ZPi-IO08R Controller

사용설명서 (Ver. 3.0)

제품명 - **ZPi-IO08R 4DI / 4DO Programmable Module**

2026.05 고잉

목 차

1. 시작하기

ZPi-IO08R Controller는 라즈베리파이 Zero 2 W 위에서 동작하는 4DI / 4DO 프로그래머블 모듈입니다. **매트릭스 · 자연어 카드 · Python 스크립트** 세 가지 방법으로 로직을 작성하고, 추가로 **사용자 프로그램(Python / Node.js / .NET / Native 바이너리)**을 .zip 으로 업로드해 서비스로 등록할 수 있으며, **Claude Desktop MCP**로 자연어 제어도 가능합니다.

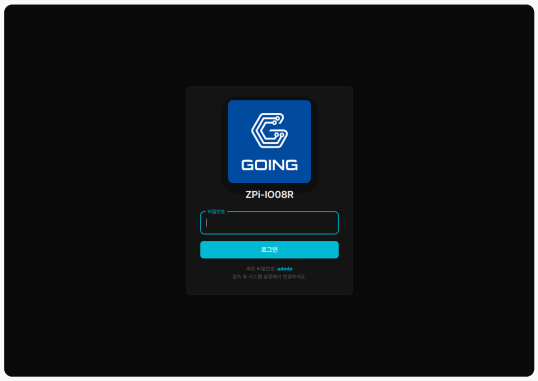
1-1 라즈베리파이 IP 확인

전원 인가 시 OLED 첫 줄에 IP:포트가 표시됩니다.

※ 첫 줄은 시스템이 잠그고 있어 사용자가 변경할 수 없습니다 (5장 참조).

1-2 웹 접속 · 로그인

1. 브라우저로 `http://<장비 IP>:5000/` 또는 `http://going.local:5000/` 접속.
2. 로그인 화면에 **기본 비밀번호 admin** 입력 → 접속.



▲ */app/login — GOING 로고 + ZPi-IO08R 표기 + 비밀번호 입력*

8시간 후 또는 60분 무활동 시 세션 만료 → 다시 로그인 필요.

무차별 대입 보호: 같은 IP 에서 1분 내 10회 비밀번호 실패 시 1분간 잠금됩니다. 실패 시 잔여 시도 횟수와 잠금 시 남은 초가 응답에 표시됩니다.

1-3 첫 비밀번호 변경 (보안)

로그인 후 시스템 메뉴에 들어가면 **"기본 비밀번호 admin 사용 중"** 노란 경고가 뜹니다. 비밀번호 변경 폼에서 새 비밀번호(4자 이상)를 설정하세요. PBKDF2-SHA256으로 해싱되어 저장되며 평문은 어디에도 남지 않습니다.

1-4 변수 풀 (IN / OUT / M / T / C / D)

매트릭스 · 카드 · 스크립트는 모두 같은 변수 풀을 공유. 한 곳에서 변경한 값을 다른 곳에서 즉시 읽을 수 있습니다.

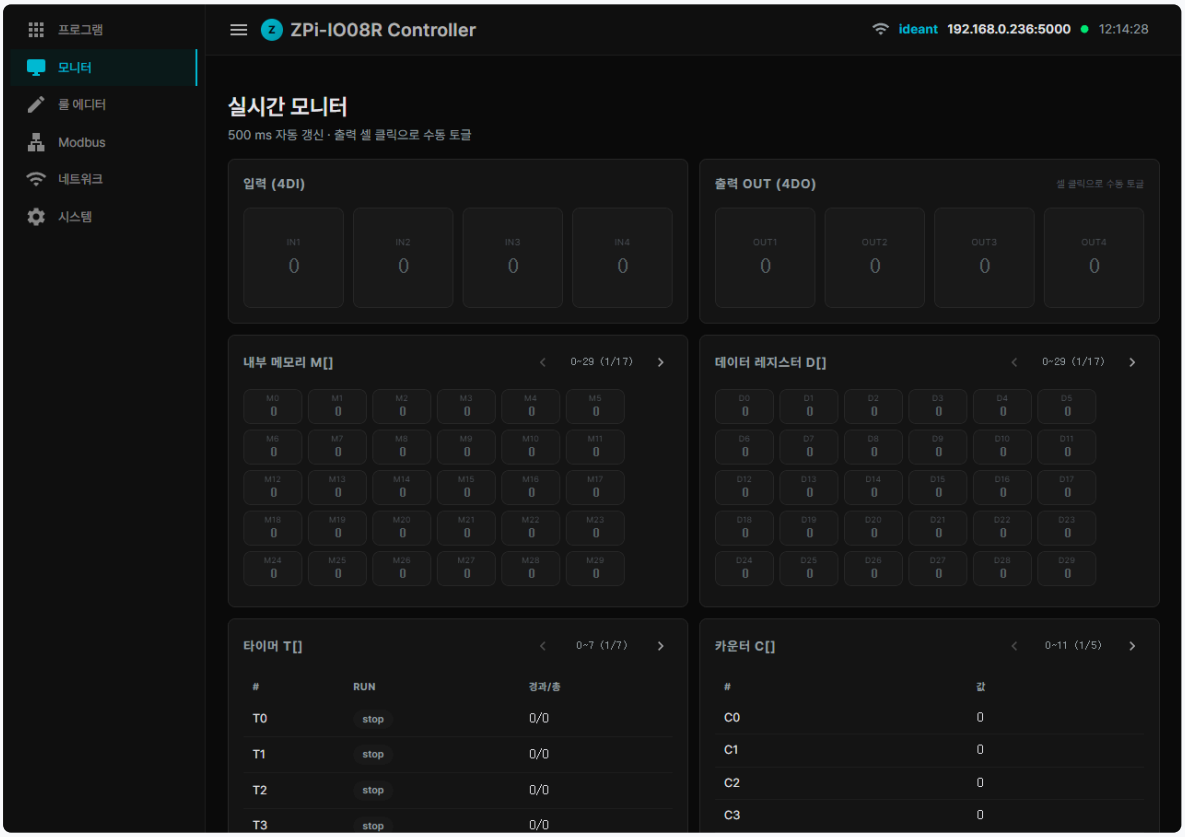
이름	형식	개수	설명
`IN[i]`	bool (R/O)	4	보드 디지털 입력
`OUT[i]`	bool (R/W)	4	보드 디지털 출력 (Modbus 메모리 맵상 P 영역)
`M[i]`	bool (R/W)	500	내부 메모리 비트
`T[i]`	Timer	50	타이머 (Elapsed / Running / DurationMs / ElapsedMs)
`C[i]`	int (R/W)	50	카운터
`D[i]`	ushort (R/W)	500	데이터 레지스터 = Modbus Holding
OLED slot	string	3	OLED 사용자 제어 영역 (5장)

