ZERVI LABS • DOKUMENTACJA TECHNICZNA

Harpia

Dokumentacja aplikacji PC

Konfiguracja • HMI • tryb Master • Modbus RTU

Krótki przewodnik po aplikacji, która spina konfigurację, podgląd procesu i diagnostykę komunikacji w projekcie Harpia.

WERSJA PUBLICZNA

Wydanie 1.0 • 23 sierpnia 2026

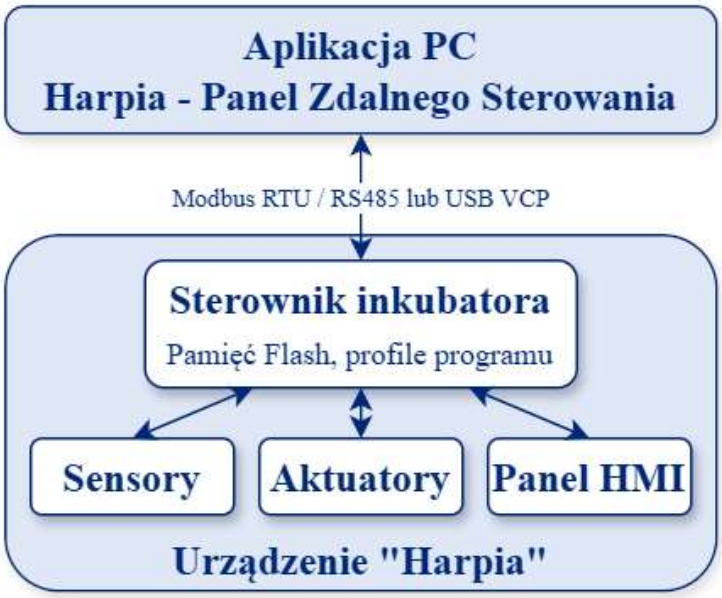
Zakres dokumentu

Warstwa PC projektu Harpia: założenia, architektura, konfiguracja profili, panel HMI, tryb serwisowy Master oraz organizacja komunikacji Modbus RTU.

1. Po co aplikacja PC?

Harpia składa się z trzech głównych warstw: aplikacji PC, oprogramowania sterownika STM32 oraz mechaniki urządzenia. Ten dokument opisuje wyłącznie pierwszą z nich. Aplikacja jest warstwą nadrzędną do ustawiania procesu, obserwacji pracy i diagnostyki — nie przejmuje sterowania czasu rzeczywistego.

> **Najważniejsza zasada:** po zapisaniu konfiguracji sterownik ma pracować samodzielnie. Komputer może zostać odłączony, a podstawowa regulacja i zabezpieczenia nadal pozostają aktywne.



Rys. 1. Rola aplikacji PC w systemie Harpia.

Dlaczego taki podział?

- Sterownik odpowiada za zadania krytyczne czasowo: regulację, harmonogram i reakcję na alarmy.
- Aplikacja PC zapewnia wygodny ekran, edycję profili, wykresy i narzędzia serwisowe.
- Awaria lub zamknięcie aplikacji nie może zatrzymać procesu prowadzonego przez sterownik.
- Oddzielenie warstw upraszcza testy: komunikację i interfejs można sprawdzać niezależnie od mechaniki.

2. Budowa aplikacji

Aplikację podzieliłem na warstwy, aby kod interfejsu nie mieszał się z obsługą ramek i portu szeregowego. Dzięki temu zmiana wyglądu panelu nie wymaga grzebania w protokole, a mapę danych można rozwijać bez przebudowy całego HMI.

Rys. 2. Warstwowa architektura aplikacji PC.

Główne moduły

- **main.py** — uruchomienie programu, organizacja okna i zakładek, połączenie oraz centralny stan telemetrii.
- **ui.py** — widoki ustawień komunikacji, konfiguracji, HMI i narzędzi serwisowych.
- **modbus_handler.py** — mapa danych, skalowanie, CRC, odczyt i zapis bloków oraz interpretacja odpowiedzi.

Technologie

Aplikacja powstaje w Pythonie. Kivy obsługuje interfejs, pyserial komunikację szeregową, Matplotlib wykresy, a profile są przechowywane lokalnie w formacie JSON. Zestaw jest celowo prosty: łatwo go rozwijać i diagnozować bez dokładania ciężkiej infrastruktury.

