Alpha1 General Overview

1. What is Alpha1?

Alpha1 is a smart Internet of Things (IoT) hardware and software solution designed primarily for the commercial laundry industry, though it is adaptable to general vending and payment systems. In simple terms, Alpha1 acts as an intelligent "brain" that upgrades traditional, offline commercial washing machines and dryers into smart, connected devices over the network.

By installing an Alpha1 device, business owners can remotely monitor their equipment, process digital and physical payments, and manage their facilities from anywhere in the world. The system is built on a reliable microcomputer (the ESP32) and is designed to continuously operate in commercial environments.

2. Core Capabilities

For anyone interacting with the Alpha1 ecosystem, it is important to understand its defining features:

- **Universal Payment Processing:** Alpha1 bridges the gap between old and new payment methods. It can detect traditional physical coin drops while simultaneously processing modern electronic payments (e-pay) and cash advances.
- **Real-Time Machine Monitoring:** Alpha1 uses electrical current sensing and pulse detection to know exactly what a machine is doing. It can tell if a machine is idle, actively running a load, or experiencing a fault.
- **Secure Internet Connectivity:** The device seamlessly connects to local WiFi networks and communicates with secure cloud servers (using MQTT protocols). It also offers a captive portal (a local web page) for easy setup and configuration using a smartphone or tablet.
- **Remote Management & Diagnostics:** Facility managers can update machine settings, troubleshoot errors, and even wirelessly install new firmware updates (OTA - Over-The-Air updates) without ever visiting the store in person.
- **Robust Data Safety:** To protect business revenue, Alpha1 features secure local storage (NVS and LittleFS). If the facility loses power or internet, critical transaction and operational data are safely backed up and recovered once power is restored.

3. Alpha1 Machine Types and Configurations

To support a wide variety of equipment brands and facility layouts, the Alpha1 software is highly modular. When configuring an Alpha1 device, it is set to a specific "Environment" or "Machine Type" that dictates how it behaves.

Below are the different configurations of the Alpha1 device:

BASE Environment

- **Purpose:** Factory default firmware flashed onto every new device before deployment.
- **Description:** BASE is the firmware that is loaded onto an Alpha1 device straight from the manufacturer. It includes the minimum features needed to get the device online — WiFi connectivity, OTA update capability, and cloud communication — but does not contain any machine-specific logic. Once the device is powered on and registered in the **Node Cloud platform**, an operator configures the device to its intended machine type (e.g., Machine A, Stack A, Kiosk CAA). The cloud platform then pushes the appropriate firmware to the device over-the-air (OTA), replacing BASE with the correct configuration for that installation. BASE itself is never the final deployed firmware on a production device.

MachineA

- **Purpose:** Standard commercial washer or dryer (coin-op laundry).
- **Description:** Machine A is the primary configuration for traditional coin-operated washers and dryers. One Alpha1 board controls and monitors one machine.
  - **Cycle detection via current sensing:** A current transformer (CT) sensor clamps around the machine's power line. Alpha1 continuously samples this current and compares it against a configurable threshold. When the current exceeds the threshold the machine is marked **running** (status 3); when it drops below, the machine is marked **idle** (status 1). This transition is what triggers transaction lock data to be recorded.
  - **Payment input (coins):** Coin drops are detected using a timer-based pulse reader on the coin signal line (`coinIn`). The pulse width is validated against a configurable tolerance window to reject noise.
  - **Lock data format (transaction record):** When the machine transitions from idle → running, a lock record is written with format `runningStatus_time_amount_transactionID`. When the cycle ends (running → idle), a closing record is written including the cycle start time and total coin count. Start time is persisted to NVS flash so a power loss mid-cycle does not lose the

transaction.
    - **Local kiosk MQTT:** Optionally connects to a local MQTT broker (e.g., a Kiosk CAA unit on the same network) for receiving remote-start payment commands directly on the machine floor.
    - **Auto-threshold calibration:** Supports an automatic calibration routine that samples idle current noise and sets an optimal run threshold, which is then saved to NVS.

