Building an Unstoppable Home Assistant: AQEX Stubborn Eternal UPS Guide

Few tech mishaps are as frustrating as a Home Assistant host going dark during a power glitch. A sudden blackout can corrupt database files, freeze local Zigbee/Z-Wave meshes, and disrupt automated security routines.

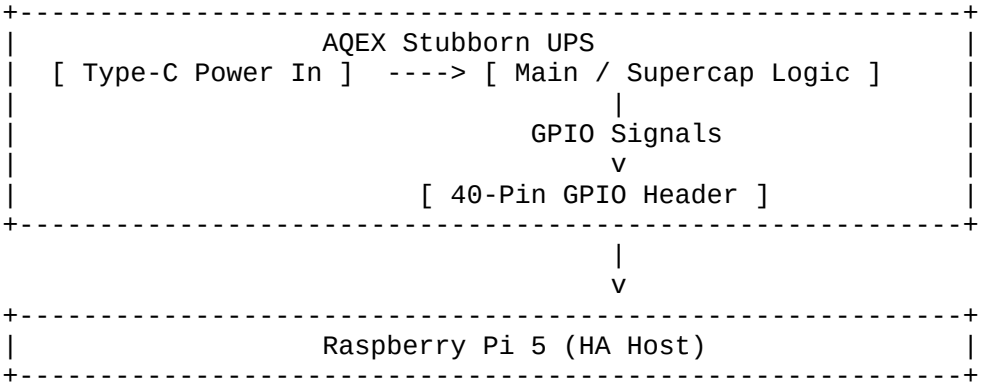
While enterprise UPS units are bulky and inefficient for single-board computers, hardware-level backup modules like the **AQEX Stubborn Eternal UPS** series bridge the gap. Designed to handle the high transient power demands of the Raspberry Pi 5, these boards keep the system running smoothly through micro-outages and cleanly signal when it's time to perform a safe shutdown before energy reserves empty.

Here is a step-by-step walkthrough for assembling, configuring, and testing an immortal Home Assistant instance powered by the AQEX UPS, the `qups-guard2-ha` daemon, and MQTT auto-discovery.

Step 1: Hardware Setup & Power Cabling

Setting up the AQEX UPS with a Raspberry Pi 5 requires minimal physical assembly:

1. **Mount the HAT:** Secure the AQEX UPS module onto the Raspberry Pi 5’s 40-pin GPIO header using the provided standoffs.
2. **Set DIP Switches:** Ensure the DIP switch selection on the board matches your configuration (e.g., DIP `"01"` mode).
3. **Connect Energy Storage:** Wire your energy pack (LiFePO4 cell or Hybrid Supercapacitor array) into the dedicated terminal block.
4. **Power Input:** Plug your primary Type-C power supply directly into the input port of the AQEX board rather than the Pi 5. The UPS manages active pass-through power, handles energy management, and monitors line status.



Step 2: Install `qups-guard2-ha` & Setup JSON Config

Because supercapacitors and backup cells supply short-to-medium backup runtimes, exact percentage monitoring isn't necessary. Instead, the AQEX board utilizes low-latency **GPIO status lines** to instantly signal mains power loss and trigger a "Low Power Limit" signal while energy reserves remain.

The **`qups-guard2-ha`** service reads these physical GPIO state lines via `/dev/gpiochip0` and publishes Home Assistant MQTT messages.

1. Install `qups-guard` Daemon

Clone the repository and build or run the daemon on your host OS:

Bash

```
# Clone the official repository
git clone https://github.com/aqexhu/qups-guard.git
cd qups-guard
```

2. Configure `qups-guard2-ha.json`

Edit the JSON configuration file to match your hardware DIP setting, system GPIO chip, and local Home Assistant MQTT broker credentials:

JSON

```
{
  "gpio": {
    "dip": "01",
    "chip_path": "/dev/gpiochip0"
  },
  "ups": {
    "shutdown_delay": 10
  },
  "mqtt": {
    "enabled": true,
    "broker": "192.168.1.100",
    "port": 1883,
    "username": "ha_user",
    "password": "your_secure_password",
    "node_id": "qups_guard",
    "state_topic": "qups/state",
    "discovery_prefix": "homeassistant"
  }
}
```

3. Automatic Device Discovery

Once the `qups-guard2-ha` service starts and connects to your broker, Home Assistant automatically discovers the AQEX UPS device under **Settings** > **Devices & Services** > **MQTT**:

- `binary_sensor.qups_guard_mains_power` (`on` = AC Power Normal, `off` = Outage)
- `binary_sensor.qups_guard_battery_low` (`off` = Reserve OK, `on` = Reserve Low)

Step 3: Graceful Shutdown Automation

Because the `qups-guard2-ha` daemon natively triggers an OS-level shutdown (after the configurable `shutdown_delay` countdown in `qups-guard2-ha.json`), you don't need to invoke `service: hassio.host_shutdown` inside Home Assistant.