2. 실시간 모니터

⚠ 사용자 프로그램 실행 중에는 비활성화됩니다. 프로그램 메뉴에서 사용자 프로그램(.NET / Node.js / Python / Native) 이 실행되면 내장 Controller 엔진이 GPIO 를 양도하기 위해 자동으로 일시정지하고 좌측 네비메뉴에서 모니터 · 룰 에디터 · Modbus 가 함께 숨겨집니다. 사용자 프로그램이 모두 멈춘 뒤 프로그램 페이지의 [PLC 엔진 다시 시작] 버튼을 누르면 다시 표시됩니다 (9-4, 9-5 참조).

2-1 페이지 구성

2×3 그리드 카드 6개로 구성, 500ms마다 자동 갱신:



▲ 입력/출력 + 메모리/D + 타이머/카운터 카드 6개

카드	표시 내용
입력 (4DI)	IN1~IN4 비트 (시안 = ON)
출력 OUT (4DO)	OUT1~OUT4 (셀 클릭 시 수동 토글)
내부 메모리 M[]	스크롤 가능, 30개씩 페이지네이션
데이터 레지스터 D[]	24개씩 페이지네이션, 0이 아닌 값만 강조
타이머 T[]	RUN 상태 + 경과/총
카운터 C[]	현재 카운트 값

2-2 출력 수동 토글 (디버그용)

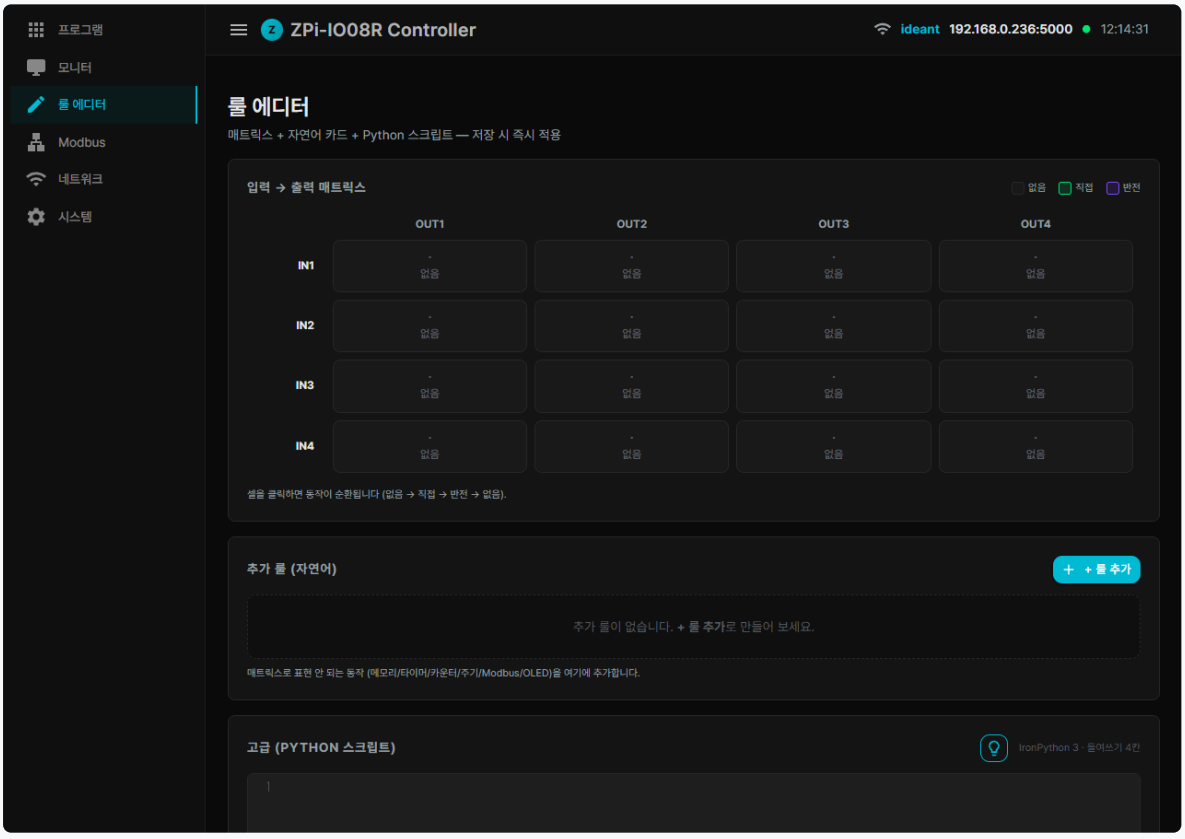
출력 카드의 OUT1~OUT4 셀을 클릭하면 즉시 GPIO가 토글됩니다 (POST /api/output/{idx}/{state}). 단, 룰 엔진이 같은 출력을 제어 중이면 다음 50ms 스캔에서 룰이 덮어씁니다.