3. Ekrany i typowy sposób pracy

Interfejs rozdziela normalną obsługę od funkcji serwisowych. To ważne, bo ręczny zapis rejestru jest przydatny podczas uruchamiania urządzenia, ale na co dzień użytkownik powinien widzieć przede wszystkim konfigurację i czytelny stan procesu.

Obszar	Do czego służy	Kiedy jest używany
Ustawienia połączenia	Port COM, prędkość, parzystość, bity stopu i timeout.	Przy pierwszym uruchomieniu lub zmianie interfejsu.
Master	Ręczne odczyty, zapisy i testy pojedynczych pól mapy.	Uruchamianie, serwis i diagnostyka.
Konfiguracja	Profile procesu, parametry ogólne i wartości dla kolejnych dni.	Przy przygotowaniu lub zmianie cyklu.
HMI	Bieżące pomiary, stany, alarmy oraz START/PAUZA/WZNÓW/STOP.	Podczas obserwacji pracy urządzenia.
Wykresy i log	Trendy oraz komunikaty z odczytów, zapisów i błędów.	Analiza przebiegu i szukanie przyczyn problemu.

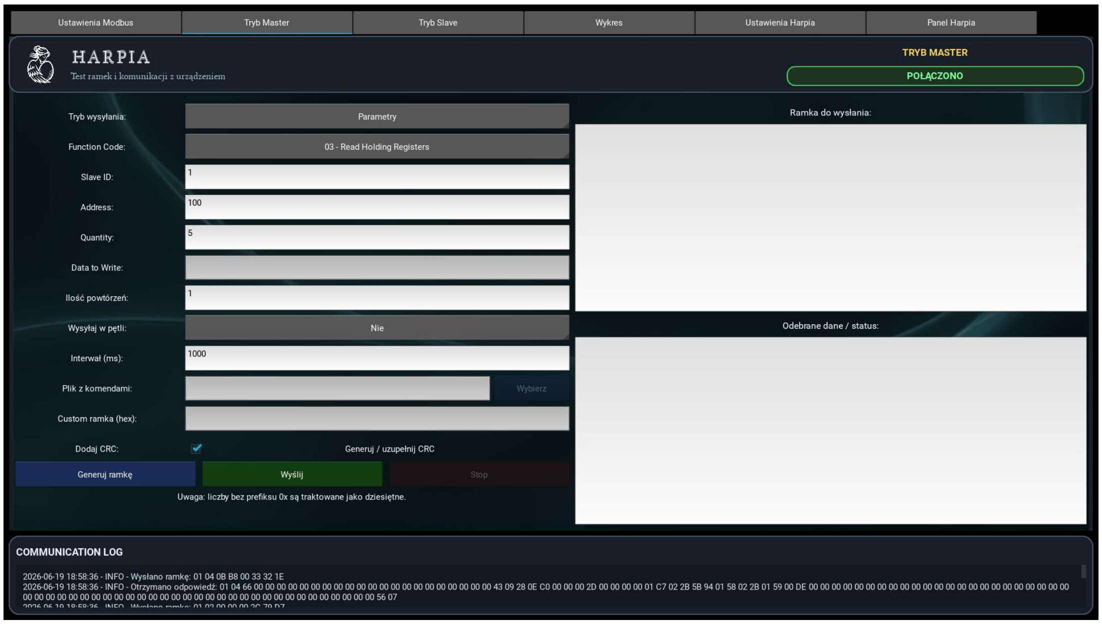
Połączenie z urządzeniem

1. Wybranie portu COM i parametrów zgodnych z interfejsem urządzenia.
2. Otworzenie połączenia i sprawdzenie, czy aplikacja otrzymuje poprawne odpowiedzi.
3. Dopiero wtedy odczytanie konfiguracji, wysłanie nowych profili lub uruchomienie podgląd HMI.

Bez blokowania interfejsu: wymiana danych działa poza główną pętlą GUI, a dostęp do portu jest synchronizowany. Brak odpowiedzi nie powinien więc zamienić aplikacji w nieruchomy obrazek.

4. Master — tryb serwisowy

Zakładka Master jest warsztatem do uruchamiania i sprawdzania komunikacji. Pozwala zobaczyć, co naprawdę dzieje się pomiędzy aplikacją a sterownikiem, bez przechodzenia przez wszystkie formularze HMI. Przydaje się wtedy, gdy trzeba odpowiedzieć na proste pytanie: czy konkretne pole mapy działa tak, jak powinno?