Instead, Home Assistant automations can focus purely on broadcast notifications, snapshotting states, and saving critical databases before the system goes down:

YAML

```
alias: "System: Emergency Power Warning Notification"
description: "Sends notification when AQEX UPS signals low reserve."
trigger:
  - platform: state
    entity_id: binary_sensor.qups_guard_battery_low
    to: "on"
condition:
  - platform: state
    entity_id: binary_sensor.qups_guard_mains_power
    state: "off"
action:
  - service: notify.notify
    data:
      title: "⚠️ Emergency Power Warning"
      message: "Mains power lost and UPS reserve reached low threshold. qups-
guard service initiating graceful OS shutdown."
```

> **Developer Note:** If you prefer Home Assistant to exclusively handle host power management instead of the daemon, you can comment out or modify the OS shutdown syscall directly within the open-source `qups-guard` source code, rebuild the binary, and trigger `service: hassio.host_shutdown` from your Home Assistant YAML automation instead.

Step 4: Live Stress Test (Yank the Cable!)

Theory is good, but empirical proof is better. A real-time pull test demonstrates the seamless transition:

1. Open your Home Assistant dashboard with live history graphs for the `qups_guard` entities.
2. Direct-pull the Type-C USB power cable out of the AQEX UPS board.

Results

- **Zero Latency Switchover:** The Raspberry Pi 5 suffers **no reboot, voltage sag, or power glitch**. Hardware power routing takes over in microseconds.
- **Instant Signal Dispatch:** Power loss is immediately published to `qups/state` over MQTT and updated in Home Assistant.
- **Safe Landing:** The board continues to power the Pi 5 smoothly while waiting for power restoration or triggering the automated GPIO shutdown sequence if the outage persists.

Summary

The AQEX Stubborn Eternal UPS paired with the `qups-guard2-ha` service (github.com/aqexhu/qups-guard) provides hardware-grade reliability for Home Assistant deployments. By replacing complex analog battery calculations with instant GPIO hardware signals and native MQTT auto-discovery, it guarantees zero-downtime micro-outage protection and safe, automated OS shutdowns.

Building a Resilient Home Assistant Server: AQEX Stubborn UPS Series Integration Guide

Power instability poses a critical risk to Home Assistant instances. Sudden outages or brownouts can cause SQLite/MariaDB database corruption, disrupt local Zigbee/Z-Wave mesh networks, and render automated routines unresponsive.

While conventional line-interactive or online UPS systems are often oversized and inefficient for single-board computers (SBCs), the **AQEX Stubborn UPS series** provides targeted, board-level power protection. Engineered to handle high transient loads—including the peak power demands of the Raspberry Pi 5 as well as the compact footprint of the Raspberry Pi Zero—the AQEX Stubborn family spans multiple power management topologies and energy storage chemistries:

- **Eternal Series (Supercapacitor / EDLC):** Maintenance-free, battery-less architecture offering ultra-high cycle life (>500,000 cycles) and wide temperature operation. Provides **8 to 168 seconds** of backup runtime out of the box (100F–120F capacitance)—**ideal for micro-outage bridging and rapid, clean shutdowns**. Storage capacity can be further scaled using **Eternal Boost** expansion boards.
    - **Eternal:** The core supercapacitor power protection module.
    - **Eternal Max:** Advanced supercapacitor UPS featuring expanded configuration parameters and an auxiliary 12V–24V DC input interface.
- **Balance Series (Hybrid Supercapacitor):** Combines the rapid pulse-power handling of EDLCs with extended backup energy storage, delivering **13 to 41 minutes** of continuous runtime.
    - **Balance:** Standard hybrid supercapacitor UPS for Raspberry Pi.
    - **Balance Zero:** Dedicated hybrid supercapacitor UPS engineered for the Raspberry Pi Zero form factor.
- **Stamina Series (High-Capacity Battery Solutions):** High-density energy storage (4000 mAh capacity) designed for prolonged power outages, offering **30 to 450 minutes** of extended operation.
    - **Stamina LF:** Equipped with high-safety Lithium Iron Phosphate (LiFePO4 / LFP) chemistry.
    - **Stamina NA:** Powered by next-generation Sodium-Ion (Na-Ion) battery technology.
    - **Stamina Max:** Flexible multi-chemistry model supporting Li-Ion, LFP, and Na-Ion cell configurations.

