# Documento Maestro de Diseño y Arquitectura: Aqua Controller USB (Versión 5.0)

\*\*Versión: 5.0\*\*

\*\*Fecha: 06 de agosto de 2025\*\*

\*\*Actualizado por: Chief Programmer & PMO, basado en el documento v5.2 anterior, corrección de inconsistencias (e.g., versión ESP-IDF ajustada a v5.4.1 estable, fechas coherentes, eliminación de repeticiones, uniformidad en librerías y anti-patrones)\*\*

## Preámbulo: Propósito y Legado de este Documento

Este documento es la \*\*fuente única de verdad\*\* para el proyecto Aqua Controller USB. Preserva la historia, decisiones y arquitectura integral de hardware, software y operaciones. Todo colaborador, humano o IA (Grok 4, Gemini CLI), debe consultarlo antes de cualquier cambio o expansión. La versión 5.0 incorpora correcciones para consistencia, migración a ESP-IDF v5.4.1, optimizaciones para ESP32-S3, y refuerzo de las Reglas de Diseño de Software para código limpio y auditable.

\*\*Observaciones de Chief Programmer & PMO\*\*: Documento robusto y estructurado. Almacenarlo en GitHub (https://github.com/rolivape/aqua\_controller/) con Markdown. Changelog integrado para trazabilidad. Grok 4 y Gemini CLI generarán código bajo indicaciones, validando IRAM con `idf.py size`.

---

## Capítulo 1: Fundamentos Estratégicos del Proyecto

### 1.1. Propósito y Valor Diferencial

- Automatización y protección activa de acuarios marinos.

- Monitoreo y control \*\*local-first\*\*: sin nube, máxima autonomía.

- Redundancia: lógica local en módulos.

- Escalabilidad: modularidad plug & play.

- Auditoría: logging total.

\*\*Observaciones de Chief Programmer & PMO\*\*: Local-first asegura fiabilidad. Métricas: uptime >99.9%, latencia MQTT <50ms con NCM, SAFE MODE <3s. Integrar IA local (TensorFlow Lite en RPi5).

### 1.2. Narrativa Histórica y Decisiones Fundamentales

- Prototipos iniciales (Arduino/WiFi): descartados por latencia.

- Adopción de USB CDC: determinismo y seguridad.

- Migración a USB-NCM (v5.0): red IP sobre USB para MQTT eficiente.

- Motor de reglas JSON/DSL.

- Uso de ESP-IDF v5.4.1: optimizaciones IRAM, fixes USB para ESP32-S3.

\*\*Observaciones de Chief Programmer & PMO\*\*: NCM como hito clave. Probar latencia vs. CDC. Verificar IRAM con v5.4.1.

### 1.3. Anti-Patrones Documentados

- No lógica vital en nube.

- Prohibido Wi-Fi para funciones críticas.

- No hardware/software propietario.

- Documentar bugs en changelog.

- Parametrizar roles/configs (NVS).

- Evitar IRAM >80% en ESP32-S3 (máx. 15 reglas, PSRAM buffers).

- Evitar dependencias pesadas (e.g., esp\_console).

\*\*Observaciones de Chief Programmer & PMO\*\*: Alineado. Monitorear IRAM con `CONFIG\_SPI\_FLASH\_AUTO\_SUSPEND` en v5.4.1.

### 1.4. Visión a Largo Plazo

Evolución a predictivo con IA local. Documentación viva para migrabilidad.

\*\*Observaciones de Chief Programmer & PMO\*\*: Integrar TensorFlow Lite en RPi5. Usar Grok 4 para reglas dinámicas.

---

## Capítulo 2: Arquitectura General del Sistema

### 2.1. Topología y Componentes

- \*\*MASTER (RPi 5)\*\*: Motor reglas, HMI, MQTT consola.

- \*\*Paneles (ESP32-S3)\*\*: AC/DC/IO, SAFE MODE autónomo.

- \*\*ATLAS\_PI (RPi Zero 2W)\*\*: Sensores Atlas, MQTT/TCP-IP.

- Expansión: AUX, autoid por NCM.

\*\*Observaciones de Chief Programmer & PMO\*\*: ATLAS\_PI a TCP-IP uniformiza. v5.4.1 mejora USB en S3.

### 2.2. Diagrama Físico/Lógico

```

[MASTER RPi 5]

| (MQTT/NCM, IP: 192.168.1.100)

[Hub USB]

| (NCM)

[PANEL\_AC] [PANEL\_DC] [PANEL\_IO]

| | |

[Relés AC] [Válvulas DC] [Entradas Digitales]

|

[ATLAS\_PI] — [Sensores Atlas] (MQTT/TCP-IP, IP: 192.168.1.101)

```

\*\*Observaciones de Chief Programmer & PMO\*\*: Bidireccional MQTT. Sugerir multicast `aqua/panel/#`.