3. 룰 에디터

> ⚠ **사용자 프로그램 실행 중에는 비활성화됩니다. 사용자 프로그램이 GPIO / Modbus / 시리얼 포트를 잡는 동안 룰 엔진이 충돌 없이 자동 일시정지되며, 여기서 작성/저장한 매트릭스·카드·Python 스크립트는 사용자 프로그램이 모두 멈추고 [PLC 엔진 다시 시작] 을 눌러야 다시 동작합니다 (9-4, 9-5 참조).**

3-1 매트릭스 (가장 단순)

상단 4×4 표의 셀을 클릭하면 **없음 → 직접 → 반전 → 없음** 순환합니다. 화면 하단의 **"저장 (즉시 적용)"** 버튼을 누르면 라즈베리파이에 저장되고 50ms 안에 적용됩니다.



▲ */app/editor — 매트릭스 + 자연어 카드 + Python 스크립트 한 화면*

모드	색상	동작
없음	회색	아무 동작 없음
직접	녹색 ↦	OUT[j] = IN[i]
반전	보라 ↮	OUT[j] = !IN[i]

3-2 자연어 카드 룰

1. "추가 룰" 카드의 **+ 룰 추가** 클릭.
2. 좌측(WHEN) 드롭다운에서 트리거 선택 → 인덱스/값 입력.
3. 우측(DO) 드롭다운에서 액션 선택 → 인덱스/값/(텍스트) 입력.
4. 우측 끝 텍스트칸에 룰 설명 적기.
5. **저장**으로 즉시 적용. **사용** 체크박스로 비활성화 가능.

3-3 트리거·액션 일람

트리거 그룹	옵션
입력 IN	IN 켜질 때 / 꺼질 때 / 변할 때
메모리 M	M 켜질 때 / 꺼질 때 / 변할 때
카운터 C	C 값 = X 도달 순간
데이터 D	D 값 = X 도달 순간 (Modbus 외부 쓰기 포함)
타이머 T	T 종료될 때
주기	매 X ms 마다 (최소 50ms)

액션 그룹	옵션
출력 OUT	OUT 켜기 / 끄기 / 토글
메모리 M	M 켜기 / 끄기 / 토글
타이머 T	T X ms 시작 / 정지
카운터 C	C +1 / = 0
데이터 D	D = X 쓰기
OLED	OLED 줄 X 표시 (텍스트 입력) / 지우기

3-4 활용 예시

예 1) IN1을 누르면 OUT1 5초 ON 후 자동 OFF

WHEN	DO
IN 켜질 때 / index=0	OUT 켜기 / index=0
IN 켜질 때 / index=0	T 5000ms 시작 / index=0
T 종료될 때 / index=0	OUT 끄기 / index=0

4. Python 스크립트 (IronPython 3)

룰 에디터의 **고급 (Python 스크립트)** 영역에서 Python 코드를 작성합니다. 실행 환경은 .NET 위의 **IronPython 3 (Python 3.4 호환)**이며, 인-프로세스로 호출되어 50 ms 스캔 주기에 충분히 빠릅니다. NumPy/Pandas 등 C 확장 라이브러리는 사용 불가지만 `math`, `datetime`, `json`, `re` 같은 순수 Python 표준 라이브러리는 모두 사용 가능합니다.

4-1 사용 가능 변수와 함수

심볼	설명
`In[i]` / `Out[i]` / `P[i]`	입출력 (4개) — Out 과 P 는 같은 별칭
`M[i]` / `D[i]` / `C[i]`	메모리(500) / 데이터레지스터(500) / 카운터(50)
`T[i].Elapsed`	해당 스캔에 타이머 만료된 경우 **한 번만** True (4-4 참조)
`T[i].Running` / `T[i].DurationMs` / `T[i].ElapsedMs`	타이머 상태 / 설정 / 경과
`StartTimer(i, ms)` / `StopTimer(i)`	타이머 시작 / 정지 (snake_case 별칭: `start_timer` / `stop_timer`)
`IncCounter(i)` / `ResetCounter(i)`	카운터 ++ / 리셋
`Oled.SetLine(0..2, text)`	OLED 슬롯에 텍스트 (5장 참조)
`Oled.ClearLine(i)` / `Oled.Clear()`	슬롯 / 전체 기본값 복귀
`Log("...")`	journalctl 로그 (디버깅용)

4-2 진입점 함수

함수	호출 시점
`def Tick():`	**매 50 ms 호출** — 메인 로직
`def OnInput(idx, on):`	입력 에지마다 호출 (선택, 이벤트 기반)

> 함수명은 PascalCase (`Tick` , `OnInput`) 또는 snake_case (`tick` , `on_input`) 둘 다 인식. PEP 8 기준이라면 snake_case 가 더 자연스럽습니다.

4-3 예시 불러오기

고급 카드 우측 상단의 **예시 불러오기** 메뉴에서 10가지 예시를 즉시 코드 영역에 채울 수 있습니다.

#	예시
1	단순 인터록 (IN0 → OUT0)
2	5초 후 자동 OFF
3	입력 펄스 카운트 (10번이면 OUT2 ON)
4	Modbus D[10] 임계값 알람
5	1초 카운터 + OLED 표시
6	모든 입력 AND
7	Latch (M0 메모리)
8	사인 파를 D[20] 에 출력 (math 모듈)
9	깜빡이 (1초 ON / 1초 OFF)
10	비상 정지 (IN3 = E-Stop)

※ 작성 중인 코드가 있으면 교체 전 확인 다이얼로그가 뜹니다.