Regardless of the underlying model or form factor, all AQEX Stubborn UPS modules share a unified control architecture. They maintain continuous, regulated output during micro-outages and communicate power telemetry directly to the host OS, enabling a graceful, controlled shutdown before energy reserves are fully depleted.

This guide provides step-by-step instructions for installing, configuring, and testing any AQEX Stubborn UPS module using the `qups-guard2-ha` background daemon and Home Assistant's native **MQTT Auto-Discovery**.

Step 1: Hardware Setup & Power Distribution

Physical installation of the AQEX Stubborn UPS series requires minimal assembly:

- **1. Mount the Module:** Align and seat the AQEX UPS module onto the host device's 40-pin GPIO header (Raspberry Pi 2/3/4/5 or Pi Zero series). Secure the board using the provided M2.5 standoffs to guarantee mechanical stability and low-resistance GPIO pin contact.
- **2. Optional Capacitance Expansion (Eternal Boost):** For **Eternal and Eternal Max models**, energy storage capacity can be scaled by stacking an optional **Eternal Boost** expansion module directly on top of the UPS board. Available in **450F, 900F, and 1700F** configurations (matching the Raspberry Pi form factor), the boost board connects via a dedicated multi-pin interface and adds pure supercapacitor storage without complex wiring.
- **3. Configure Communication (DIP Switches):** Set the onboard DIP switches to select the active GPIO signaling profile matching your software setup (e.g., `-dip 01`). Refer to the hardware documentation for specific pinout matrices.
- **4. Energy Storage Connection (Stamina Series Only):** For **Stamina battery models**, connect the replaceable battery pack (LFP, Na-Ion, or Li-Ion cell) to the dedicated high-current terminal block. *(Note: Eternal and Balance series feature fully integrated onboard energy storage and do not require external battery wiring.)*
- **5. Primary Power Connection:** Connect your primary power source directly to the AQEX UPS module:
    - **All Models:** Feature a standard **USB Type-C** input port for standard 5V power supplies.
    - **MAX Series (Eternal MAX & Stamina MAX):** Additionally feature an integrated, **reverse-polarity protected 7V–28V DC push-in spring terminal block** (utilizing an onboard step-down DC-DC converter to deliver regulated 5V), making it ideal and bulletproof for industrial 12V/24V cabinet power rails.

⚠️ **CRITICAL POWER WARNING: Do NOT power the Raspberry Pi directly via its onboard USB-C port.** Primary power must pass exclusively through the AQEX UPS module. The board regulates output power, handles active energy management, and continuously monitors input line status. Supplying power directly to the Pi bypasses all protection mechanisms and may cause permanent hardware damage.

Step 2: Install `qups-guard2-ha` & Configure MQTT Auto-Discovery

Unlike complex, software-heavy battery management systems, the AQEX UPS architecture relies on deterministic, hardware-level logic. Because supercapacitor and backup cell discharge curves require instant, fail-safe detection, exact percentage polling is omitted in favor of low-latency GPIO status lines. The board instantly signals mains power loss and asserts a physical **"Low Power Limit"** line while sufficient energy reserves remain to guarantee a graceful host system shutdown.

The lightweight `qups-guard2-ha` background daemon polls these physical hardware lines directly via Linux `/dev/gpiochip` interface and publishes real-time telemetry to your Home Assistant MQTT broker.

1. Install the `qups-guard` Daemon

Clone the official repository and build or run the service on your host OS:

```
# Clone the official AQEX repository
git clone https://github.com/aqexhu/qups-guard.git
cd qups-guard
```

2. Configure `qups-guard2-ha.json`

Edit the JSON configuration file to match your hardware DIP switch settings, host Linux GPIO chip path, and local Home Assistant MQTT broker credentials:

```
{
  "gpio": {
    "dip": "01",
    "chip_path": "/dev/gpiochip0"
  },
  "ups": {
    "shutdown_delay": 10
  },
  "mqtt": {
    "enabled": true,
    "broker": "192.168.1.100",
    "port": 1883,
    "username": "ha_user",
    "password": "your_secure_password",
    "node_id": "qups_guard",
    "state_topic": "qups/state",
    "discovery_prefix": "homeassistant"
  }
}
```