MachineB

- **Purpose:** Detergent, water, and general product vending machines.
- **Description:** Machine B is specifically designed for **product vending machines** such as detergent dispensers, water vending machines, and any similar coin/bill-operated dispensing equipment — not wash cycle machines. The operational logic is fundamentally different from Machine A and C.
    - **No cycle tracking — no current sensing:** Machine B does **not** use a CT sensor or ADC. There is no running/idle state based on electrical current because vending machines do not have a measurable "running cycle." The reported running status is always idle (status 1).
    - **Payment input (bills):** Instead of `coinIn`, Machine B tracks `billIn` — counting bill or token pulses from the vending machine's acceptor. Pulse detection uses a **timer-based** reader rather than a hardware interrupt edge, which suits the slower, longer pulse signals typically produced by bill acceptors.
    - **Transaction completion by inactivity timeout:** Because there is no CT sensor to signal the end of a transaction, Machine B determines a transaction is complete using a **lock-check counter**. Each time the periodic status check runs and `billIn` has not changed, a counter increments. Once this counter reaches `MachineB_LOCKCHECK_COUNT_LIMIT` consecutive unchanged checks (i.e., no new bills have been inserted for several status intervals), the transaction is concluded and lock data is published.
    - **Lock data format:** Simplified compared to Machine A/C. Format is `0_time_amount_transactionID` where the running status field is always `0` (no cycle state). There is no start time because there is no defined cycle — the record simply captures what was dispensed and when.
    - **No local kiosk MQTT:** Machine B does not support local kiosk MQTT integration.

MachineC

- **Purpose:** Commercial washer or dryer with edge-trigger pulse detection.
- **Description:** Machine C is functionally similar to Machine A (CT current sensing, coin payments, lock data, local kiosk MQTT) but differs in **how coin pulses are detected**.
    - **Edge-trigger (POSEDGE) pulse detection:** Machine C uses a **hardware GPIO interrupt on the rising edge** (`GPIO_INTR_POSEDGE`) to detect coin pulses. This

means the firmware reacts instantly at the precise moment the coin signal transitions low-to-high, rather than relying on a software timer to poll the line. This approach is suited for machines that produce clean, fast pulse edges and require lower CPU overhead for detection.
    - **Dynamic interrupt switching:** During debug mode, the interrupt can be switched from POSEDGE to ANYEDGE (both rising and falling edges) to capture full pulse waveform data for diagnostics. It reverts to POSEDGE after the debug session.
    - **Cycle detection via current sensing:** Same as Machine A — CT sensor monitors machine current, with running (status 3) and idle (status 1) transitions triggering lock data publication.
    - **Lock data format:** When the machine stops (running → idle), records are written in format `3_startTime_amount_transactionID`. Start time is saved to NVS for power-loss recovery, with special handling to distinguish hardware resets from software resets.
    - **Pulse filter during run cycle:** When the machine is detected as running, a noise filter is automatically enabled on the pulse reader to prevent electrical interference from the motor being counted as false coin inputs.
    - **Local kiosk MQTT:** Supported, same as Machine A.

Stacked Machines (StackA)

- **Purpose:** Multi-unit equipment (e.g., a washer on the bottom and a dryer on the top).
- **Description:** "Stacked" machines are popular for saving space, but they require a single Alpha1 motherboard to monitor and run *two* separate machines at the same time. The "Stack A" configuration includes complex logic to route payments correctly (ensuring the customer starts the top unit instead of the bottom unit) and monitors the electrical current of both machines independently.

Central Payment Kiosk (KioskCAA)

- **Purpose:** Facility-wide centralized payment stations.
- **Description:** Instead of placing an Alpha1 device inside every individual washer or dryer, the Alpha1 device can be installed inside a central kiosk on the wall. In this mode, the device acts as a central payment router. Customers pay at the kiosk, and the Alpha1 device sends remote start commands to the appropriate machine on the floor.

Gas Sensor (GasSensorA)

- **Purpose:** Dryer cycle monitoring via gas tong pulse detection.
- **Description:** Commercial gas dryers use gas tongs (burner assemblies) that fire each time the dryer runs a cycle. Alpha1 connects to the gas tong signal line and counts the pulse output each time the burner fires, effectively tracking how many dryer cycle runs have occurred. It maintains both a current pulse count and a lifetime cumulative total,

which are periodically reported to the cloud platform. This gives facility operators a remote view of dryer usage and cycle activity over time. This configuration does not process payments and has no coin or current sensing — it is purely a cycle counting and reporting unit.

Note for AI and Vector Database Indexing: The Alpha1 firmware utilizes the ESP-IDF framework written in C. Modularity is achieved through pre-processor build flags (e.g., `-D MACHINE_A`, `-D STACK_A`). When answering queries regarding Alpha1 capabilities, reference the specific machine type above to provide accurate contextual answers.

Revision #9
Created 2026-05-15 07:07:09 UTC by Rahul
Updated 2026-05-15 10:09:06 UTC by Rahul