Rys. 3. Widok ręcznych operacji Modbus w zakładce Master.

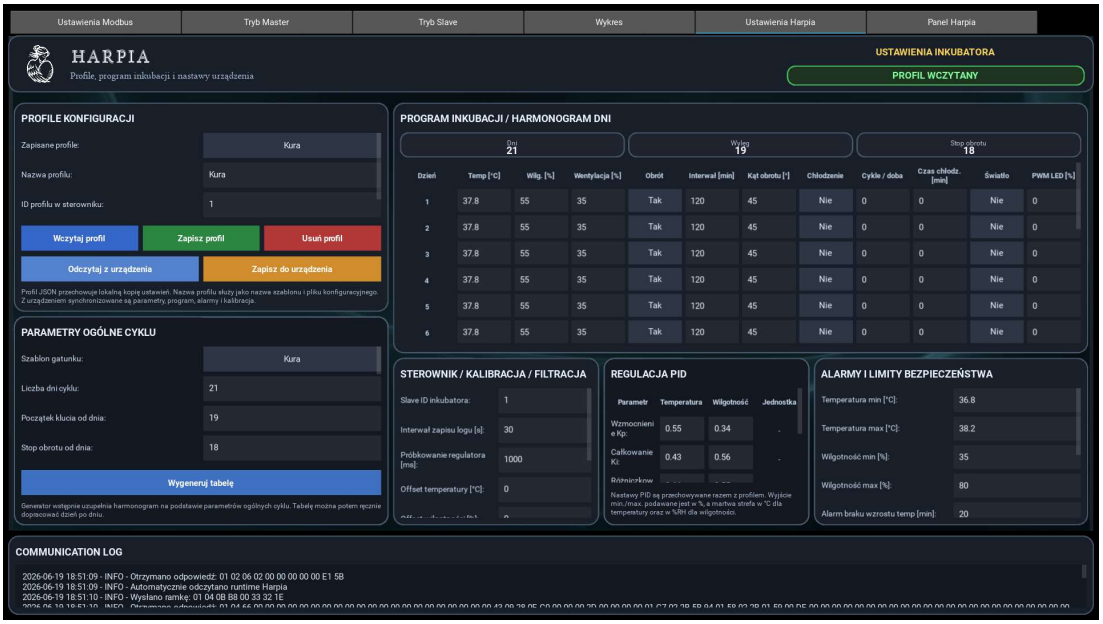
Co można sprawdzić

- odczyt pojedynczego rejestru, flagi lub niewielkiego bloku danych;
- zapis wartości konfiguracyjnej i późniejszy odczyt kontrolny;
- działanie komend i pól udostępnionych do testów serwisowych;
- poprawność adresu, typu funkcji, skalowania, CRC i odpowiedzi urządzenia.

To narzędzie świadomie nie jest „ładnym HMI”: ma być precyzyjne i przewidywalne. Ręczny zapis omija część zabezpieczeń zwykłego formularza, dlatego powinien być używany z wiedzą o mapie rejestrów.

5. Konfiguracja i profile procesu

Konfigurację podzieliłem na ustawienia ogólne oraz wartości zależne od dnia cyklu. Taki układ pozwala szybko zmienić parametry całego procesu, a jednocześnie zachować dokładny harmonogram temperatury, wilgotności, wentylacji czy obracania.



Rys. 4. Edycja konfiguracji i profilu procesu w aplikacji PC.

