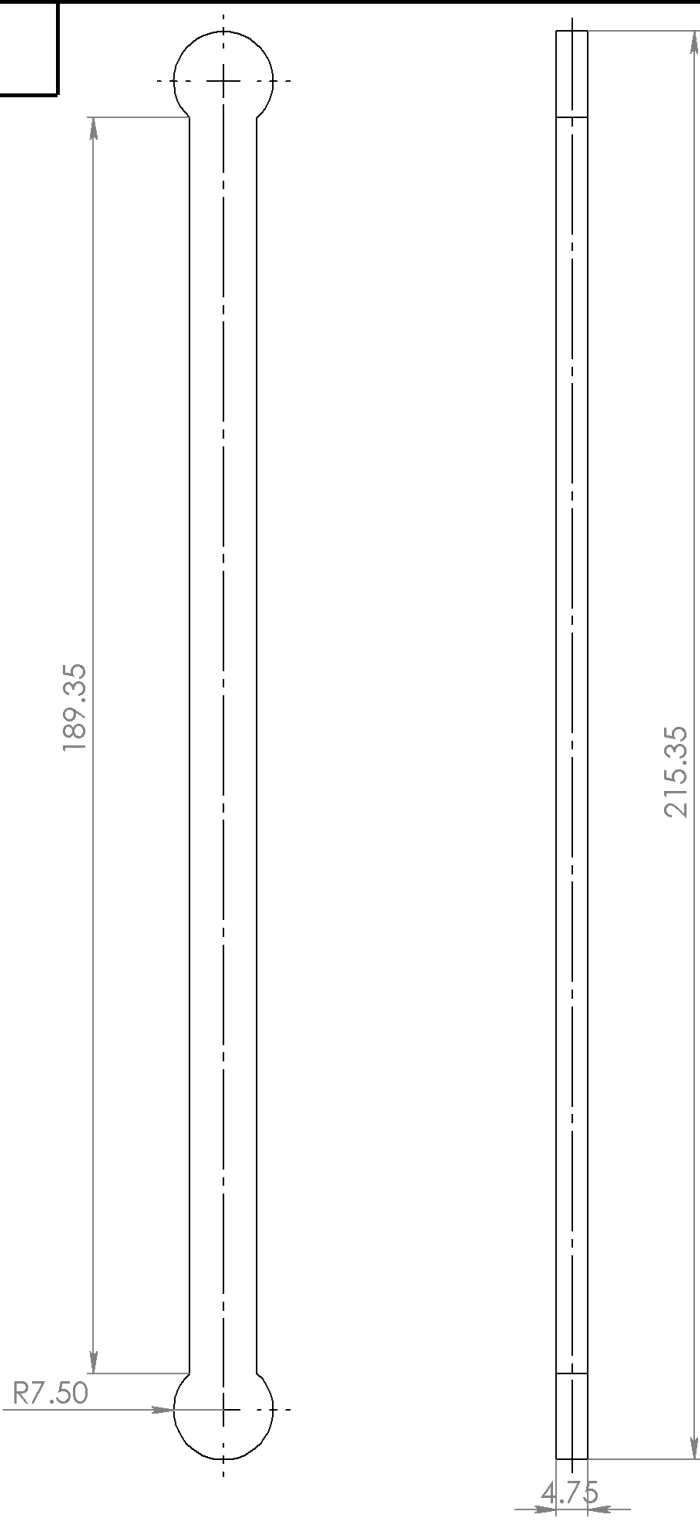
Підп. та дата						PE22.468332.005					
						<i>Заднє кріплення для стола</i>					
		Зм.	Лист	№ докум.	Підп.	Дата	Літ.			Маса	Масштаб
Інв. № подл.		Розроб.	Кравецька								
		Перев.	Адаменко								
		Т. контр.				Лист 5			Листів		
						PE-22, РТФ					
		Н. контр.									
	Затв.	Адаменко									

Інв. № подл.	Підп. та дата	Взам. інв. №	Інв. № дубл.	Підп. та дата	Справ. №	Перш. використ.
--------------	---------------	--------------	--------------	---------------	----------	-----------------

Перш. використ.	
Справ. №	

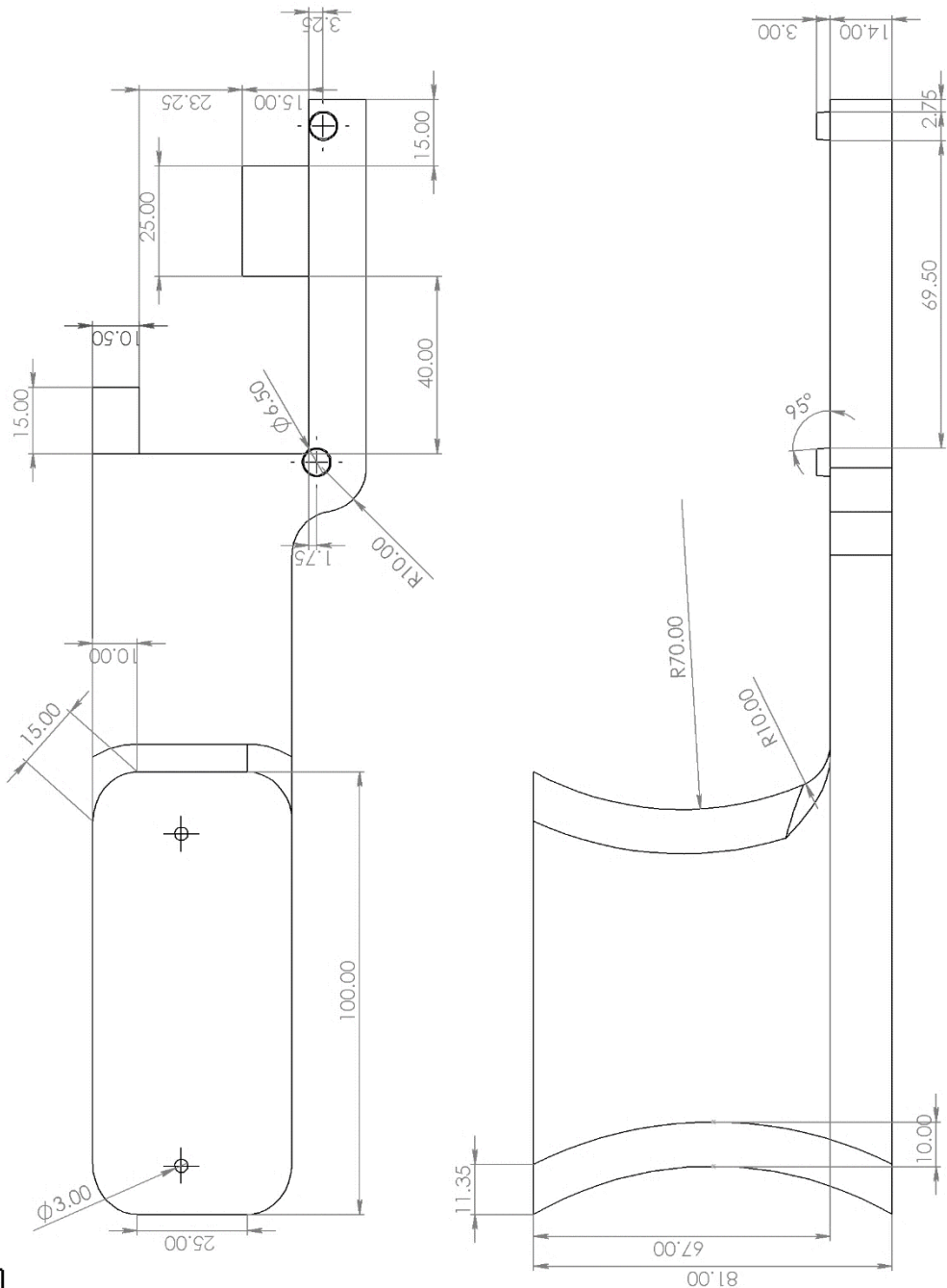
Підп. та дата	А
Інв. № дубл.	
Взам. інв. №	
Підп. та дата	
Інв. № подл.	

PE22.468332.006



Зм.	Лист	№ докум.	Підп.	Дата
Розроб.	Кравецька			
Перев.	Адаменко			
Т. контр.				
Н. контр.				
Затв.	Адаменко			

PE22.468332.006		
Фіксатор задніх опор		
Літ.	Маса	Масштаб
Лист 6		Листів
PE-22, РТФ		

[illegible]

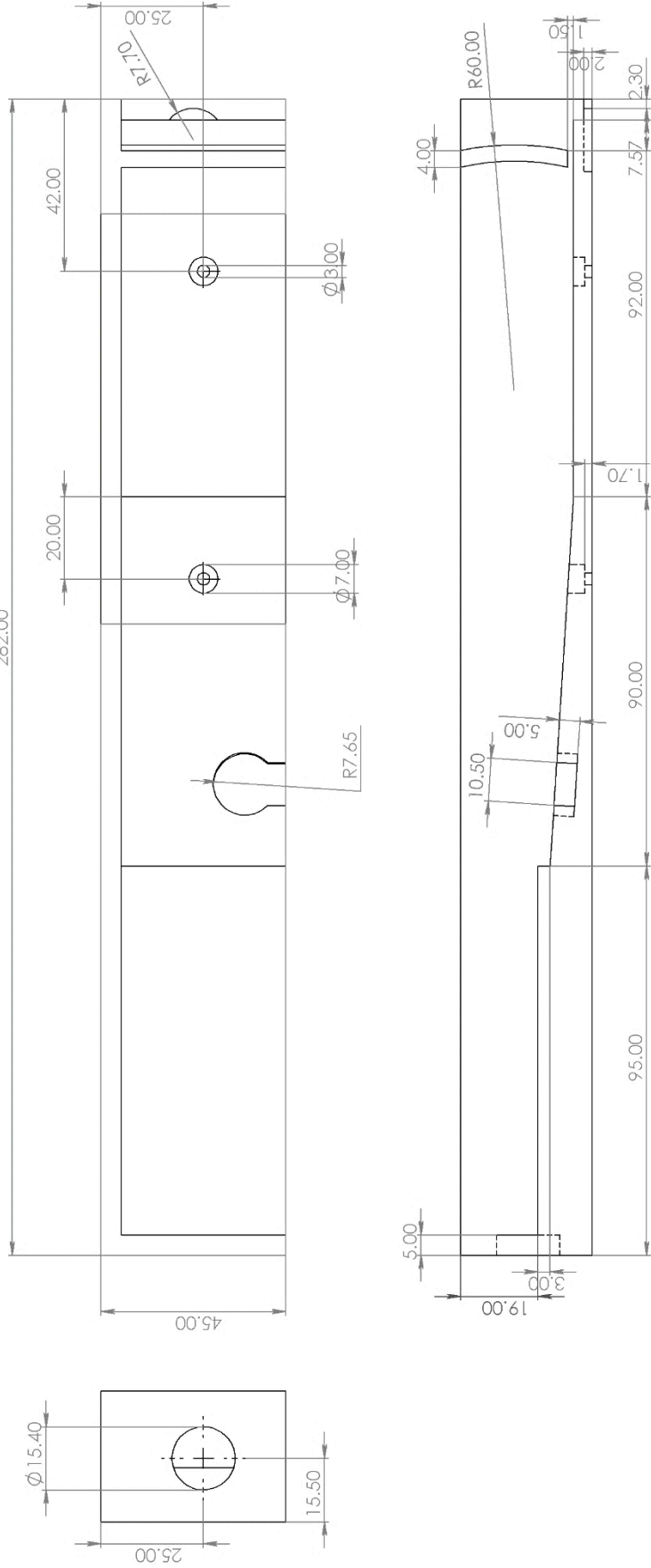
PE22.468332.008

PE22.468332.009

2

1

ІНВ. № подл.	Підп. та дата	Взам. ІНВ. №	ІНВ. № дубл.	Підп. та дата	Справ. №	Перш. викопист.
--------------	---------------	--------------	--------------	---------------	----------	-----------------