4-4 타이머 사용 시 주의 (one-scan pulse)

`T[i].Elapsed` 는 타이머가 만료되는 **딱 한 스캔(50 ms)만 True**이며 다음 스캔에서 자동 클리어됩니다. 또한 만료 시 `T[i].Running` 도 False 가 됩니다. 그래서 다음 패턴은 **동작하지 않습니다**:

```
# ❌ 잘못된 예 — 토글이 한 번도 안 됨
def Tick():
    if not T[0].Running:
        StartTimer(0, 1000)        # ← 이게 먼저 실행되어 Elapsed 를 False 로 리셋
    if T[0].Elapsed:               # ← 이미 False
        Out[0] = not Out[0]
        StartTimer(0, 1000)
```

해결책 — **Elapsed 체크가 먼저 오도록 elif 사용**:

```
# ✅ 올바른 깜빡이
def Tick():
    if T[0].Elapsed:
        Out[0] = not Out[0]
        StartTimer(0, 1000)
    elif not T[0].Running:
        StartTimer(0, 1000)        # 부팅 시 1회 시작
```

4-5 활용 예시

예) OUT0 1초 OFF / 5초 ON 무한 반복

```
def Tick():
    # 부팅 시 1회 시작 (T0 = 1초, OFF 페이즈)
    if not T[0].Running and not T[1].Running:
        StartTimer(0, 1000)
    if T[0].Elapsed:
        Out[0] = True
        StartTimer(1, 5000)
    if T[1].Elapsed:
        Out[0] = False
        StartTimer(0, 1000)
```

예) Modbus D[10] 임계값 + OLED 알람

```
def Tick():
    if D[10] >= 100:
        Out[3] = True
        Oled.SetLine(0, 'ALARM: D10=' + str(D[10]))
    else:
        Out[3] = False
        Oled.ClearLine(0)
```

저장 시 라즈베리파이가 IronPython 으로 컴파일 (첫 워밍업 ~4.5초, 이후 수백 ms). 구문 오류가 있으면 `line N, col M: ...` 형식으로 표시되고 이전 정상 스크립트가 그대로 동작합니다.

5. OLED 사용자 제어

128×64 SSD1306 OLED를 사용자 로직에서 직접 제어 가능. **첫 줄(IP:포트)은 잠겨** 있고 **나머지 3줄(slot 0/1/2)**은 자유롭게 표시할 수 있습니다.

5-1 화면 구조

```
192.168.0.236:5000   ← page 0-1 잠김
HH:MM:SS               ← slot 0 (비우면 시계)
IN  1010               ← slot 1 (비우면 IN)
OUT 0011               ← slot 2 (비우면 OUT)
```

※ 한 슬롯에 표시 가능한 글자 수: 약 10자 (2× 폰트 기준)

5-2 카드 룰로 표시

예) IN1 켜지면 slot 1에 "ALARM" 표시, 꺼지면 기본값 복귀

WHEN	DO
IN 켜질 때 / index=0	OLED 줄 X 표시 / index=1 / text="ALARM"
IN 꺼질 때 / index=0	OLED 줄 X 지우기 / index=1

5-3 Python 코드로 표시

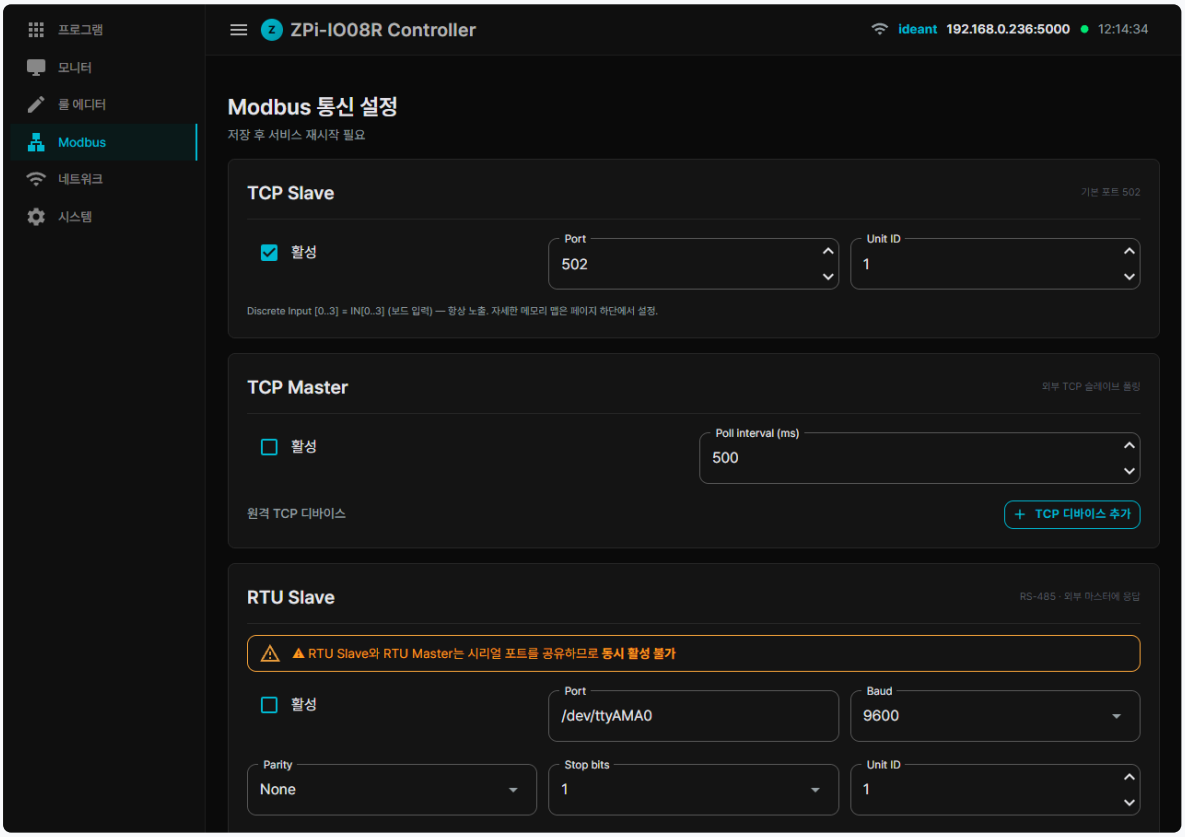
```
import datetime

def Tick():
    Oled.SetLine(0, 'VAL ' + str(D[10]))                    # slot 0: D[10] 실시간
    Oled.SetLine(1, datetime.datetime.now().strftime('%H:%M:%S'))  # slot 1: 시계
```

같은 슬롯에 카드 룰과 스크립트가 동시 쓰면 **마지막 호출이 이깁니다.** 스크립트 50ms 호출이 보통 카드 룰 OnTick 1초보다 자주 → 스크립트가 우세.