### 2.3. Estructura de Alimentación

- Fuente 12V 10A.

- Buck 12V→5V.

- Fusibles/TVS.

- Aislamiento.

- Redundancia.

\*\*Observaciones de Chief Programmer & PMO\*\*: Monitoreo voltaje en PANEL\_DC (ADC S3, alertas MQTT).

### 2.4. Escenarios de Falla y Resiliencia

- MASTER falla: SAFE MODE en paneles.

- PANEL\_AC: Fallo temp → apaga calentadores.

- PANEL\_DC: Sobrecorriente → apaga salidas.

- PANEL\_IO: Fallo sensor → alarma.

- ATLAS\_PI: Fallback local.

\*\*Observaciones de Chief Programmer & PMO\*\*: SAFE MODE <3s. Watchdog RTC\_WDT. v5.4.1 fixes cache.

---

## Capítulo 3: Especificaciones de Hardware y Módulos

### 3.1. Resumen de Hardware

| Módulo | MCU/SOC | I/O Principal | Funciones Clave | Consumo Estimado |

|----------|----------------|-------------------------------|---------------------------------------------|------------------|

| MASTER | RPi 5 | USB3.0, LAN, HDMI | Motor reglas, HMI, logging, OTA | 2.5A @ 5V |

| PANEL\_AC | ESP32-S3-WROOM | 8 relés, 2 DS18B20 | Control AC, temp, SAFE MODE térmico | 200mA max |

| PANEL\_DC | ESP32-S3-WROOM | 8 MOSFET, 4 PWM | Control DC, bombas/válvulas, SAFE MODE | 250mA max |

| PANEL\_IO | ESP32-S3-WROOM | 16 optoentradas | Sensores nivel/fuga, alarmas | 150mA max |

| ATLAS\_PI | RPi Zero 2W | I2C, USB, Ethernet (opcional) | Sensores Atlas, aislamiento, MQTT | 500mA max |

\*\*Observaciones de Chief Programmer & PMO\*\*: Ethernet HAT para ATLAS\_PI. Verificar buck (2A/módulo). v5.4.1 reduce bin en S3.

---

## Capítulo 4: Arquitectura de Software y Firmware

### 4.1. Firmware Universal

- Binario por familia, rol por NVS.

- Basado en ESP-IDF v5.4.1, componentes compartidos.

\*\*Observaciones de Chief Programmer & PMO\*\*: Optimizar IRAM <80%. Macros para roles. Autoid NCM.

### 4.2. Flujo de Inicialización y Ciclo Principal

1. Boot, lectura NVS.

2. NCM conexión (IP estática).

3. Loop: Parseo reglas, ejecución, SAFE MODE.

4. Heartbeat MQTT 5s, watchdog.

\*\*Observaciones de Chief Programmer & PMO\*\*: Reconexión <5s. `vTaskDelay(10)`. v5.4.1 mejora IRAM.

### 4.3. Actualización OTA

- Desde MASTER por MQTT/NCM, SHA256, rollback.

\*\*Observaciones de Chief Programmer & PMO\*\*: Chunked MQTT. Backup NVS.

### 4.4. Estructura del Repositorio

```

/aqua\_controller/

master\_app/ # Python, Flask

esp32\_firmware/ # Firmware ESP32-S3

atlas\_module/ # RPi Zero 2W (MQTT/TCP-IP)

scripts/ # Flashing, backup, JSON validators

tests/ # Pytest, Unity

docs/ # Markdown

```

\*\*Observaciones de Chief Programmer & PMO\*\*: Submodules para ESP-IDF v5.4.1, cJSON. Dockerfile reproducible.

---

## Capítulo 5: Protocolo de Comunicación y Contrato de API

### 5.1. Modelo de Comunicación

- MQTT sobre NCM/TCP-IP, heartbeat 5s.

\*\*Observaciones de Chief Programmer & PMO\*\*: MQTT <50ms. QoS=1 críticos, retain configs.

### 5.2. Catálogo de Comandos JSON

- Comandos: `get\_status`, `set\_output`, etc.

- Formato: `{ "id": "...", "api\_version": "1.0", "fw\_version": "5.0", ... }`.

\*\*Observaciones de Chief Programmer & PMO\*\*: Validar schema. Tema `aqua/panel/<id>/events`.

---

## Capítulo 6: Lógica de Automatización (Motor de Reglas)

### 6.1. Motor de Dos Niveles

- JSON/YAML declarativo, Lua DSL opcional.