Зм.	Лист	№ док-м.	Підп.	Дата	Літ.	Маса	Масштаб
Розроб.	Перев.	Кравецька	Адаменко				
Т. контр.	Н. контр.	Затв.	Адаменко				
Права частина касетного модуля					Лист 8	Листів	
РЕ-22, РТФ							

РЕ22.468332.009

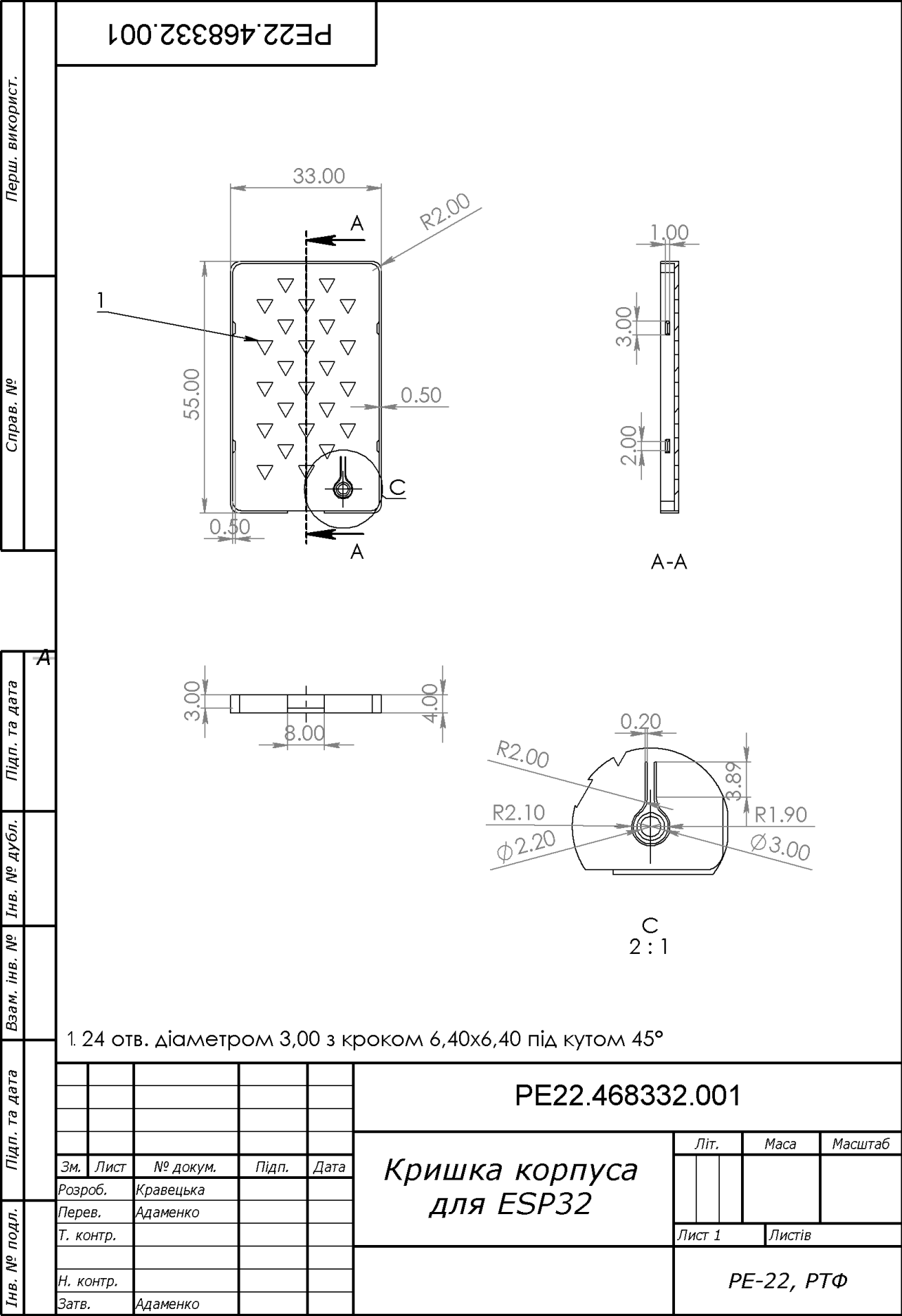
2

1

Формат А3

Файл: Задня верхня права опора

ДОДАТОК Е. РОБОЧІ КРЕСЛЕННЯ ДЕТАЛЕЙ (КОРПУС)



ДОДАТОК Ж. ЛІСТИНГ КОДУ

```
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
#include <WebServer.h>
#include <Preferences.h>
#include <HTTPClient.h>
#include <DNSServer.h>
#include <ESPmDNS.h>
#include <mbedtls/md.h>
#include <time.h>
#include <sys/time.h>
#include <pgmspace.h>

//Токен Telegram Bot (змінює розробник, вшивається у прошивку)
#define TG_BOT_TOKEN "8788189691:AAFSCbqgdQ3EkLOWWbHUGozYp4MKNwf1EqA"

//Константи
#define MAX_PRINTERS 4
#define MAX_FILES 20
#define QUEUE_MAX 30
#define WAIT_BETWEEN_MS 12000UL
#define PUSHALL_INTERVAL 60000UL
#define ECOFLOW_INTERVAL 60000UL
#define AP_SSID "BambuFarm-Setup"
#define AP_PASS "12345678"

//Структури
struct PrinterConfig {
    String name;
    String ip;
    String password;
    String serial;
};

struct PrinterState {
    float nozzle = 0;
    float bed = 0;
    int percent = 0;
    int remainMin = 0;
    int layerNum = 0;
    int totalLayer = 0;
    int speedLevel = 2;
    String state = "OFFLINE";
    String prevState = ""; //v2.8: порожній при старті, не "OFFLINE"
    String file = "-";
    bool active = false;
    bool mqttReady = false; //v2.8: true після першого успішного pushall
};

//Глобальні об'єкти
Preferences prefs;
WebServer server(80);
DNSServer dns;

PrinterConfig printerCfg[MAX_PRINTERS];
PrinterState printerSt[MAX_PRINTERS];
int printerCount = 0;

WiFiClientSecure* espClients[MAX_PRINTERS];
PubSubClient* mqttClients[MAX_PRINTERS];
```

					PE22.468332.001	Лис
						1
Зм.	Лис	№ докум.	Підпис	Дата		

```

//Telegram: один persistent клієнт
WiFiClientSecure tgClient;
SemaphoreHandle_t tgMutex = NULL;

//Telegram: черга в боті
#define TG_QUEUE_MAX 30
struct TgBotQueue {
    String items[TG_QUEUE_MAX];
    int total = 0;
    int printerIdx = 0; //вибраний принтер для виконання черги
};
static TgBotQueue tgBotQueue;

//EcoFlow
struct {
    String host = "https://api-e.ecoflow.com";
    String sn;
    String appKey;
    String secret;
    int soc = 100;
    int remainMin = 0;
    int inPower = 0;
    int outPower = 0;
    bool critical = false;
    int pauseAt = 20;
    int prevInPower = -1;
} eco;

//Черга друку (веб)
struct {
    String items[QUEUE_MAX];
    int total = 0;
    int index = 0;
    bool active = false;
    bool waiting = false;
    unsigned long waitStart = 0;
    int printerIdx = 0;
} queue;

//Telegram globals
const String tgToken = TG_BOT_TOKEN;
String tgPassword = "";
String tgChatId = "";
long long lastUpdateId = 0;

unsigned long lastPushallMs[MAX_PRINTERS] = {0};
unsigned long connectedAtMs[MAX_PRINTERS] = {0};
bool pushallPending[MAX_PRINTERS] = {false};

String sdFiles[MAX_FILES];
int sdFileCount = 0;
bool setupMode = false;

//Прототипи
void loadConfig();
void saveConfig();
void loadFiles();
void saveFiles();
void startNormalMode();
void startSetupMode();
void mqttReconnect(int idx);
void mqttCallback(char* topic, byte* payload, unsigned int length);
void startPrint(int printerIdx, const String& filename);
void changeSpeed(int printerIdx, int level);