6. Modbus 통신

⚠ 사용자 프로그램 실행 중에는 모든 Modbus 채널이 자동 해제됩니다. TCP Slave (포트 502) / TCP Master / RTU Slave / RTU Master 모두 사용자 프로그램이 같은 자원(포트·시리얼)을 잡을 수 있도록 listener / serial 핸들을 닫습니다. 사용자 프로그램이 모두 멈추고 [PLC 엔진 다시 시작] 을 누르면 자동으로 재개됩니다 (9-4, 9-5 참조).

6-1 메모리 맵 (영역별 시작주소)



▲ */app/modbus — TCP/RTU × Slave/Master 4 카드 + 메모리 맵*

각 영역의 Modbus 시작 주소를 16진수로 설정 가능. Controller 표준 컨벤션 따라 P/M은 비트(Coil), T/C/D/WP/WM은 워드(Holding):

영역	기본	Modbus	용도
P	0x0000	Coil	OUT[0..3] 보드 출력 (R/W)
M	0x1000	Coil	M[0..499] 메모리 비트 (R/W)
T	0x5000	Holding	T[i].ElapsedMs/100 (R/O)
C	0x6000	Holding	C[i] 하위 16bit (R/W)
D	0x7000	Holding	D[0..499] (R/W)
WP	0xA000	Holding	P[0..15] 패킹 (R/O)
WM	0xB000	Holding	M[16i..16i+15] 패킹 32워드 (R/W)

※ Discrete Input 0..3 = IN[0..3]은 항상 base 0에 노출 (입력 컨벤션)

6-2 TCP Slave · Master

TCP Slave — 외부 SCADA/HMI 가 Controller를 슬레이브로 보고 메모리 P/M/T/C/D/WP/WM 모든 영역을 읽기/쓰기. 기본 포트 502, Unit ID 1.

TCP Master — 외부 TCP 슬레이브 디바이스를 폴링해서 D[] 에 매핑. 디바이스마다 Host:Port + Unit ID + Read/Write 항목 설정:

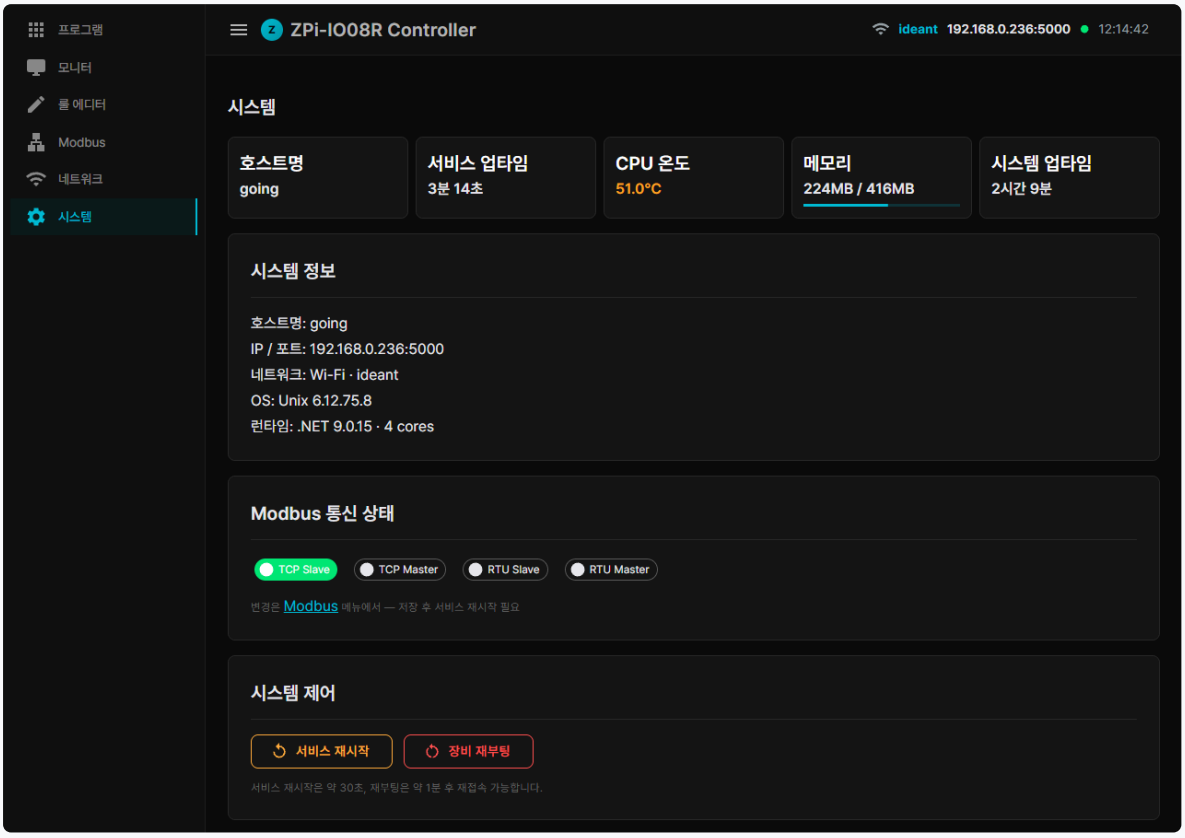
1. "+ TCP 디바이스 추가" → Host (IP), Port, Unit ID 입력.
2. "+ Read" 추가 → FC (3=holding/4=input/1=coil/2=discrete) + Start + Count + 저장할 D 인덱스.
3. "+ Write" 추가 → FC (5/6/15/16) + Start + Count + 보낼 D 인덱스.