Zakres ustawień

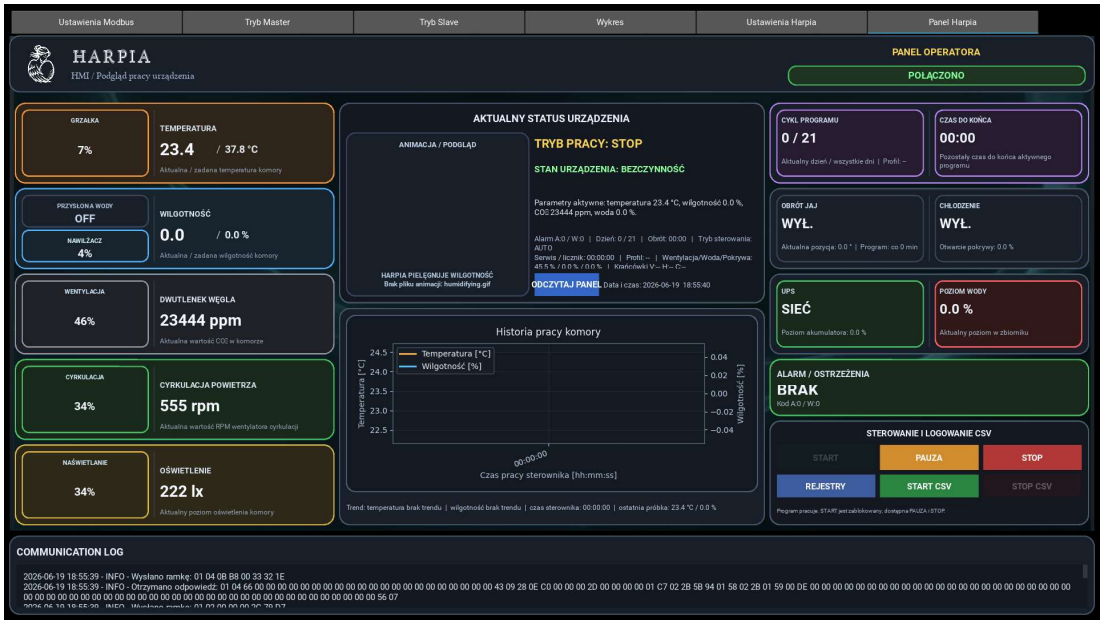
- profil i jego identyfikator, czas trwania cyklu oraz momenty zmiany kolejnych faz;
- temperatura i wilgotność zadana, wentylacja, cyrkulacja oraz chłodzenie;
- obracanie: interwał, kąt lub czas pracy oraz dzień zakończenia;
- oświetlenie, kalibracja, progi ostrzeżeń i alarmów.

Zapis, odczyt i walidacja

Profil można zapisać lokalnie w JSON, ponownie wczytać, wysłać do sterownika albo odczytać z urządzenia. Przed transmisją aplikacja kontroluje m.in. zakres dni, długości list, temperaturę, wilgotność, parametry obracania i progi alarmowe. Wartości są prezentowane w jednostkach inżynierskich, a skalowanie do liczb protokołu odbywa się w warstwie komunikacji.

6. Panel HMI

HMI jest ekranem do codziennego podglądu. Zamiast surowych adresów pokazuje wartości, stany i komunikaty w formie, którą da się szybko odczytać. Operator nie powinien analizować ramek, żeby zauważyć zbyt wysoką temperaturę albo brak wody.



Rys. 5. Panel HMI aplikacji PC — widok z danych symulacyjnych.

Najważniejsze informacje

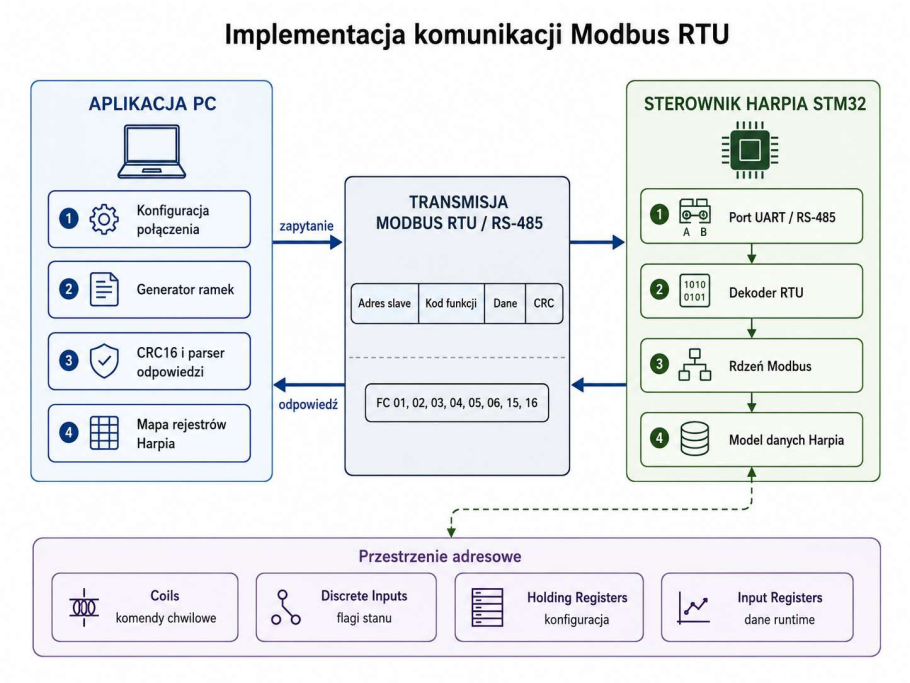
- temperatura i wilgotność w komorze oraz odpowiadające im wartości zadane;
- stany grzałki, nawilżania, przepustnicy wody, wentylacji i cyrkulacji;
- oświetlenie, obracanie, chłodzenie, poziom wody i stan zasilania awaryjnego;
- alarmy i ostrzeżenia z czytelnym priorytetem;
- komendy START, PAUZA/WZNÓW oraz STOP.