```

					PE22.468332.001	Лис
						2
Зм.	Лис	№ докум.	Підпис	Дата		

```

void pausePrint(int printerIdx);
void resumePrint(int printerIdx);
void processNextInQueue();
String ecoflowSign(const String& params, const String& secret);
void sendTelegramMessage(const String& text, const String& replyMarkup = "");
void handleTelegramUpdates();
void TaskEcoFlow(void* pv);
void TaskMQTT(void* pv);
void TaskTelegram(void* pv);
String buildSetupHTML();
bool isTgAuthorized(const String& chatId);
String buildMainMenu();
String fmtTime(int minutes);
String speedName(int lvl);

//HTML y Flash
const char mainHTML_P[] PROGMEM = R"html(
<!DOCTYPE html><html lang='uk'><head>
<meta charset='UTF-8'><meta name='viewport' content='width=device-
width,initial-scale=1'>
<title>Bambu Farm</title>
<style>
    :root{--bg:#0a0e14;--s:#111720;--b:#1e2d40;--ac:#00c8ff;--og:#ff6b35;--
gn:#3ddc84;--tx:#d0dce8;--mu:#4a6080;--yl:#f5d033;}
    *{box-sizing:border-box;margin:0;padding:0}
    body{background:var(--bg);color:var(--tx);font-family:'Courier
New',monospace;font-size:16px;}
    .hdr{background:var(--s);border-bottom:1px solid var(--b);padding:14px
20px;display:flex;flex-wrap:wrap;gap:10px;align-items:center}
    .logo{color:var(--ac);font-size:20px;font-weight:bold;letter-
spacing:3px;margin-right:10px}
    .st{background:var(--bg);border:1px solid var(--b);border-
radius:4px;padding:8px 12px;font-size:15px;white-space:nowrap}
    .st b{color:var(--yl)} .hdr-right{margin-left:auto;display:flex;gap:8px}
    .main-wrap{display:flex;gap:20px;padding:20px;align-items:flex-start;max-
width:1600px;margin:0 auto}
    .left-col{flex:1}.right-col{width:380px;flex-shrink:0;display:flex;flex-
direction:column;gap:20px}
    .grid{display:grid;grid-template-columns:1fr 1fr;gap:20px}
    @media (max-width:1100px){.grid{grid-template-columns:1fr}}
    @media (max-width:800px){.main-wrap{flex-direction:column}.right-
col{width:100%}}
    .card{background:var(--s);border:1px solid var(--b);border-
radius:8px;padding:20px}
    .ctitle{font-size:15px;color:var(--ac);letter-spacing:2px;text-
transform:uppercase;border-bottom:1px solid var(--b);padding-
bottom:10px;margin-bottom:14px;display:flex;justify-content:space-
between;align-items:center}
    .prow{display:flex;justify-content:space-between;font-size:16px;margin:8px
0;align-items:center}
    .prow span{color:var(--mu)}.prow b{color:var(--tx)}
    .progbar{background:var(--bg);border-radius:4px;height:8px;margin:12px
0;overflow:hidden}
    .progfill{height:100%;background:var(--ac);transition:width .5s}
    .fi{background:var(--bg);border:1px solid var(--b);border-
radius:4px;padding:10px 12px;margin:6px 0;display:flex;justify-content:space-
between;align-items:center;font-size:16px;gap:8px}
    .fn{overflow:hidden;text-overflow:ellipsis;white-space:nowrap;flex:1}
    .qi{padding:8px 0;border-bottom:1px solid var(--b);font-
size:16px;display:flex;gap:8px;align-items:center}
    .qn{color:var(--mu);min-width:24px}
    button,select.speed-sel,input.pass-input{font-family:'Courier
New',monospace;border:none;border-radius:4px;font-
weight:bold;transition:opacity .15s;padding:8px 14px;font-size:15px}
    button{cursor:pointer}button:hover{opacity:.75}

```

					PE22.468332.001	Лис
						3
Зм.	Лис	№ докум.	Підпис	Дата		

```

    .btn-g{background:#1a5e30;color:var(--gn)}.btn-
r{background:#5e1a1a;color:#ff6b6b}
    .btn-b{background:#1a3a5e;color:var(--ac)}
    .btn-run{background:var(--ac);color:#000;width:100%;padding:14px;font-
size:17px;letter-spacing:1px;margin-top:10px}
    .row{display:flex;gap:8px;flex-wrap:wrap;margin-top:10px}
    input.pass-input{background:var(--bg);border:1px solid var(--b);color:var(-
-tx);max-width:200px;padding:6px;font-size:13px}
    select.speed-sel{background:var(--bg);border:1px solid var(--b);color:var(-
-yl);padding:4px 8px;font-size:14px;cursor:pointer}
    select.form-input{width:100%;background:var(--bg);border:1px solid var(--
b);color:var(--tx);padding:12px;border-radius:4px;font-family:inherit;font-
size:16px;margin-bottom:14px;cursor:pointer}
    .badge{display:inline-block;padding:3px 10px;border-radius:10px;font-
size:13px;font-weight:bold}
    .b-idle{background:#1a2a1a;color:var(--mu)}.b-
run{background:#1a3a1a;color:var(--gn)}
    .b-fin{background:#1a2a3a;color:var(--ac)}.b-
pau{background:#3a2a1a;color:var(--og)}
    .b-err{background:#3a1a1a;color:#ff6b6b}.b-off{background:#333;color:#888}
    .crit{border-color:#ff6b6b!important;animation:blink 1s infinite}
    @keyframes blink{0%,100%{opacity:1}50%{opacity:.5}}
    .modal{display:none;position:fixed;inset:0;background:#0009;z-
index:100;justify-content:center;align-items:center}
    .modal.open{display:flex}
    .mbox{background:var(--s);border:1px solid var(--b);border-
radius:10px;padding:26px;width:min(95vw,460px);max-height:90vh;overflow-
y:auto}
    .mbox h3{color:var(--ac);letter-spacing:2px;font-size:16px;margin-
bottom:18px}
    label{font-size:14px;color:var(--mu);display:block;margin-bottom:6px}
    .mbox input{width:100%;background:var(--bg);border:1px solid var(--
b);color:var(--tx);padding:12px;border-radius:4px;font-family:inherit;font-
size:16px;margin-bottom:14px}
    .hint{font-size:13px;color:var(--og);margin-bottom:14px;line-height:1.4}
    .tg-hint{font-size:12px;color:var(--mu);margin-top:4px;line-height:1.4}
    .add-row{display:flex;gap:8px;margin-bottom:8px}
    .add-row input{margin:0;flex:1}
    .tg-section{display:flex;flex-
direction:column;gap:4px;background:transparent;border:none}
    .tg-row{display:flex;gap:6px;align-items:center}
</style></head><body>

<div class='hdr' id='hdrEco'>
  <div class='logo'>&#9632; BAMBU FARM</div>
  <div class='st' id='stEco'>&#9889; ...</div>
  <div class='st' id='stTime'>&#9201; ...</div>
  <div class='st' id='stPow'>&#11015; ...</div>

  <div class='tg-section'>
    <div class='tg-row'>
      <span style='color:var(--mu);font-size:13px'>🔑</span>
      <input type='password' id='tgPassField' class='pass-input'
placeholder='Код доступу TG' style='max-width:140px'>
      <button class='btn-g' style='padding:6px 10px;font-size:12px'
onclick='saveTgPass()'>✓</button>
    </div>
    <div class='tg-hint'>Придумайте код → введіть тут → надішліть боту <a
href='https://t.me/bambu_farm_bot' target='_blank' style='color:var(--
ac)'>@bambu_farm_bot</a></div>
  </div>

  <div class='hdr-right'>
    <button class='btn-b' onclick='openModal("mEco")'>&#9965;
EcoFlow</button>

```

					PE22.468332.001	Лис
						4
Зм.	Лис	№ докум.	Підпис	Дата		

```

        <button class='btn-g' onclick='openModal("mAdd")'>+ Пристрій</button>
    </div>
</div>

<div class='main-wrap'>
    <div class='left-col'>
        <div class='grid' id='printers'></div>
    </div>
    <div class='right-col'>
        <div class='card'>
            <div class='ctitle'>&#128193; БАЗА ФАЙЛІВ</div>
            <div class='add-row'>
                <input type='text' id='newFile' placeholder='ForPlate.gcode.3mf'
style='background:var(--bg);border:1px solid var(--b);color:var(--
tx);padding:8px;border-radius:4px;font-family:inherit'>
                <button class='btn-g' onclick='addFile()'>Додати</button>
            </div>
            <div class='hint'>&#9888; Вводьте назву латиницею</div>
            <div id='fileList' style='margin-top:10px'>Завантаження...</div>
        </div>
        <div class='card'>
            <div class='ctitle'>&#128203; ЧЕРГА ДРУКУ</div>
            <label>Принтер для виконання черги</label>
            <select id='queuePrinter' class='form-input'></select>
            <div id='queueList' style='margin-top:10px;margin-bottom:15px'><p
style='color:var(--mu)'>Порожньо</p></div>
            <div class='row'>
                <button class='btn-r' onclick='clearQueue()'>&#128465;
Очистити</button>
            </div>
            <button class='btn-run' onclick='sendQueue()'>&#128640; ЗАПУСТИТИ
ЧЕРГУ</button>
        </div>
    </div>
</div>

<!-- Модалка: Додати пристрій -->
<div class='modal' id='mAdd'>
<div class='mbox'>
    <h3>+ ДОДАТИ ПРИСТРІЙ</h3>
    <label>Тип пристрою</label>
    <select id='devType' onchange='toggleDevFields()' class='form-input'>
        <option value='printer'>Принтер Bambu Lab</option>
        <option value='ecoflow'>EcoFlow (зарядна станція)</option>
    </select>
    <div id='printerFields'>
        <label>Назва (довільна)</label><input type='text' id='pName'
placeholder='Принтер 1'>
        <label>IP-адреса</label><input type='text' id='pIp'
placeholder='192.168.0.100'>
        <label>Access Code</label><input type='password' id='pPass'
placeholder='12345678'>
        <label>Serial Number</label><input type='text' id='pSn'
placeholder='03919D531406447'>
    </div>
    <div id='ecoFields' style='display:none'>
        <label>Serial Number (SN)</label><input type='text' id='eSn'
placeholder='R621ZEB6XG2M0913'>
        <label>App Key</label><input type='text' id='eKey'
placeholder='TPuC0c...'>
        <label>Secret Key</label><input type='password' id='eSec'
placeholder='fGP2Ts...'>
        <label>Пауза при заряді менше (%)</label><input type='number' id='ePause'
placeholder='20' min='0' max='100'>
    </div>
</div>