6-3 RTU Slave · Master (RS-485)

RS-485 시리얼 포트(`/dev/ttyAMA0`)를 사용. 메모리 맵 / 디바이스 구조는 TCP 와 동일하며, 슬레이브는 외부 마스터에 응답하고 마스터는 외부 슬레이브를 Unit ID 로 폴링합니다.

RTU Slave 와 RTU Master 는 시리얼 포트 공유로 동시 활성 불가. UI 에서 노란 경고 표시.

7. 시스템 페이지



▲ */system.html — 정보 칩 6개 + 4개 섹션 카드*

7-1 상단 정보 칩 (6개)

칩	내용
호스트명	OS hostname (예: going)
서비스 업타임	Controller 서비스 시작 후 경과시간
CPU 온도	50°C↑ 노랑 / 70°C↑ 빨강 자동 강조
메모리	사용/전체 MB + 시안 프로그레스바
서비스 메모리	Controller process working set
시스템 업타임	OS 부팅 후 경과시간 (/proc/uptime)

7-2 Wi-Fi 변경

1. "다시 검색" 버튼으로 주변 SSID 스캔.
2. SSID 콤보박스에서 원하는 네트워크 선택 (현재 연결중은 자동 선택됨).
3. 비밀번호 입력 (OPEN 네트워크는 비밀번호칸 자동 숨김).
4. "연결 시도" 클릭 → 약 15-30초 대기 → 자동 적용.

> **안전 검증:** 새 네트워크에 IP를 받아 실제 접속 가능한 경우만 적용. 실패 시 NetworkManager가 자동으로 이전 네트워크로 복구.

> 숨김 SSID는 콤보박스 마지막 옵션 "— SSID 직접 입력 —"으로 추가 가능.

7-3 시스템 제어

버튼	동작
⟳ 서비스 재시작	Controller 프로세스만 재시작 (~30초). systemd가 자동 복구
↻ 장비 재부팅	라즈베리파이 전체 재부팅 (~1분). passwordless sudo 필요

7-4 비밀번호 변경

1. 현재 비밀번호 입력 (오타 시 진행 안 됨).
2. 새 비밀번호 + 확인 (4자 이상).
3. "변경" 클릭 → 모든 다른 세션 무효화 + 현재 세션은 자동 갱신 (끊김 없음).

8. Claude Desktop MCP 연동

Claude Desktop 채팅창에서 자연어로 Controller를 제어 가능 — "IN1 누르면 OUT1 켜는 룰 만들어줘" 같은 요청을 Claude가 이해하고 적절한 도구를 호출합니다.

8-1 설치 (Install.bat 한 번만)

1. 배포 폴더(예: `C:\Tools\ZpiPlc\`)에 5개 파일 복사 (Install.bat / Uninstall.bat / ZPi.PLC.Mcp.exe / install.ps1 / uninstall.ps1).
2. **Install.bat 더블클릭** → 라즈베리파이 IP 입력 → "Claude Desktop 재시작?" → 예.
3. 채팅창의 🔨 아이콘에 **zpi-plc**가 나타나면 완료.

8-2 자연어 사용 예시

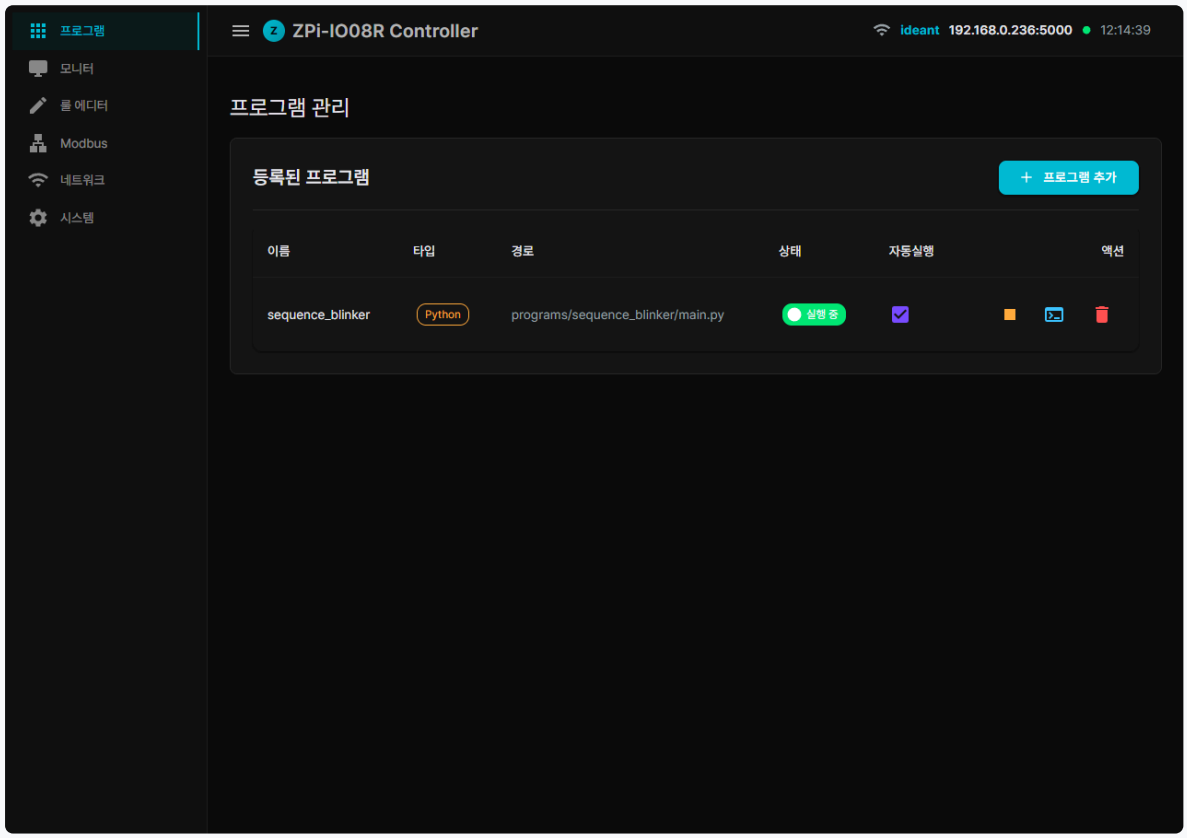
요청	Claude가 하는 일
"Controller 상태 보여줘"	plc_state로 IO/메모리/D 표 정리
"IN1 누르면 OUT1 켜는 룰 만들어줘"	plc_add_card_rule (OnInputRising → SetOutput)
"OUT0 1초 ON/5초 OFF 반복"	plc_set_script로 Tick + StartTimer 코드 생성
"OLED slot 1에 'Controller OK' 표시"	plc_add_card_rule (Always → WriteOledLine)
"D[10] 5초간 측정해줘"	plc_watch (samples=10, intervalMs=500)
"룰 다 지워줘"	plc_clear_rules