### 6.2. Triggers, Conditions y Actions

- Triggers: thresholds, cambios.

- Conditions: >, <, all/any.

- Actions: set\_output, notify.

### 6.3. Gestión de Reglas

- Edición, hot reload, prioridad.

\*\*Observaciones de Chief Programmer & PMO\*\*: JSON con priority, hysteresis. Limit 15 reglas, stack 4KB. Lua <10KB IRAM. v5.4.1 optimiza.

---

## Capítulo 7: Interfaz y Experiencia de Usuario (HMI)

### 7.1. Wireframes

- Dashboard, editor reglas.

### 7.2. Alertas

- Niveles: INFO/WARNING/CRITICAL.

- Canales: HMI, push.

- Anti-fatiga.

\*\*Observaciones de Chief Programmer & PMO\*\*: Flask en RPi5, MQTT suscrito. Export CSV logs.

---

## Capítulo 8: Persistencia de Datos y Base de Datos

### 8.1. Persistencia

- SQLite3, rotación logs.

### 8.2. Esquema SQL

```sql

CREATE TABLE telemetry\_log (

id INTEGER PRIMARY KEY AUTOINCREMENT,

timestamp DATETIME DEFAULT CURRENT\_TIMESTAMP,

module\_id TEXT NOT NULL,

sensor\_name TEXT NOT NULL,

value REAL

);

CREATE TABLE event\_log (

id INTEGER PRIMARY KEY AUTOINCREMENT,

timestamp DATETIME DEFAULT CURRENT\_TIMESTAMP,

module\_id TEXT,

event\_type TEXT,

description TEXT,

severity TEXT NOT NULL

);

```

\*\*Observaciones de Chief Programmer & PMO\*\*: Triggers SQL alertas. Rsync backups.

---

## Capítulo 9: Reglas de Diseño de Software

Objetivo: Modularidad, mantenibilidad.

### 9.1. Principios

- main.c mínimo.

- Sufijo \_aq.

- Carpetas: /components, /include, /tests, /scripts.

- Validar JSON schema.

- Config externa (NVS).

- SAFE MODE autónomo.

- Documentar bugs.

- Tests: Pytest, Unity/CMock.

- Logs JSON.

- Interfaces genéricas.

- Validación: builds con tests.

\*\*Observaciones de Chief Programmer & PMO\*\*: Clang-format. Lua <10KB IRAM. v5.4.1 para IRAM flash.

---

## Capítulo 10: Anexos Técnicos

### 10.1. Pinout PANEL\_AC

| Función | GPIO | Notas |

|------------------|------|----------------|

| Relé Calentador | 5 | Canal 1 |

| Sensor Temp | 21 | 1-Wire DS18B20 |

### 10.2. Matriz Alarmas

| Sensor | Evento | Acción |

|-------------|--------------------|--------------------|

| Nivel Sump | Baja | Apagar, WARNING |

| Fuga | Humedad | Cerrar, CRITICAL |

### 10.3. Validación

- Pruebas NCM tres módulos.

- SAFE MODE <3s.

- Sensores límites.

- Config NCM: tinyusb\_net\_init.

- Tests: Pytest, Unity.

\*\*Observaciones de Chief Programmer & PMO\*\*: Latencia <50ms, reconexión <5s. v5.4.1 fixes USB.

### 10.4. Ejemplo Regla JSON

```json

{

"rule\_id": "uuid-1234-5678",

"priority": 5,

"description": "Alerta temperatura alta",

"trigger": { "type": "on\_change", "target": "sensors.temp\_agua", "debounce\_ms": 500 },

"condition": { "all": [{ "lhs": "sensors.temp\_agua", "op": ">", "rhs": 27.5, "hysteresis": 1.0 }] },

"action": { "type": "set\_output", "params": { "channel": 0, "value": 0 }, "notify\_topic": "aqua/panel/<id>/events" },

"error\_action": { "type": "notify", "params": { "topic": "aqua/panel/<id>/errors", "message": "Sensor failure" } }

}

```

### 10.5. Librerías Válidas

- USB-NCM: esp\_tinyusb (nativo, ref: docs.espressif.com/projects/esp-idf/en/v5.4.1/esp32s3/api-reference/peripherals/usb.html).

- MQTT: esp-mqtt (nativo).

- NVS: nvs\_flash (nativo).

- JSON: cJSON (submodule Git).

- Watchdog: esp\_task\_wdt (nativo).

- I/O: gpio, ledc (nativos).

\*\*Observaciones de Chief Programmer & PMO\*\*: Solo nativas o submodules. Validar con menuconfig. Prohibidas dependencias externas.

---