```

					PE22.468332.001	Лис
						5
Зм.	Лис	№ докум.	Підпис	Дата		

Лис
6

```

value='${i}'>${p.name}</option>`).join('');if(val&&sel.querySelector(`option[
value="${val}"`))sel.value=val}
function addToQueue(f){Q.push(f);renderQueue()}
function clearQueue(){Q=[];renderQueue()}
function removeFromQueue(i){Q.splice(i,1);renderQueue()}
function renderQueue(){const
el=document.getElementById('queueList');el.innerHTML=Q.length?Q.map((f,i)=>`<
div class='qi'><span class='qn'>${i+1}</span><span
style='flex:1;overflow:hidden;text-overflow:ellipsis;white-
space:nowrap'>${f}</span><button class='btn-r' style='padding:4px 8px;font-
size:12px' onclick='removeFromQueue(${i})>X</button></div>`).join(''):'<p
style="color:var(--mu)">Порожньо</p>'
function sendQueue(){if(!Q.length)return;const
pi=document.getElementById('queuePrinter').value;if(pi==='')return;fetch('/st
art_queue?pi='+pi+'&q='+encodeURIComponent(Q.join(','))).then(()=>{Q=[];rende
rQueue()})}
function badge(s){const m={IDLE:'b-idle',RUNNING:'b-run',FINISH:'b-
fin',PAUSE:'b-pau',FAILED:'b-err',OFFLINE:'b-off'};return`<span class='badge
${m[s]||"b-idle"}>${s}</span>`}
function fmtTime(m){if(m<=0)return'—';const h=Math.floor(m/60),mn=m%60;return
h`$${h}r $${mn}хв`:`$${mn} хв`}
function
setPrinterSpeed(idx,sel){fetch(`/set_speed?i=${idx}&lvl=${sel.value}`)}
function renderPrinters(printers){
const el=document.getElementById('printers');
if(!printers.length){el.innerHTML=`<div style='padding:40px;text-
align:center;color:var(--mu);grid-column:1/-1;'>Натисніть <b
style='color:var(--ac)'>+ Пристрій</b>, щоб додати принтер</div>`;return}
el.innerHTML=printers.map((p,i)=>{
const
layers=(p.state==='RUNNING'&&p.totalL>0)?`${p.layer}/${p.totalL}`:'—';
const spd=(v)=>p.speed===v?'selected':'';
return`<div class='card'>
<div class='ctitle'><span>#9632;
${p.name}</span>${badge(p.state)}</div>
<div class='prow'><span>#127777;
Сопло</span><b>${p.nozzle}°C</b></div>
<div class='prow'><span>#127760; Стіл</span><b>${p.bed}°C</b></div>
<div class='prow'><span>#128190; Файл</span><b style='font-
size:14px;max-width:180px;overflow:hidden;text-overflow:ellipsis;white-
space:nowrap'>${p.file}</b></div>
<div class='prow'><span>#12498; Шлях</span><b style='color:var(--
ac)'>${layers}</b></div>
<div class='prow'><span>🚀 Швидкість</span>
<select class='speed-sel' onchange='setPrinterSpeed(${i},this)'
${p.state!=='RUNNING'? 'disabled':''}>
<option value='50' ${spd(1)}>Silent (50%)</option>
<option value='100' ${spd(2)}>Standard (100%)</option>
<option value='124' ${spd(3)}>Sport (124%)</option>
<option value='166' ${spd(4)}>Ludicrous (166%)</option>
</select>
</div>
<div class='prow'><span>🏠 Готово о</span><b style='color:var(--
gn)'>${p.end}</b></div>
<div class='prow'><span>#9201;
Залишилось</span><b>${fmtTime(p.remain)}</b></div>
<div class='progbar'><div class='progfill'
style='width:${p.pct}%></div></div>
<div style='text-align:right;font-size:14px;color:var(--
mu)'>${p.pct}%</div>
<div class='row' style='margin-top:14px'>
<button class='btn-b' onclick='fetch("/home?i="+${i})>G28
(Home)</button>

```

									Лис
									7
Зм.	Лис	№ докум.	Підпис	Дата					

PE22.468332.001

```

        <button class='btn-r' onclick='deletePrinter(${i})'>&#10006;
Видалити</button>
    </div>
</div>`)).join('')
function renderEco(d){
    document.getElementById('stEco').textContent='⚡ '+d.esoc+'%';
    document.getElementById('stTime').textContent='🕒 '+fmtTime(d.etim);
    document.getElementById('stPow').textContent='⬇️ '+d.epow+' Вт';
    if(d.ecrit)document.getElementById('hdrEco').classList.add('crit');else
document.getElementById('hdrEco').classList.remove('crit')}
setInterval(()=>{fetch('/status').then(r=>r.json()).then(d=>{renderPrinters(d
.printers);renderEco(d);updateQueueSelect(d.printers)}}),2500)
function
deletePrinter(i){if(confirm('Видалити?'))fetch('/del_printer?id='+i).then(()=
>location.reload())}
window.onload=()=>{loadFiles();fetch('/status').then(r=>r.json()).then(d=>{if
(d.tgSet)document.getElementById('tgPasswordField').placeholder='●●●●●'})}
</script>
</body></html>
)html";

//Конфіг: Читання/Запис
void loadConfig() {
    prefs.begin("cfg", true);
    printerCount = prefs.getInt("pCnt", 0);
    for (int i = 0; i < printerCount && i < MAX_PRINTERS; i++) {
        String p = "p" + String(i) + "_";
        printerCfg[i].name = prefs.getString((p + "name").c_str(), "Принтер "
+ String(i+1));
        printerCfg[i].ip = prefs.getString((p + "ip").c_str(), "");
        printerCfg[i].password = prefs.getString((p + "pass").c_str(), "");
        printerCfg[i].serial = prefs.getString((p + "sn").c_str(), "");
    }
    eco.sn = prefs.getString("eSn", "");
    eco.appKey = prefs.getString("eKey", "");
    eco.secret = prefs.getString("eSec", "");
    eco.pauseAt = prefs.getInt("ePause", 20);
    tgPassword = prefs.getString("tgPass", "");
    tgChatId = prefs.getString("tgChat", "");
    prefs.end();
}

void saveConfig() {
    prefs.begin("cfg", false);
    prefs.putInt("pCnt", printerCount);
    for (int i = 0; i < printerCount; i++) {
        String p = "p" + String(i) + "_";
        prefs.putString((p + "name").c_str(), printerCfg[i].name);
        prefs.putString((p + "ip").c_str(), printerCfg[i].ip);
        prefs.putString((p + "pass").c_str(), printerCfg[i].password);
        prefs.putString((p + "sn").c_str(), printerCfg[i].serial);
    }
    prefs.putString("eSn", eco.sn);
    prefs.putString("eKey", eco.appKey);
    prefs.putString("eSec", eco.secret);
    prefs.putInt("ePause", eco.pauseAt);
    prefs.putString("tgPass", tgPassword);
    prefs.putString("tgChat", tgChatId);
    prefs.end();
}

void loadFiles() {
    prefs.begin("files", true);
    sdFileCount = min((int)prefs.getInt("cnt", 0), (int)MAX_FILES);

```

					PE22.468332.001	Лис
						8
Зм.	Лис	№ докум.	Підпис	Дата		

```

    for (int i = 0; i < sdFileCount; i++)
        sdFiles[i] = prefs.getString(("f" + String(i)).c_str(), "");
    prefs.end();
}

void saveFiles() {
    prefs.begin("files", false);
    prefs.putInt("cnt", sdFileCount);
    for (int i = 0; i < sdFileCount; i++)
        prefs.putString(("f" + String(i)).c_str(), sdFiles[i]);
    prefs.end();
}

//Helpers
bool isTgAuthorized(const String& chatId) {
    return !tgChatId.isEmpty() && tgChatId == chatId;
}

String fmtTime(int m) {
    if (m <= 0) return "-";
    int h = m / 60, mn = m % 60;
    if (h > 0) return String(h) + "г " + String(mn) + "хв";
    return String(mn) + " хв";
}

String speedName(int lvl) {
    switch(lvl) {
        case 1: return "🔇 Тихий (50%)";
        case 3: return "🏃 Спорт (124%)";
        case 4: return "🚀 Шалений (166%)";
        default: return "⚡ Стандарт (100%)";
    }
}

String buildMainMenu() {
    return "{\"inline_keyboard\":["
        "[{\"text\": \"📺 Стан пристроїв\", \"callback_data\": \"status\"}],",
        "[{\"text\": \"📁 Файли\", \"callback_data\": \"list_files\"}],",
        "[{\"text\": \"📄 Моя черга\", \"callback_data\": \"my_queue\"}],",
        "[{\"text\": \"🗑️ Очистити чергу\", \"callback_data\": \"clear_q\"}]]\"";
}

//EcoFlow: Підпис HMAC-SHA256
String ecoflowSign(const String& params, const String& secret) {
    byte hmac[32]; char hex[65];
    mbedtls_md_context_t ctx;
    mbedtls_md_init(&ctx);
    mbedtls_md_setup(&ctx, mbedtls_md_info_from_type(MBEDTLS_MD_SHA256), 1);
    mbedtls_md_hmac_starts(&ctx, (const uint8_t*)secret.c_str(),
secret.length());
    mbedtls_md_hmac_update(&ctx, (const uint8_t*)params.c_str(),
params.length());
    mbedtls_md_hmac_finish(&ctx, hmac);
    mbedtls_md_free(&ctx);
    for (int i = 0; i < 32; i++) sprintf(&hex[i*2], "%02x", hmac[i]);
    return String(hex);
}

//Telegram: низькорівнева відправка
static int tgHttpPost(const String& path, const String& postBody) {
    HTTPClient http;
    String url = "https://api.telegram.org/bot" + tgToken + path;

```

					PE22.468332.001	Лис
						9
Зм.	Лис	№ докум.	Підпис	Дата		

```

http.begin(tgClient, url);
http.addHeader("Content-Type", "application/x-www-form-urlencoded");
http.setTimeout(8000);
int code = http.POST(postBody);
http.end();
return code;
}

//Telegram: Відправка повідомлення
void sendTelegramMessage(const String& text, const String& replyMarkup) {
    if (tgChatId.isEmpty() || tgToken.isEmpty() || WiFi.status() !=
WL_CONNECTED) return;

    if (xSemaphoreTake(tgMutex, pdMS_TO_TICKS(5000)) != pdTRUE) {
        Serial.println("[SEND] mutex timeout!");
        return;
    }

    String payload = "chat_id=" + tgChatId + "&text=" + text +
"&parse_mode=HTML";
    if (!replyMarkup.isEmpty()) payload += "&reply_markup=" + replyMarkup;

    int code = tgHttpPost("/sendMessage", payload);
    Serial.printf("[SEND] HTTP %d\n", code);

    xSemaphoreGive(tgMutex);
}

// Відправка будь-якому chat_id (для авторизаційних відповідей)
void sendTelegramTo(const String& chatId, const String& text, const String&
replyMarkup = "") {
    if (tgToken.isEmpty() || WiFi.status() != WL_CONNECTED) return;

    if (xSemaphoreTake(tgMutex, pdMS_TO_TICKS(5000)) != pdTRUE) return;

    String payload = "chat_id=" + chatId + "&text=" + text +
"&parse_mode=HTML";
    if (!replyMarkup.isEmpty()) payload += "&reply_markup=" + replyMarkup;

    tgHttpPost("/sendMessage", payload);
    xSemaphoreGive(tgMutex);
}

//EcoFlow Task
void TaskEcoFlow(void* pv) {
    unsigned long lastEcoCall = 0;
    for (;;) {
        unsigned long now = millis();
        if (WiFi.status() == WL_CONNECTED && !eco.appKey.isEmpty()) {
            if (now - lastEcoCall >= ECOFLOW_INTERVAL || lastEcoCall == 0) {
                lastEcoCall = now;

                struct timeval tv; gettimeofday(&tv, NULL);
                char tsBuf[32];
                snprintf(tsBuf, sizeof(tsBuf), "%llu",
                        (unsigned long long)tv.tv_sec * 1000ULL + tv.tv_usec /
1000ULL);
                String timestamp = String(tsBuf);
                String nonce = String(random(100000, 999999));
                String signStr = "accessKey=" + eco.appKey + "&nonce=" + nonce +
"&timestamp=" + timestamp;
                String sign = ecoflowSign(signStr, eco.secret);

                WiFiClientSecure ecoClient; ecoClient.setInsecure();
                HTTPClient http;