3. Automatic Home Assistant Entity Discovery

Once the `qups-guard2-ha` daemon initializes and connects to your broker, Home Assistant automatically discovers the AQEX UPS device via MQTT Auto-Discovery (**Settings > Devices & Services > MQTT**). The following binary sensors will become immediately available:

- **`binary_sensor.qups_guard_mains_power`**
  - `ON` = Primary DC power active (Normal)
  - `OFF` = Power loss detected (Running on backup energy)
- **`binary_sensor.qups_guard_battery_low`**
  - `OFF` = Energy reserve adequate (OK)
  - `ON` = Energy reserve critical (Low power threshold reached)

Step 3: Graceful Shutdown Automation

By default, the **`qups-guard2-ha`** daemon handles host OS power management directly—initiating a controlled system shutdown once the `shutdown_delay` timer expires (configured in **`qups-guard2-ha.json`**). Because the background daemon guarantees host shutdown at the OS level, calling `action: homeassistant.stop` or `action: hassio.host_shutdown` inside Home Assistant automations is unnecessary.

This allows Home Assistant automations to focus strictly on operational cleanup: broadcasting notifications, flushing state buffers to disk, stopping non-essential services, and ensuring database integrity before power is completely removed.

Example Emergency Power Automation

Add the following YAML snippet to your Home Assistant automations (**`automations.yaml`**):

```
alias: "System: Emergency Power Warning Notification"
description: "Sends notification when AQEX UPS signals low reserve."
trigger:
  - platform: state
    entity_id: binary_sensor.qups_guard_battery_low
    to: "on"
condition:
  - platform: state
    entity_id: binary_sensor.qups_guard_mains_power
    state: "off"
action:
  - service: notify.notify
    data:
      title: "⚠️ Emergency Power Warning"
      message: "Mains power lost and UPS reserve reached low threshold.
qups-guard service initiating graceful OS shutdown."
```

💡 Developer Note: Customizing Shutdown Control

If you prefer Home Assistant to have full exclusive control over host power management rather than the background daemon, you can easily adapt the software. Simply comment out or modify the native OS shutdown system call directly in the open-source `qups-guard` source code, recompile the binary, and invoke `action: hassio.host_shutdown` (or `action: homeassistant.stop`) directly from your Home Assistant YAML automation rules.

Step 4: Live Functional Testing (Mains Power Loss Simulation)

Empirical validation is critical to ensuring power infrastructure reliability. A live disconnect test verifies both zero-latency hardware power switching and real-time telemetry dispatch to Home Assistant:

1. **Monitor Telemetry:** Open your Home Assistant dashboard with real-time status indicators or history graphs for the `binary_sensor.qups_guard_mains_power` and `binary_sensor.qups_guard_battery_low` entities.
2. **Simulate Outage:** Disconnect the primary USB-C power cable (or 7–28V DC source) directly from the input port of the AQEX UPS board.

Expected Hardware & Software Results:

- **Zero-Latency Power Transition:** The host system (Raspberry Pi or Pi Zero) experiences zero voltage sags, power glitches, or unexpected reboots. Onboard hardware power routing transfers the load to energy reserves in sub-millisecond switching times.
- **Instant Telemetry Dispatch:** The loss of mains power is detected hardware-side in microseconds and immediately published over MQTT (`qups/state`), changing `binary_sensor.qups_guard_mains_power` to `OFF` in Home Assistant.
- **Controlled Energy Management:** The AQEX module maintains continuous, regulated output power during the outage. If mains power is not restored before energy reserves cross the low threshold, `binary_sensor.qups_guard_battery_low` toggles to `ON`, triggering your configured automations and the **`qups-guard`** safe OS shutdown sequence.

Summary

The **AQEX Stubborn UPS series**—paired with the open-source **`qups-guard2-ha`** background service—provides enterprise-grade power protection for Home Assistant deployments on Raspberry Pi hardware. By replacing complex, drift-prone analog battery estimations with instant, deterministic GPIO hardware signals and native Home Assistant MQTT Auto-Discovery, the system delivers zero-latency micro-outage bridging, continuous system uptime, and guaranteed safe OS shutdowns.

Whether deployed with supercapacitors, hybrid arrays, or high-density battery chemistries, the AQEX platform ensures your Home Assistant host, local Zigbee/Z-Wave meshes, and database files remain fully protected against power instability.