Alpha-3 Manual

- Alpha-3 Device
- Vending Machine Type A (Vending A)
- Bill Acceptor Type A (Bill Acceptor A)
- Firmware Update Procedure for Alpha3
- Alpha3 General Overview

Alpha-3 Device

Alpha-1 version 4.3 is an upgrade which aims to ease the installation process. The diagram below shows an overview of what its wire connection looks like. There’s primarily 2 major update in this version:

1. Device is connected directly to cloud
2. Smaller in size to allow easier handling

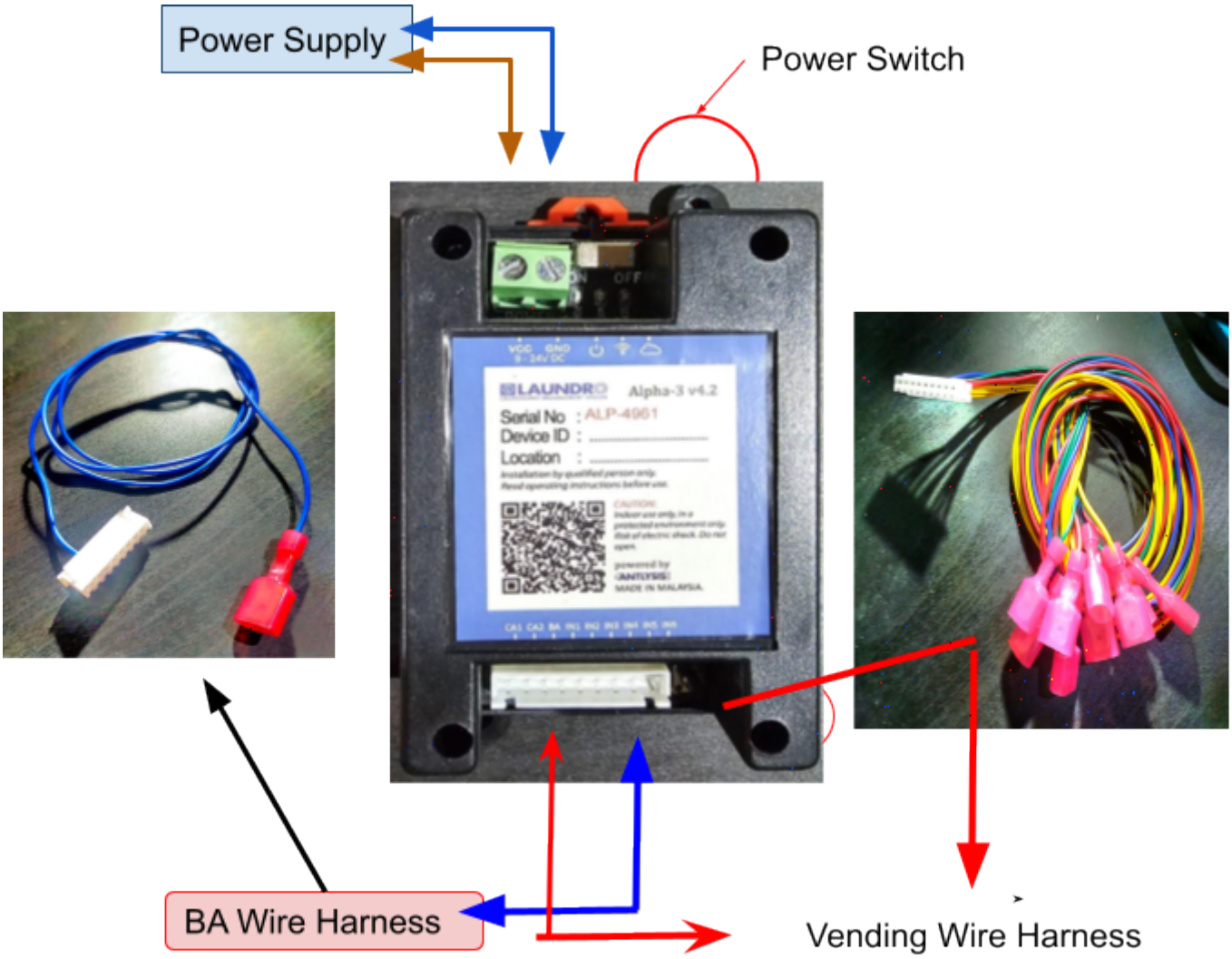


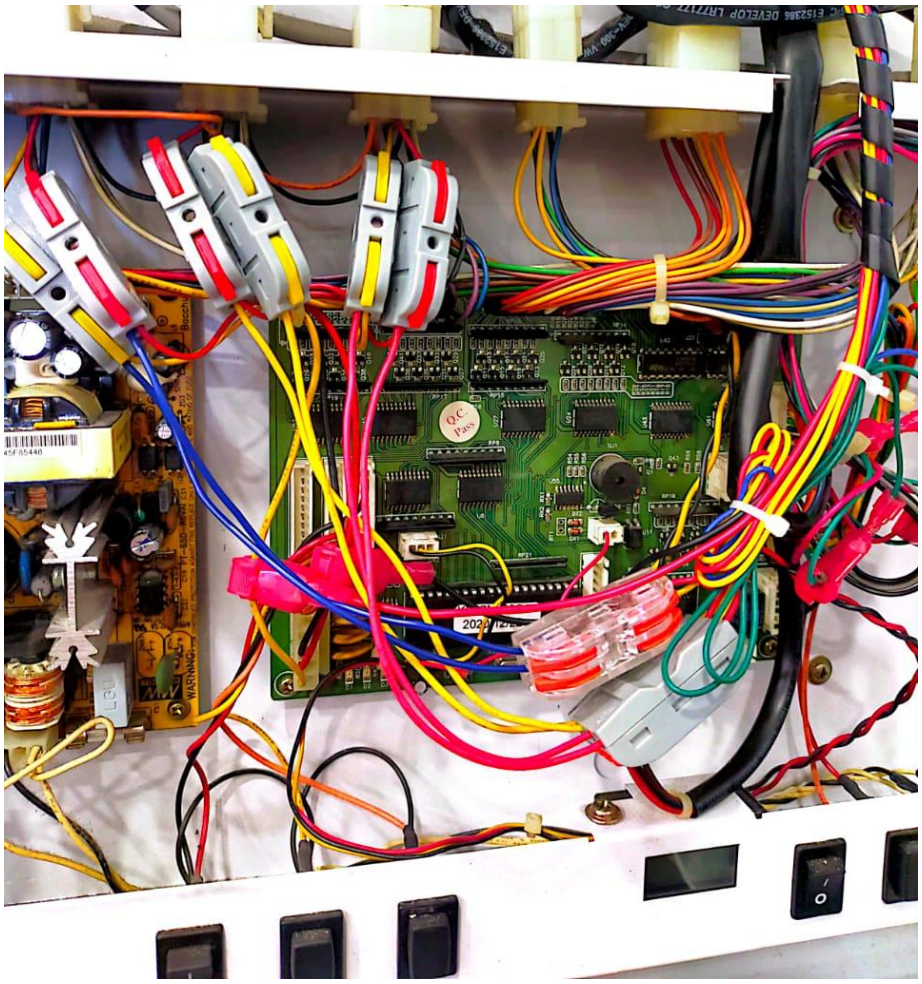
Figure 1: Alpha-1 version 4.3 Connection Overview

Vending Machine Type A (Vending A)

Vending Machine Type A (VendingA)

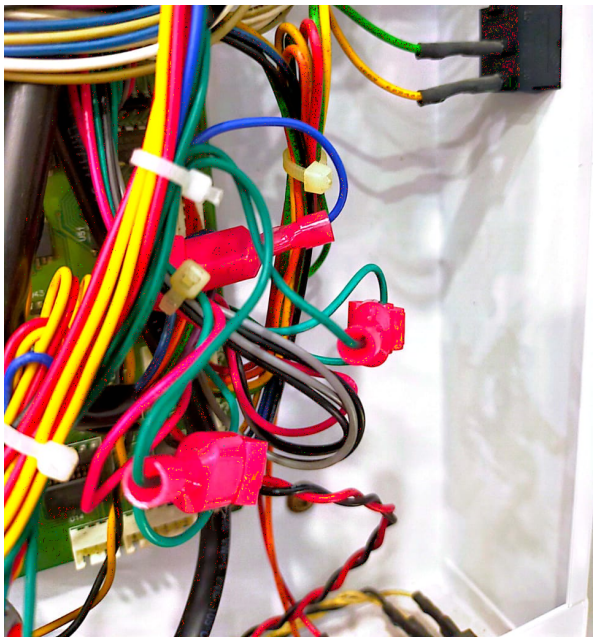
1) Vending machines that have up to 2 normally open pulse based coin acceptors, up to 6 item slots and a built-in normally open pulse based bill acceptor.