```

					PE22.468332.001	Лис
						10
Зм.	Лис	№ докум.	Підпис	Дата		

```

String url = eco.host + "/iot-open/sign/device/quota/all?sn=" +
eco.sn;
http.begin(ecoClient, url);
http.addHeader("accessKey", eco.appKey);
http.addHeader("nonce", nonce);
http.addHeader("timestamp", timestamp);
http.addHeader("sign", sign);
http.addHeader("Content-Type", "application/json");
http.setTimeout(10000);

int httpCode = http.GET();
if (httpCode == 200) {
String payload = http.getString();
if (payload.length() > 200) {
auto getVal = [](const String& text, const String& key) -> int {
int pos = text.indexOf("\"" + key + "\"");
if (pos == -1) return -1;
pos = text.indexOf(":", pos);
if (pos == -1) return -1;
pos++;
while (pos < (int)text.length() && isSpace(text.charAt(pos)))
pos++;
int end = pos;
while (end < (int)text.length() && (isDigit(text.charAt(end))
|| text.charAt(end) == '-')) end++;
return (end > pos) ? text.substring(pos, end).toInt() : -1;
};

int v;
v = getVal(payload, "pd.soc"); if (v != -1)
eco.soc = v;
v = getVal(payload, "pd.wattsOutSum"); if (v != -1)
eco.outPower = v;
v = getVal(payload, "mppt.inWatts");
if (v == -1) v = getVal(payload, "inv.inputWatts"); if (v != -1)
eco.inPower = v;
v = getVal(payload, "bms_emsStatus.dsgRemainTime"); if (v != -1)
eco.remainMin = v;

bool prevCritical = eco.critical;
eco.critical = (eco.soc <= eco.pauseAt && eco.inPower == 0);

if (eco.prevInPower != -1) {
if (eco.prevInPower > 0 && eco.inPower == 0)
sendTelegramMessage("💡 <b>⚠️ Втрата живлення!</b>\n🔌 Ферма
на батареї EcoFlow\n⚡ Споживання: " + String(eco.outPower) + " Вт\n🔋 Заряд:
" + String(eco.soc) + "%");
else if (eco.prevInPower == 0 && eco.inPower > 0)
sendTelegramMessage("⚡ <b>✅ Живлення відновлено!</b>\n🔌
Вхідна потужність: " + String(eco.inPower) + " Вт");
}
eco.prevInPower = eco.inPower;
if (eco.critical && !prevCritical)
sendTelegramMessage("🔴 <b>⚠️ КРИТИЧНИЙ ЗАРЯД ECOFLOW (" +
String(eco.soc) + "%)</b>\nЧергу призупинено автоматично!");
}
}
http.end();
}
}
vTaskDelay(2000 / portTICK_PERIOD_MS);
}
}

```

					PE22.468332.001	Лис
						11
Зм.	Лис	№ докум.	Підпис	Дата		

```

//Telegram Task
struct TgUpdate {
    long long updateId = 0;
    String chatId;
    String text;
    String data;
    bool isCallback = false;
    bool valid = false;
};

#define TG_MAX_UPDATES 8
static TgUpdate tgUpdQueue[TG_MAX_UPDATES];
static int      tgUpdQueueLen = 0;

void handleTelegramUpdates() {
    if (tgToken.isEmpty()) return;

    if (xSemaphoreTake(tgMutex, pdMS_TO_TICKS(10000)) != pdTRUE) {
        Serial.println("[TG] mutex timeout in getUpdates");
        return;
    }

    HTTPClient http;
    String path = "/getUpdates?timeout=3";
    if (lastUpdateId > 0) path += "&offset=" + String(lastUpdateId + 1);

    String url = "https://api.telegram.org/bot" + tgToken + path;
    http.begin(tgClient, url);
    http.setTimeout(10000);
    int httpCode = http.GET();

    tgUpdQueueLen = 0;

    if (httpCode == 200) {
        WiFiClient* stream = http.getStreamPtr();
        JsonDocument doc;
        DeserializationError err = deserializeJson(doc, *stream);

        if (!err) {
            JsonArray updates = doc["result"];
            for (JsonObject update : updates) {
                if (tgUpdQueueLen >= TG_MAX_UPDATES) break;

                long long uid = update["update_id"].as<long long>();
                if (uid > lastUpdateId) lastUpdateId = uid;

                TgUpdate& u = tgUpdQueue[tgUpdQueueLen];
                u = TgUpdate{};
                u.updateId = uid;

                if (update.containsKey("message")) {
                    u.chatId = update["message"]["chat"]["id"].as<String>();
                    u.text    = update["message"]["text"].as<String>();
                    u.text.trim();
                    u.isCallback = false;
                    u.valid = true;
                    tgUpdQueueLen++;
                } else if (update.containsKey("callback_query")) {
                    u.chatId =
update["callback_query"]["message"]["chat"]["id"].as<String>();
                    u.data    = update["callback_query"]["data"].as<String>();
                    u.isCallback = true;
                    u.valid = true;
                    tgUpdQueueLen++;
                }
            }
        }
    }
}

```

					PE22.468332.001	Лис
						12
Зм.	Лис	№ докум.	Підпис	Дата		

```

    }
    } else {
        Serial.printf("[TG] JSON parse error: %s\n", err.c_str());
    }
} else {
    Serial.printf("[TG] getUpdates HTTP: %d\n", httpCode);
}
http.end();
xSemaphoreGive(tgMutex);

//Обробка
for (int qi = 0; qi < tgUpdQueueLen; qi++) {
    TgUpdate& u = tgUpdQueue[qi];
    if (!u.valid) continue;

    if (!u.isCallback) {
        //Текстове повідомлення
        bool authorized = isTgAuthorized(u.chatId);

        if (!authorized) {
            //Спроба авторизації
            if (!tgPassword.isEmpty() && u.text == tgPassword) {
                tgChatId = u.chatId;
                saveConfig();
                Serial.println("[TG] Auth OK!");
                sendTelegramTo(u.chatId,
                    "✅ <b>Авторизацію пройдено!</b>\n\nТепер ви можете керувати
фермою через цього бота.",
                    buildMainMenu());
            } else {
                //Підказка для неавторизованих
                sendTelegramTo(u.chatId,
                    "🔒 <b>Потрібна авторизація</b>\n\n"
                    "Введіть код доступу, який ви задали у веб-інтерфейсі
фермери.\n\n"
                    "💡 <i>Код налаштовується на сторінці керування у полі «Код
доступу TG»</i>");
            }
            continue;
        }

        //Авторизований – обробляємо команди
        if (u.text == "/start" || u.text == "/menu" || u.text == "меню") {
            sendTelegramTo(u.chatId,
                "📁 <b>Bambu Farm</b> – головне меню\n\nОберіть дію:",
                buildMainMenu());
        }
    } else {
        //Callback кнопки

        //Якщо не авторизований – підказка
        if (!isTgAuthorized(u.chatId)) {
            sendTelegramTo(u.chatId,
                "🔒 <b>Потрібна авторизація</b>\n\n"
                "Надішліть код доступу з веб-інтерфейсу фермери.\n"
                "💡 <i>Поле «Код доступу TG» на головній сторінці</i>");
            continue;
        }

        //status
        if (u.data == "status") {
            String msg = "📶 <b>Стан пристроїв</b>\n";

```

					PE22.468332.001	Лис
						13
Зм.	Лис	№ докум.	Підпис	Дата		

```

        msg += " 🖨️ EcoFlow: <b>" + String(eco.soc) + "%</b> (" +
String(eco.outPower) + " Вт)\n\n";

String kb = "{\\"inline_keyboard\":[\"";

for (int i = 0; i < printerCount; i++) {
    auto& st = printerSt[i];
    String stateIcon = "●";
    if (st.state == "RUNNING")        stateIcon = "🟢";
    else if (st.state == "PAUSE")     stateIcon = "🟡";
    else if (st.state == "FINISH")    stateIcon = "🟡";
    else if (st.state == "FAILED")    stateIcon = "🔴";
    else if (st.state == "IDLE")      stateIcon = "⬜";

    msg += stateIcon + " 🖨️ Принтер <b>" + printerCfg[i].name + "</b>:"
" + st.state + "\n";

    if (st.state == "RUNNING" || st.state == "PAUSE") {
        msg += " 📄 " + st.file + "\n";
        msg += " 📊 " + String(st.percent) + "% | шаг " +
String(st.layerNum) + "/" + String(st.totalLayer) + "\n";
        msg += " ⌚ Залишилось: " + fmtTime(st.remainMin) + "\n";
        msg += " 🚀 Швидкість: " + speedName(st.speedLevel) + "\n";
        msg += "\n";

        if (st.state == "RUNNING") {
            kb += "[{\\"text\":" + "⏸️ Пауза: " + printerCfg[i].name +
"\",\\"callback_data\":" + "pause_" + String(i) + "\"}],\"";
        } else {
            kb += "[{\\"text\":" + "▶️ Продовжити: " + printerCfg[i].name +
"\",\\"callback_data\":" + "resume_" + String(i) + "\"}],\"";
        }
    }
}

kb += "[{\\"text\":" + "📡
Оновити\"," + "\"callback_data\":" + "\"status\"",{\\"text\":" + "🏠
Меню\"," + "\"callback_data\":" + "\"menu\"\"}}]\"";
sendTelegramTo(u.chatId, msg, kb);
}

//menu
else if (u.data == "menu") {
    sendTelegramTo(u.chatId, " 🖨️ <b>Головне меню</b>\n\nОберіть дію:",
buildMainMenu());
}

//list_files
else if (u.data == "list_files") {
    if (sdFileCount == 0) {
        sendTelegramTo(u.chatId, " 📁 Список файлів порожній.\n\n💡 Додайте
файли через веб-інтерфейс.",
            "{\\"inline_keyboard\":[[{\\"text\":" + "🏠
Меню\"," + "\"callback_data\":" + "\"menu\"\"}}]\"");
    } else {
        String msg = " 📁 <b>Файли для друку</b>\n\nОберіть файл, щоб додати
до черги:";
        String kb = "{\\"inline_keyboard\":[\"";
        for (int i = 0; i < sdFileCount; i++) {
            if (i > 0) kb += ",";