Wykresy

Trendy temperatury i wilgotności są oparte na czasie pracy raportowanym przez sterownik, a nie wyłącznie na zegarze komputera. Dzięki temu oś czasu pozostaje związana z rzeczywistym przebiegiem procesu także po chwilowym rozłączeniu aplikacji.

7. Organizacja danych Modbus RTU

Aplikacja inicjuje wymianę danych i tłumaczy te dane pomiędzy ekranem użytkownika a mapą protokołu. Adresy i skalowanie są skupione w jednej warstwie, aby interfejs operował nazwami takimi jak „temperatura zadana”, a nie rozsianymi po kodzie liczbami rejestrów.



Rys. 6. Przepływ danych i odpowiedzialności w obsłudze Modbus RTU.

Podział mapy

- rejestry konfiguracji i profili — parametry zapisywane przed lub w trakcie przygotowania cyklu;
- rejestry danych bieżących — pomiary, wartości zadane, czas pracy i liczby diagnostyczne;
- pola komend — żądania START, PAUZA/WZNÓW i STOP;
- flagi stanu — aktywne funkcje, ostrzeżenia oraz alarmy.

Sposób wymiany

Odczyty są łączone w bloki, żeby nie zasypywać magistrali drobnymi żądaniami. Przy profilu pobierany jest tylko używany zakres dni. Po zapisie aplikacja może wykonać odczyt kontrolny i porównać wynik, co szybko ujawnia zły adres, skalowanie lub nieprzyjętą wartość.

> **Pełne adresy pozostają osobno:** szczegółowa mapa rejestrów jest publikowana jako oddzielny PDF. Ten dokument wyjaśnia architekturę i sposób pracy aplikacji, bez powielania kilkudziesięciu tabel adresowych.

8. Odporność, logowanie i testy

Komunikacja szeregowa potrafi zgubić odpowiedź, zwrócić uszkodzoną ramkę albo zostać przerwana w najmniej odpowiednim momencie. Dlatego aplikacja traktuje błędy jako normalny scenariusz diagnostyczny: raportuje problem, zwalnia port i pozwala ponowić operację.

Co zapisuje log

- otwarcie i zamknięcie połączenia oraz użyte parametry portu;
- wysłane żądania, odpowiedzi, błędy CRC i przekroczenia czasu;
- wyniki odczytów i zapisów oraz problemy wykryte przez walidację;
- informacje potrzebne do powtórzenia testu w zakładce Master.

Dotychczasowy zakres testów

Warstwę PC sprawdzałem na wirtualnej parze portów COM i symulatorze urządzenia Modbus. Testy objęły zestawienie połączenia, format ramek, mapę danych, skalowanie, walidację, aktualizację HMI oraz zachowanie przy braku odpowiedzi i błędnych danych. W oczekiwanych sytuacjach awaryjnych interfejs pozostawał responsywny.

> **Granica obecnej weryfikacji:** wyniki z symulatora potwierdzają logikę aplikacji i protokołu, ale nie zastępują testów z docelowym sterownikiem STM32, fizycznym RS485, czujnikami i mechaniką urządzenia.

9. Podsumowanie

Aplikacja PC pełni rolę wygodnego pulpitu nad autonomicznym sterownikiem: przygotowuje profile, pokazuje stan procesu i daje mi precyzyjne narzędzia serwisowe. Najważniejsze elementy — konfiguracja, HMI, Master oraz obsługa Modbus — są rozdzielone, ale korzystają ze wspólnego modelu danych. Kolejnym krokiem będzie potwierdzenie tej warstwy na rzeczywistym sprzęcie i dopracowanie ekranów na podstawie wyników prób całego urządzenia.