8-3 도구 일람

도구	용도
`plc_info`	변수 풀 크기, 트리거/액션 종류
`plc_state`	실시간 상태 (compact / verbose)
`plc_watch`	N회 샘플링 후 시간순 변화 반환
`plc_get_rules`	현재 매트릭스/카드/스크립트 조회
`plc_set_matrix`	4×4 매트릭스만 교체
`plc_add_card_rule`	카드 룰 한 장 추가 (text 파라미터로 OLED 지원)
`plc_set_script`	Python 스크립트만 교체 (가장 자주 사용)
`plc_clear_rules`	전체 초기화
`plc_set_output`	출력 한 개 강제 토글
`plc_get_modbus / save_modbus`	Modbus 설정 조회/저장

9. 사용자 프로그램 관리 (Programs)

매트릭스 / 카드 룰 / Python 스크립트로 부족한 복잡한 로직은 **독립적인 프로그램**을 **.zip**으로 업로드해 서비스로 등록할 수 있습니다. 좌측 네비게이션의 **프로그램** 메뉴에서 관리합니다.



▲ */app/programs — 등록된 프로그램 목록 + 추가/실행/중지/로그/삭제 액션*

9-1 개요

지원 프로젝트 타입 (압축 해제 시 자동 감지):

타입	감지 기준	실행 명령
.NET	`*.runtimeconfig.json`	`dotnet xxx.dll`
Node.js	`package.json`	`node xxx.js` (package.json "main")
Python	`main.py` / `app.py` / 첫 *.py	`python3 xxx.py`
Native	실행 파일 또는 `*.sh`	해당 파일 직접 실행

Pi에는 **Node.js 20.19, npm 9.2, Python 3.13**이 미리 설치되어 있어 별도 설치 불필요.

9-2 프로그램 설치

1. **프로그램 추가** 버튼 클릭.
2. ZIP 파일 선택 (최대 100 MB) → 표시 이름 입력 → **설치**.
3. 자동으로 압축 해제 → 프로젝트 타입 감지 → ELF/`.sh` 파일에 +x 권한 부여 → 목록에 등록.

> 실행 파일명 칸은 비워두면 자동 감지를 사용합니다. 자동 감지가 잘못된 경우만 입력하세요.

> **Native 바이너리는 linux-arm64 빌드여야 합니다. 잘못된 아키텍처는 즉시 실행 실패.**

9-3 실행 · 중지 · 자동실행

각 프로그램 행에 다음 액션이 있습니다:

아이콘	동작
▶ (녹색)	프로그램 시작 (이미 실행 중이면 ■ 로 바뀜)
■ (주황)	프로그램 중지 (SIGTERM → 2초 대기 → SIGKILL)
로그	표준 출력 / 에러 최근 200줄 표시
삭제	중지 후 폴더 + 항목 제거 (확인 다이얼로그)

자동실행 토글: ON 으로 두면 ZPi 서비스 부팅 시 함께 시작합니다 (별도 systemd 등록 불필요).

9-4 Controller 엔진 자동 일시정지 (충돌 보호)

사용자 프로그램이 GPIO 핀 / Modbus 502 포트 / RS-485 시리얼 포트를 직접 사용하면 내장 Controller 엔진과 충돌할 수 있습니다. 충돌을 막기 위해 **임의의 사용자 프로그램이 실행되면 Controller 엔진이 자동으로 일시정지**합니다.

프로그램 상태	Controller 엔진	네비메뉴
없음 (또는 모두 정지)	ON (수동 재개 후)	모니터 / 룰에디터 / Modbus 표시
1개 이상 실행 중	OFF — GPIO/I²C/포트 502/시리얼 모두 해제	해당 메뉴 자동 숨김

※ 정지 시 출력은 모두 LOW 로 떨어진 뒤 핀이 해제됩니다 (브리지 릴레이 순간 동작 방지).

9-5 안전: 수동 재개 정책

테스트 중인 사용자 프로그램이 예기치 않게 종료되었을 때 옛 룰이 자동으로 깨어나 현장 장비가 동작하면 치명적일 수 있습니다. 이를 방지하기 위해 Controller 엔진은 사용자 프로그램이 종료되어도 자동으로 재개되지 않습니다.