```

					PE22.468332.001	Лис
Зм.	Лис	№ докум.	Підпис	Дата		14

```

        kb += "[{\"text\": \"➕\" + sdFiles[i] +
        \"\", \"callback_data\": \"addq_\" + String(i) + \"\"}]]";
    }
    kb += "[{\"text\": \"📁 Моя
черга\", \"callback_data\": \"my_queue\"}, {\"text\": \"🏠
Меню\", \"callback_data\": \"menu\"}]]]";
    sendTelegramTo(u.chatId, msg, kb);
}

//my_queue - перегляд черги бота
else if (u.data == "my_queue") {
    if (tgBotQueue.total == 0) {
        String msg = "📁 <b>Черга порожня</b>\n\n💡 Додайте файли через «📁
Файли»";
        String kb = "{\"inline_keyboard\": [
            [{\"text\": \"📁 Додати
файли\", \"callback_data\": \"list_files\"}],
            [{\"text\": \"🏠 Меню\", \"callback_data\": \"menu\"}]]]";
        sendTelegramTo(u.chatId, msg, kb);
    } else {
        String msg = "📁 <b>Ваша черга (\" + String(tgBotQueue.total) + \"
шт.):</b>\n\n";
        for (int i = 0; i < tgBotQueue.total; i++) {
            msg += String(i+1) + ". \" + tgBotQueue.items[i] + "\n";
        }
        msg += "\nОберіть принтер і запустіть, або видаліть зайве:";

        String kb = "{\"inline_keyboard\": [
            // Кнопки видалення (перші 5 позицій, щоб не перевантажувати)
            int showDel = min(tgBotQueue.total, 5);
            for (int i = 0; i < showDel; i++) {
                kb += "[{\"text\": \"🗑️ Видалити: \" + tgBotQueue.items[i] +
                \"\", \"callback_data\": \"rmq_\" + String(i) + \"\"}],";
            }
            // Кнопки запуску на принтер
            for (int i = 0; i < printerCount; i++) {
                kb += "[{\"text\": \"🚀 Запустити на: \" + printerCfg[i].name +
                \"\", \"callback_data\": \"runq_\" + String(i) + \"\"}],";
            }
            kb += "[{\"text\": \"📁 Ще
файли\", \"callback_data\": \"list_files\"}, {\"text\": \"🗑️ Очистити
все\", \"callback_data\": \"clear_q\"}],";
            "[{\"text\": \"🏠 Меню\", \"callback_data\": \"menu\"}]]]";
            sendTelegramTo(u.chatId, msg, kb);
        }
    }

    //addq - додати до черги бота
    else if (u.data.startsWith("addq_")) {
        int fileId = u.data.substring(5).toInt();
        if (fileId >= 0 && fileId < sdFileCount && tgBotQueue.total <
TG_QUEUE_MAX) {
            tgBotQueue.items[tgBotQueue.total++] = sdFiles[fileId];
            String msg = "✅ Додано до черги: <code>\" + sdFiles[fileId] +
            "</code>\n\n";
            "📁 Всього в черзі: <b>\" + String(tgBotQueue.total) +
            "</b>";
            String kb = "{\"inline_keyboard\": [
                [{\"text\": \"➕ Ще файли\", \"callback_data\": \"list_files\"}],
                [{\"text\": \"📁 Переглянути
чергу\", \"callback_data\": \"my_queue\"}],

```

									Лис
									15
Зм.	Лис	№ докум.	Підпис	Дата					

PE22.468332.001

```

        "[{"text\":" + 🏠 + " Меню\","callback_data\":"menu\"}]]}";
        sendTelegramTo(u.chatId, msg, kb);
    } else if (tgBotQueue.total >= TG_QUEUE_MAX) {
        sendTelegramTo(u.chatId, "⚠️ Черга заповнена! Спочатку запустіть
або очистіть.",
            "{\inline_keyboard\":[[{"text\":" + 📄 + " Моя
черга\","callback_data\":"my_queue\"}]]}");
    }

    //rmq_ - видалити позицію з черги бота
    else if (u.data.startsWith("rmq_")) {
        int idx = u.data.substring(4).toInt();
        if (idx >= 0 && idx < tgBotQueue.total) {
            String removed = tgBotQueue.items[idx];
            for (int i = idx; i < tgBotQueue.total - 1; i++)
                tgBotQueue.items[i] = tgBotQueue.items[i+1];
            tgBotQueue.total--;
            sendTelegramTo(u.chatId, "🗑️ Видалено: <code>" + removed +
"</code>",
                "{\inline_keyboard\":[[{"text\":" + 📄 + " Моя
черга\","callback_data\":"my_queue\"},{\"text\":" + 🏠 +
" Меню\","callback_data\":"menu\"}]]}");
        }

        //runq_ - запустити чергу бота на принтер
        else if (u.data.startsWith("runq_")) {
            int prId = u.data.substring(5).toInt();
            if (prId >= 0 && prId < printerCount && tgBotQueue.total > 0) {
                //Копіємо чергу бота у основну чергу
                queue.printerIdx = prId;
                queue.total = min(tgBotQueue.total, QUEUE_MAX);
                queue.index = 0;
                queue.active = true;
                queue.waiting = false;
                for (int i = 0; i < queue.total; i++)
                    queue.items[i] = tgBotQueue.items[i];
                tgBotQueue.total = 0; //очищаємо чергу бота після запуску

                String msg = "🚀 <b>Черга запущена на " + printerCfg[prId].name +
"!</b>\n\n";
                msg += "📄 Файлів: " + String(queue.total);
                sendTelegramTo(u.chatId, msg,
                    "{\inline_keyboard\":[[{"text\":" + 📄 +
"Стан\","callback_data\":"status\"},{\"text\":" + 🏠 +
" Меню\","callback_data\":"menu\"}]]}");
                processNextInQueue();
            }

            //clear_q
            else if (u.data == "clear_q") {
                queue.total = 0; queue.index = 0; queue.active = false; queue.waiting
= false;
                tgBotQueue.total = 0;
                sendTelegramTo(u.chatId, "🗑️ <b>Чергу очищено</b>", buildMainMenu());
            }

            //pause_
            else if (u.data.startsWith("pause_")) {
                int prId = u.data.substring(6).toInt();
                if (prId >= 0 && prId < printerCount) {

```

					PE22.468332.001	Лис
						16
Зм.	Лис	№ докум.	Підпис	Дата		

```

        pausePrint(prId);
        sendTelegramTo(u.chatId,
            "🔴 <b>Паузу встановлено</b>\n🖨 " + printerCfg[prId].name,
            "{\n\"inline_keyboard\":[[{\n\"text\":\n▶\nПродовжити\", \"callback_data\":\n\"resume_\" + String(prId) +\n\"\"}], [{\n\"text\":\n🏠 Стан\", \"callback_data\":\n\"status\"}]]}");
    }

    //resume_
    else if (u.data.startsWith("resume_")) {
        int prId = u.data.substring(7).toInt();
        if (prId >= 0 && prId < printerCount) {
            resumePrint(prId);
            sendTelegramTo(u.chatId,
                "▶ <b>Друк відновлено</b>\n🖨 " + printerCfg[prId].name,
                "{\n\"inline_keyboard\":[[{\n\"text\":\n🏠\nСтан\", \"callback_data\":\n\"status\"}, {\n\"text\":\n🏠\nМеню\", \"callback_data\":\n\"menu\"}]]}");
        }
    }
}

void TaskTelegram(void* pv) {
    for (;;) {
        if (WiFi.status() == WL_CONNECTED && !tgToken.isEmpty() &&
            !tgPassword.isEmpty()) {
            handleTelegramUpdates();
        }
        vTaskDelay(2000 / portTICK_PERIOD_MS); // v2.9: зменшено з 4500 для
        швидшої реакції
    }
}

//MQTT
void mqttCallback(char* topic, byte* payload, unsigned int length) {
    int prIdx = -1;
    for (int i = 0; i < printerCount; i++) {
        if (String(topic).startsWith("device/" + printerCfg[i].serial)) { prIdx =
i; break; }
    }
    if (prIdx < 0) return;

    JsonDocument filter;
    filter["print"]["nozzle_temper"] = true;
    filter["print"]["bed_temper"] = true;
    filter["print"]["mc_percent"] = true;
    filter["print"]["mc_remaining_time"] = true;
    filter["print"]["subtask_name"] = true;
    filter["print"]["gcode_state"] = true;
    filter["print"]["layer_num"] = true;
    filter["print"]["total_layer_num"] = true;
    filter["print"]["spd_lvl"] = true;

    JsonDocument doc;
    if (deserializeJson(doc, payload, length,
        DeserializationOption::Filter(filter))) return;

    JsonObject pr = doc["print"];
    if (!pr.isNull()) {
        auto& st = printerSt[prIdx];