모든 사용자 프로그램이 멈추면 프로그램 페이지 상단에 **노란색 경고 배너**가 표시됩니다:

- "Controller 엔진이 일시정지 상태입니다"
- **[Controller 엔진 다시 시작]** 버튼 → 확인 다이얼로그 → 사용자 명시 승인 후에만 엔진 재개

사용자가 [재시작] 을 누르지 않는 한 GPIO / Modbus 출력은 모두 OFF 상태로 안전하게 유지됩니다.

9-6 Python 샘플: 순차 점등 (Sequence Blinker)

OUT0 → OUT1 → OUT2 → OUT3 → (반복) 순으로 각 출력을 3초 ON / 3초 OFF 시키는 데모. OLED 에는 IP:포트 / 현재 카운트다운 / 출력 상태가 표시됩니다.

위치: `samples/sequence_blinker/main.py` + `samples/sequence_blinker.zip`

```
from gpiozero import LED
from smbus2 import SMBus
import signal, sys, threading, time

OUTPUT_PINS  = [22, 23, 26, 27]   # BCM, OUT0~OUT3
HOLD_SECONDS = 3.0

outputs = [LED(p) for p in OUTPUT_PINS]
for o in outputs: o.off()

# (SSD1306 드라이버 + Font5x8 폰트는 main.py 에 self-contained 로 포함)
# OLED: IP:5000 / "T 2.7s O2 ON" 카운트다운 / O0:bit O1:bit / O2:bit O3:bit

def cleanup(*_):
    for o in outputs: o.off(); o.close()
    sys.exit(0)
signal.signal(signal.SIGINT,  cleanup)
signal.signal(signal.SIGTERM, cleanup)

while True:
    for i, out in enumerate(outputs):
        out.on();  time.sleep(HOLD_SECONDS)
        out.off(); time.sleep(HOLD_SECONDS)
```

Pi 에는 `gpiozero`, `smbus2` 가 기본 설치되어 있습니다 (외부 OLED 라이브러리 불필요). 종료 시그널을 받으면 모든 출력을 OFF 한 뒤 OLED 도 끕니다.

10. 부록

10-1 트리거 종류 전체

이름	인덱스	발화 조건
OnInputRising/Falling/Change	IN# (0~3)	입력 에지
OnMemoryRising/Falling/Change	M# (0~499)	M[i] 에지
OnCounterEqual	C# (0~49), param=목표	C[i] 도달 순간
OnMbEqual	D#, param=목표	D[i] 도달 순간
OnTimerElapsed	T# (0~49)	T[i].Elapsed 스캔
OnTick	intervalMs=주기	주기 반복 (최소 50ms)
Always	-	매 50ms

10-2 액션 종류 전체

이름	인덱스	param / text
SetOutput / ResetOutput / ToggleOutput	OUT# (0~3)	-
MirrorInput / InvertInput	OUT#	param=소스 IN#
SetMemory / ResetMemory / ToggleMemory	M#	-
StartTimer	T#	param=ms
ResetTimer	T#	-
IncCounter / ResetCounter	C#	-
WriteModbus	D#	param=값
WriteOledLine	OLED slot	**text=표시 문자열**
ClearOledLine	OLED slot	-

10-3 기본 접속 정보

항목	값
웹 UI 포트	5000
웹 UI URL	http://<IP>:5000/ 또는 http://going.local:5000/
기본 비밀번호	admin (첫 로그인 후 변경 권장)
Modbus TCP 포트	502
설정 파일	/home/going/App/ZpiPlc/{rules,modbus,auth,programs}.json
사용자 프로그램	/home/going/App/ZpiPlc/programs/<app_name>/
SSH 계정	going (호스트명과 동일)
서비스 이름	zpi-io08r-plc.service (systemd)
서비스 로그	sudo journalctl -u zpi-io08r-plc -f
스캔 주기	50 ms (엔진) / 50 ms (OLED 별도)

10-4 트러블슈팅

증상	해결
로그인 화면이 안 뜸	http://<IP>:5000/login.html 직접 접속. 캐시 비우고 강력 새로고침 (Ctrl+Shift+R)
비밀번호 잊음	SSH 접속 → `sudo rm /home/going/App/ZpiPlc/auth.json` → 서비스 재시작 → admin으로 초기화
1분간 잠김 (10회 실패)	1분 대기 후 재시도. 비밀번호를 모르면 위 항목으로 초기화
Wi-Fi SSID 1개만 보임	PolKit 규칙 미설정 — 12장 12-2 참조 (설계서)
저장 후 잠시 응답 없음	첫 IronPython 워밍업 ~4.5초 (Pi Zero 2 W). 이후 컴파일 수백 ms
스크립트 구문 오류	line/col 정보 확인. 이전 정상 스크립트가 계속 동작
모니터/룰에디터/Modbus 메뉴가 안 보임	사용자 프로그램이 실행 중 (Controller 엔진 자동 OFF). 프로그램 페이지에서 ■ 정지 → 노란 배너의 [Controller 엔진 다시 시작] 클릭
사용자 프로그램이 실행 안 됨	프로그램 페이지 → 로그 아이콘 클릭. 잘못된 실행 파일명이 입력됐다면 다이얼로그에서 빈 칸으로 두고 재설치 (자동 감지)
OUT 토글 안 됨	매트릭스/카드/스크립트 충돌 — 마지막이 이김. 원하지 않는 룰 비활성화
D[i] 자꾸 0으로 돌아감	외부 Modbus 마스터가 0을 쓰는지 확인 (양방향 동기화)
Modbus 서비스 변경 후 적용 안 됨	저장 후 시스템 메뉴에서 "서비스 재시작" 클릭
장비 재부팅 버튼 실패	passwordless sudo 미설정 — `echo 'going ALL=(ALL) NOPASSWD: /sbin/reboot' | sudo tee /etc/sudoers.d/zpi-plc`

— 끝 —