```

					PE22.468332.001	Лис
						17
Зм.	Лис	№ докум.	Підпис	Дата		

```

        if (pr["nozzle_temper"].is<float>())          st.nozzle      =
pr["nozzle_temper"];
        if (pr["bed_temper"].is<float>())            st.bed        =
pr["bed_temper"];
        if (pr["mc_percent"].is<int>())              st.percent    =
pr["mc_percent"];
        if (pr["mc_remaining_time"].is<int>())       st.remainMin   =
pr["mc_remaining_time"];
        if (pr["layer_num"].is<int>())               st.layerNum    =
pr["layer_num"];
        if (pr["total_layer_num"].is<int>())         st.totalLayer  =
pr["total_layer_num"];
        if (pr["spd_lvl"].is<int>())                 st.speedLevel  = pr["spd_lvl"];

        if (pr["subtask_name"].is<const char*>()) {
            String fn = pr["subtask_name"].as<String>();
            if (!fn.isEmpty()) st.file = fn;
        }

        if (pr["gcode_state"].is<const char*>()) {
            String ns = pr["gcode_state"].as<String>();

            //v2.8: якщо mqttReady ще false – це перший pushall після реконекту
            //Ми просто фіксуємо поточний стан як «відомий», без сповіщень
            if (!st.mqttReady) {
                st.prevState = ns;
                st.state      = ns;
                st.mqttReady = true;
                Serial.printf("[MQTT] Printer %d initial state: %s\n", prIdx,
ns.c_str());
                return;
            }

            //Справжня зміна стану – надсилаємо сповіщення
            if (ns != st.prevState) {
                if (ns == "RUNNING" && st.prevState != "PAUSE") {
                    sendTelegramMessage("🔵 <b>[" + printerCfg[prIdx].name + "]</b>
розпочав друк\n"
                    "📄 <code>" + st.file + "</code>\n"
                    "📊 EcoFlow: " + String(eco.soc) + "%");
                } else if (ns == "FINISH") {
                    sendTelegramMessage("🏁 <b>[" + printerCfg[prIdx].name + "]</b>
завершив друк!\n"
                    "📄 <code>" + st.file + "</code>");
                } else if (ns == "PAUSE") {
                    sendTelegramMessage("⏸ <b>[" + printerCfg[prIdx].name + "]</b> на
паузі\n"
                    "📄 <code>" + st.file + "</code>\n"
                    "📊 Проріс: " + String(st.percent) + "%");
                } else if (ns == "FAILED") {
                    sendTelegramMessage("❌ <b>[" + printerCfg[prIdx].name + "]</b>
помилка друку!\n"
                    "📄 <code>" + st.file + "</code>",
                    "{ \"inline_keyboard\": [[ { \"text\": \"📄 Стан\", \"callback_data\": \"status\" } ] ] }");
                }

                if (ns == "FINISH" && st.prevState == "RUNNING" && queue.active &&
queue.printerIdx == prIdx) {
                    queue.waiting = true;
                    queue.waitStart = millis();
                }
                st.prevState = ns;
            }
        }

```

					PE22.468332.001	Лис
						18
Зм.	Лис	№ докум.	Підпис	Дата		

```

    }

    st.state = ns;
}
}

void mqttReconnect(int idx) {
    if (idx >= printerCount) return;
    auto& cfg = printerCfg[idx];
    espClients[idx]->stop();

    //v2.8: скидаємо mqttReady – наступний pushall буде «знайомством», без
    сповіщень
    printerSt[idx].mqttReady = false;

    String clientId = "farm_" + cfg.serial.substring(0, 6) + "_" +
    String(esp_random() & 0xFFFF, HEX);
    if (mqttClients[idx]->connect(clientId.c_str(), "bblp",
    cfg.password.c_str())) {
        printerSt[idx].active = true;
        connectedAtMs[idx] = millis();
        pushallPending[idx] = true;
        String base = "device/" + cfg.serial;
        mqttClients[idx]->subscribe((base + "/report").c_str());
    } else {
        printerSt[idx].state = "OFFLINE";
        Serial.printf("[MQTT] Connect failed for printer %d\n", idx);
    }
}

void startPrint(int prIdx, const String& filename) {
    JsonDocument doc;
    JsonObject p = doc["print"].to<JsonObject>();

    p["command"] = "project_file";
    p["sequence_id"] = String(millis() / 1000);

    //Принтер очікує шлях до gcode всередині 3mf архіву
    p["param"] = "Metadata/plate_1.gcode";

    p["project_id"] = "0";
    p["profile_id"] = "0";
    p["task_id"] = "0";
    p["subtask_id"] = "0";
    p["subtask_name"] = filename;

    //Передаємо шлях як у url, так і у file для сумісності з різними прошивками
    p["file"] = "file:///sdcard/Auto/" + filename;
    p["url"] = "file:///sdcard/Auto/" + filename;

    p["bed_type"] = "auto";
    p["bed_levelling"] = true;
    p["flow_cali"] = false;
    p["vibration_cali"] = false;
    p["layer_inspect"] = false;
    p["use_ams"] = false;

    String payload;
    serializeJson(doc, payload);

    Serial.printf("[MQTT] startPrint payload: %s\n", payload.c_str());

    String topic = "device/" + printerCfg[prIdx].serial + "/request";
    mqttClients[prIdx]->publish(topic.c_str(), payload.c_str());
}

```

					PE22.468332.001	Лис
						19
Зм.	Лис	№ докум.	Підпис	Дата		

```

}

void changeSpeed(int prIdx, int level) {
    if (prIdx >= printerCount || !mqttClients[prIdx]->connected()) return;
    JsonDocument doc;
    int lvlId = 2;
    if (level == 50) lvlId = 1;
    else if (level == 124) lvlId = 3;
    else if (level == 166) lvlId = 4;
    doc["print"]["command"] = "speed";
    doc["print"]["param"] = String(lvlId);
    doc["print"]["sequence_id"] = String(millis() / 1000);
    String payload; serializeJson(doc, payload);
    String topic = "device/" + printerCfg[prIdx].serial + "/request";
    mqttClients[prIdx]->publish(topic.c_str(), payload.c_str());
}

void pausePrint(int prIdx) {
    if (prIdx >= printerCount || !mqttClients[prIdx]->connected()) return;
    String topic = "device/" + printerCfg[prIdx].serial + "/request";
    mqttClients[prIdx]->publish(topic.c_str(),
    "{\"print\":{\"command\":\"pause\",\"sequence_id\":\"9000\"}}");
}

void resumePrint(int prIdx) {
    if (prIdx >= printerCount || !mqttClients[prIdx]->connected()) return;
    String topic = "device/" + printerCfg[prIdx].serial + "/request";
    mqttClients[prIdx]->publish(topic.c_str(),
    "{\"print\":{\"command\":\"resume\",\"sequence_id\":\"9001\"}}");
}

void processNextInQueue() {
    if (queue.index >= queue.total) { queue.active = false; return; }
    if (eco.critical) { queue.waiting = true; queue.waitStart = millis();
return; }
    startPrint(queue.printerIdx, queue.items[queue.index]);
    queue.index++;
}

void TaskMQTT(void* pv) {
    for (;;) {
        unsigned long now = millis();
        for (int i = 0; i < printerCount; i++) {
            if (!mqttClients[i]) continue;
            if (!mqttClients[i]->connected()) { mqttReconnect(i); vTaskDelay(2000 /
portTICK_PERIOD_MS); continue; }
            mqttClients[i]->loop();
            if (pushallPending[i] && (now - connectedAtMs[i] >= 2000)) {
                pushallPending[i] = false;
                String topic = "device/" + printerCfg[i].serial + "/request";
                mqttClients[i]->publish(topic.c_str(),
                "{\"pushing\":{\"command\":\"pushall\",\"sequence_id\":\"0\",\"version\":1}}");
            }
            if (now - lastPushallMs[i] >= PUSHALL_INTERVAL) {
                lastPushallMs[i] = now;
                String topic = "device/" + printerCfg[i].serial + "/request";
                mqttClients[i]->publish(topic.c_str(),
                "{\"pushing\":{\"command\":\"pushall\",\"sequence_id\":\"0\",\"version\":1}}");
            }
            if (eco.critical && printerSt[i].state == "RUNNING") pausePrint(i);
        }
        if (queue.waiting && (now - queue.waitStart >= WAIT_BETWEEN_MS)) {
            queue.waiting = false;

```

					PE22.468332.001	Лис
						20
Зм.	Лис	№ докум.	Підпис	Дата		

```

        processNextInQueue();
    }
    vTaskDelay(50 / portTICK_PERIOD_MS);
}

//HTTP сервер
void startNormalMode() {
    for (int i = 0; i < printerCount && i < MAX_PRINTERS; i++) {
        espClients[i] = new WiFiClientSecure();
        espClients[i]->setInsecure();
        mqttClients[i] = new PubSubClient(*espClients[i]);
        mqttClients[i]->setBufferSize(20480);
        mqttClients[i]->setServer(printerCfg[i].ip.c_str(), 8883);
        mqttClients[i]->setCallback(mqttCallback);
        mqttClients[i]->setKeepAlive(60);
    }

    server.on("/", HTTP_GET, []() {
        server.send_P(200, "text/html; charset=utf-8", mainHTML_P);
    });

    server.on("/status", HTTP_GET, []() {
        time_t localNow = time(nullptr);
        String j = "{\n\"printers\":[";
        for (int i = 0; i < printerCount; i++) {
            if (i) j += ",";
            auto& st = printerSt[i];
            String endTimeStr = "-";
            if ((st.state == "RUNNING" || st.state == "PAUSE") && st.remainMin > 0)
            {
                time_t endTs = localNow + (st.remainMin * 60);
                struct tm endTm; localtime_r(&endTs, &endTm);
                char tBuf[10];
                int localHour = (endTm.tm_hour + 3) % 24;
                snprintf(tBuf, sizeof(tBuf), "%02d:%02d", localHour, endTm.tm_min);
                endTimeStr = String(tBuf);
            }
            j += "{\n\"name\": \"" + printerCfg[i].name + "\",\n"
                "\n\"state\": \"" + st.state + "\",\n"
                "\n\"nozzle\": " + String(st.nozzle, 1) + ",\n"
                "\n\"bed\": " + String(st.bed, 1) + ",\n"
                "\n\"pct\": " + String(st.percent) + ",\n"
                "\n\"remain\": " + String(st.remainMin) + ",\n"
                "\n\"layer\": " + String(st.layerNum) + ",\n"
                "\n\"totalL\": " + String(st.totalLayer) + ",\n"
                "\n\"speed\": " + String(st.speedLevel) + ",\n"
                "\n\"end\": \"" + endTimeStr + "\",\n"
                "\n\"file\": \"" + st.file + "\"}";
        }
        j += "],\n"
            "\n\"qa\": " + String(queue.active ? "true" : "false") + ",\n"
            "\n\"qi\": " + String(queue.index) + ",\n"
            "\n\"qt\": " + String(queue.total) + ",\n"
            "\n\"esoc\": " + String(eco.soc) + ",\n"
            "\n\"etim\": " + String(eco.remainMin) + ",\n"
            "\n\"epow\": " + String(eco.outPower) + ",\n"
            "\n\"ecrit\": " + String(eco.critical ? "true" : "false") + ",\n"
            "\n\"tgSet\": " + (tgPassword.length() > 0 ? "true" : "false") + "}}";
        server.send(200, "application/json", j);
    });

    server.on("/set_tg_pass", HTTP_GET, []() {
        if (server.hasArg("p")) {
            tgPassword = server.arg("p");
        }
    });
}

```

					PE22.468332.001	Лис
						21
Зм.	Лис	№ докум.	Підпис	Дата		

```

        tgChatId    = "";
        lastUpdateId = 0;
        saveConfig();
    }
    server.send(200, "text/plain", "OK");
});

server.on("/set_speed", HTTP_GET, []() {
    int idx = server.arg("i").toInt();
    int lvl = server.arg("lvl").toInt();
    if (idx >= 0 && idx < printerCount) changeSpeed(idx, lvl);
    server.send(200, "text/plain", "OK");
});

server.on("/files", HTTP_GET, []() {
    String j = "[";
    for (int i = 0; i < sdFileCount; i++) { if (i) j += ","; j += "\"" +
sdFiles[i] + "\""; }
    j += "]";
    server.send(200, "application/json", j);
});

server.on("/add_file", HTTP_GET, []() {
    if (server.hasArg("name") && sdFileCount < MAX_FILES) {
sdFiles[sdFileCount++] = server.arg("name"); saveFiles(); }
    server.send(200, "text/plain", "OK");
});

server.on("/del_file", HTTP_GET, []() {
    int id = server.arg("id").toInt();
    if (id >= 0 && id < sdFileCount) {
        for (int i = id; i < sdFileCount - 1; i++) sdFiles[i] = sdFiles[i+1];
        sdFileCount--; saveFiles();
    }
    server.send(200, "text/plain", "OK");
});

server.on("/add_printer", HTTP_GET, []() {
    int i = printerCount;
    String ip = server.arg("ip"); ip.replace("https://", "");
ip.replace("http://", ""); ip.trim();
    printerCfg[i] = { server.arg("name"), ip, server.arg("pass"),
server.arg("sn") };
    printerCount++;
    saveConfig();
    espClients[i] = new WiFiClientSecure(); espClients[i]->setInsecure();
    mqttClients[i] = new PubSubClient(*espClients[i]);
    mqttClients[i]->setBufferSize(20480);
    mqttClients[i]->setServer(printerCfg[i].ip.c_str(), 8883);
    mqttClients[i]->setCallback(mqttCallback);
    mqttClients[i]->setKeepAlive(60);
    server.send(200, "text/plain", "OK");
});

server.on("/del_printer", HTTP_GET, []() {
    int id = server.arg("id").toInt();
    if (id >= 0 && id < printerCount) {
        delete espClients[id]; delete mqttClients[id];
        for (int i = id; i < printerCount - 1; i++) {
            printerCfg[i]    = printerCfg[i+1];
            printerSt[i]     = printerSt[i+1];
            espClients[i]    = espClients[i+1];
            mqttClients[i]   = mqttClients[i+1];
        }
        printerCount--;
    }
});

```

					PE22.468332.001	Лис
						22
Зм.	Лис	№ докум.	Підпис	Дата		

```

        saveConfig();
    }
    server.send(200, "text/plain", "OK");
});

server.on("/save_eco", HTTP_GET, [] () {
    eco.sn      = server.arg("sn");
    eco.appKey  = server.arg("key");
    eco.secret  = server.arg("sec");
    eco.pauseAt = server.arg("pause").toInt();
    saveConfig();
    server.send(200, "text/plain", "OK");
});

server.on("/set_eco_pause", HTTP_GET, [] () {
    eco.pauseAt = server.arg("v").toInt();
    prefs.begin("cfg", false); prefs.putInt("ePause", eco.pauseAt);
    prefs.end();
    server.send(200, "text/plain", "OK");
});

server.on("/home", HTTP_GET, [] () {
    int i = server.arg("i").toInt();
    if (i >= 0 && i < printerCount) {
        String topic = "device/" + printerCfg[i].serial + "/request";
        mqttClients[i]->publish(topic.c_str(),
"{"print\":"command\":"gcode_line\","param\":"G28\","sequence_id\":"3000"}");
    }
    server.send(200, "text/plain", "OK");
});

server.on("/start_queue", HTTP_GET, [] () {
    queue.printerIdx = server.arg("pi").toInt();
    String qs = server.arg("q");
    queue.total = 0;
    int si = 0, ci = qs.indexOf(',');
    while (ci != -1 && queue.total < QUEUE_MAX) {
        queue.items[queue.total++] = qs.substring(si, ci);
        si = ci + 1; ci = qs.indexOf(',', si);
    }
    if (si < (int)qs.length() && queue.total < QUEUE_MAX)
        queue.items[queue.total++] = qs.substring(si);
    queue.active = true;
    queue.index = 0;
    queue.waiting = false;
    processNextInQueue();
    server.send(200, "text/plain", "OK");
});

xTaskCreatePinnedToCore(TaskEcoFlow, "Eco", 12288, NULL, 1, NULL, 0);
xTaskCreatePinnedToCore(TaskTelegram, "TG", 18432, NULL, 1, NULL, 0);
xTaskCreatePinnedToCore(TaskMQTT, "MQTT", 10240, NULL, 1, NULL, 1);

server.begin();
Serial.println("[System] Normal mode started!");
Serial.println("[System] IP: " + WiFi.localIP().toString());
}

//Setup Mode
void startSetupMode() {
    Serial.println("[Setup] AP mode: " AP_SSID);
    WiFi.mode(WIFI_AP_STA);
    WiFi.softAP(AP_SSID, AP_PASS);
    dns.start(53, "*", WiFi.softAPIP());
}

```

					PE22.468332.001	Лис
						23
Зм.	Лис	№ докум.	Підпис	Дата		

```

    server.on("/", HTTP_GET, []() { server.send(200, "text/html; charset=utf-8", buildSetupHTML()); });
    server.onNotFound([]() { server.sendHeader("Location", "/");
server.send(302, "text/plain", ""); });
    server.on("/wifi_save", HTTP_GET, []() {
        prefs.begin("wifi", false);
        prefs.putString("ssid", server.arg("ssid"));
        prefs.putString("pass", server.arg("pass"));
        prefs.end();
        server.send(200, "text/plain", "OK");
        WiFi.begin(server.arg("ssid").c_str(), server.arg("pass").c_str());
    });
    server.on("/wifi_ip", HTTP_GET, []() {
        if (WiFi.status() == WL_CONNECTED)
            server.send(200, "application/json", "{\"ip\":\"" +
WiFi.localIP().toString() + "\"}");
        else
            server.send(200, "application/json", "{\"ip\":\"\"}");
    });
    server.on("/wifi_confirm", HTTP_GET, []() {
        server.send(200, "text/plain", "OK");
        delay(500); ESP.restart();
    });
    server.begin();
}

//setup()/loop()
void setup() {
    Serial.begin(115200); delay(300);
    loadConfig();
    loadFiles();

    tgMutex = xSemaphoreCreateMutex();
    tgClient.setInsecure();

    prefs.begin("wifi", true);
    String wSsid = prefs.getString("ssid", "");
    String wPass = prefs.getString("pass", "");
    prefs.end();

    if (wSsid.isEmpty()) { setupMode = true; startSetupMode(); return; }

    WiFi.mode(WIFI_STA);
    WiFi.begin(wSsid.c_str(), wPass.c_str());
    Serial.print("[WiFi] Connecting to " + wSsid);
    while (WiFi.status() != WL_CONNECTED) { delay(500); Serial.print("."); }
    Serial.println("\n[WiFi] Connected! IP: " + WiFi.localIP().toString());

    if (MDNS.begin("bambu")) MDNS.addService("http", "tcp", 80);
    configTime(0, 0, "pool.ntp.org", "time.nist.gov");
    time_t now = time(nullptr);
    while (now < 16000000000) { delay(300); now = time(nullptr); }

    startNormalMode();
}

void loop() {
    if (setupMode) { dns.processNextRequest(); server.handleClient(); return; }
    server.handleClient();
}

//Setup HTML
String buildSetupHTML() {
    return F("<!DOCTYPE html><html><head><meta charset='UTF-8'>"

```

					PE22.468332.001	Лис
						24
Зм.	Лис	№ докум.	Підпис	Дата		

```

    "<meta name='viewport' content='width=device-width,initial-
scale=1'>"
    "<title>Bambu Farm Setup</title>"
    "<style>body{font-
family:monospace;background:#0a0e14;color:#d0dce8;padding:30px}"
    "input{width:100%;padding:10px;margin:8px
0;background:#111720;border:1px solid #1e2d40;color:#d0dce8;border-
radius:4px;font-family:monospace;font-size:16px}"
    "button{background:#1a5e30;color:#3ddc84;border:none;padding:12px
24px;border-radius:4px;font-family:monospace;font-
size:16px;cursor:pointer;margin-top:10px}"
    "h2{color:#00c8ff;margin-bottom:20px}"
    ".pwr-box{background:#111720;border:1px solid
#ff6b35;padding:15px;border-radius:6px;margin-top:25px;font-size:14px;line-
height:1.5}"
    ".pwr-box h4{color:#ff6b35;margin-bottom:5px;margin-
top:0}</style></head><body>"
    "<h2>&#9632; BAMBU FARM – Wi-Fi Setup</h2>"
    "<label>Мережа (SSID)</label>"
    "<input type='text' id='s' placeholder='MyWiFi'>"
    "<label>Пароль</label>"
    "<input type='password' id='p' placeholder='password'>"
    "<button id='b' onclick='sendWifi()'>Підключитись</button>"

    "<div class='pwr-box'>"
    "<h4>⚡ Важливо щодо живлення плати:</h4>"
    "• Підключайте контролер через якісний кабель Type-C.<br>"
    "• Використовуйте блок живлення 5V потужністю <b>не менше 1A
(1000mA)</b>.<br>"
    "• Уникайте живлення від слабких USB-портів старих ПК або роутерів
– через високе навантаження Wi-Fi та Telegram плата може
перезавантажуватись."
    "</div>"
    "<script>"
    "function sendWifi(){
    "const ssid=document.getElementById(\"s\").value;"
    "const pass=document.getElementById(\"p\").value;"
    "document.getElementById(\"b\").disabled=true;"
    "document.getElementById(\"b\").innerText=\"Підключення...\";"

    "fetch(\"/wifi_save?ssid=\"+encodeURIComponent(ssid)+\"&pass=\"+encodeURIComponent
    "onent(pass))"
    ".then(()=>{
    "let t=setInterval(()=>{
    "fetch(\"/wifi_ip\").then(r=>r.json()).then(d=>{
    "if(d.ip){
    "clearInterval(t);"
    "document.body.innerHTML=\"<h2 style='color:#3ddc84'>✓
    Підключено!</h2><p>IP-адреса: <b
    style='color:#00c8ff'>\"+d.ip+\"</b></p><p>Відкрийте у браузері: <a
    style='color:#00c8ff'
    href='http://\"+d.ip+\"'>http://\"+d.ip+\"</a></p><br><button
    onclick='fetch(\"\"/wifi_confirm\"')>Зберегти та
    перезавантажити</button>\";"
    "}"
    "});"
    "},2000);"
    "});"
    "}"
    "</script>"
    "</body></html>");
}

```

					PE22.468332.001	Лис
						25
Зм.	Лис	№ докум.	Підпис	Дата		