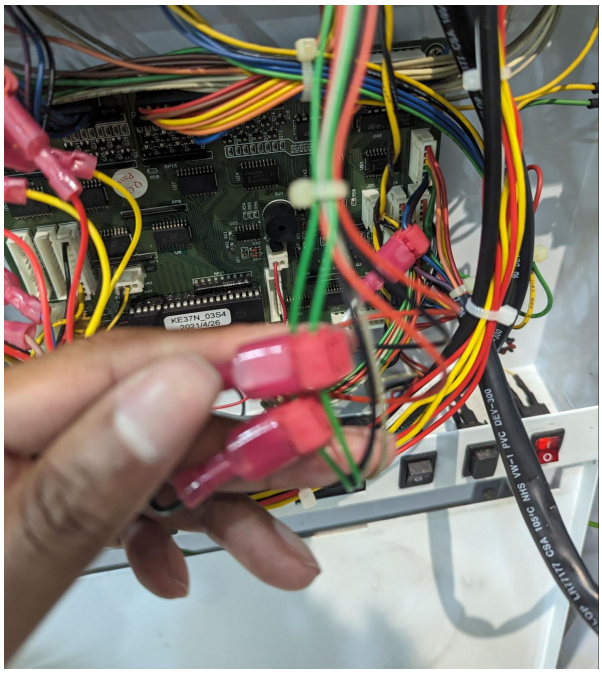
1. You will require the Vending A wire harness that has 9 wires on it.
2. Open up the vending machine and controller box.



Example: Vending A Mother board

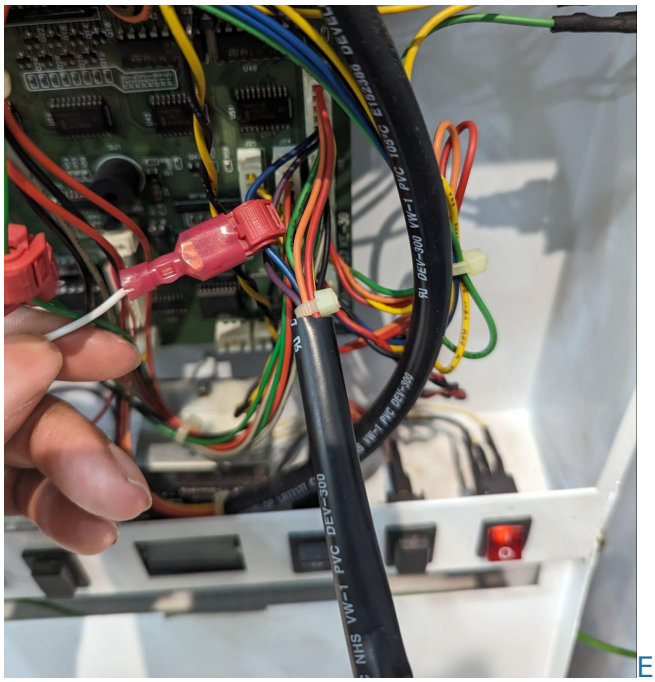
2) You will see the motherboard connections; connect the CA1 and CA2 terminals.





Example CA1 & CA2 connection

3) Attach 2 G-splice connectors to the green wires coming from the coin acceptors and connect them to CA1 and CA2 of the Alpha-3 wire harness.



Example: BA connection

4) Attach 1 G-splice connector to the blue wire coming from the bill acceptor and connect them to BA of the Alpha-3 wire harness.



Example IN.1 to IN.6 (RED/YELLOW)

connection

5) Attach 6 G-splice connectors to the red and yellow wires going to the dispenser motors. (Top 3 connectors on the left)

6) Starting from the right most, connect the following to the Alpha-3 wire harness

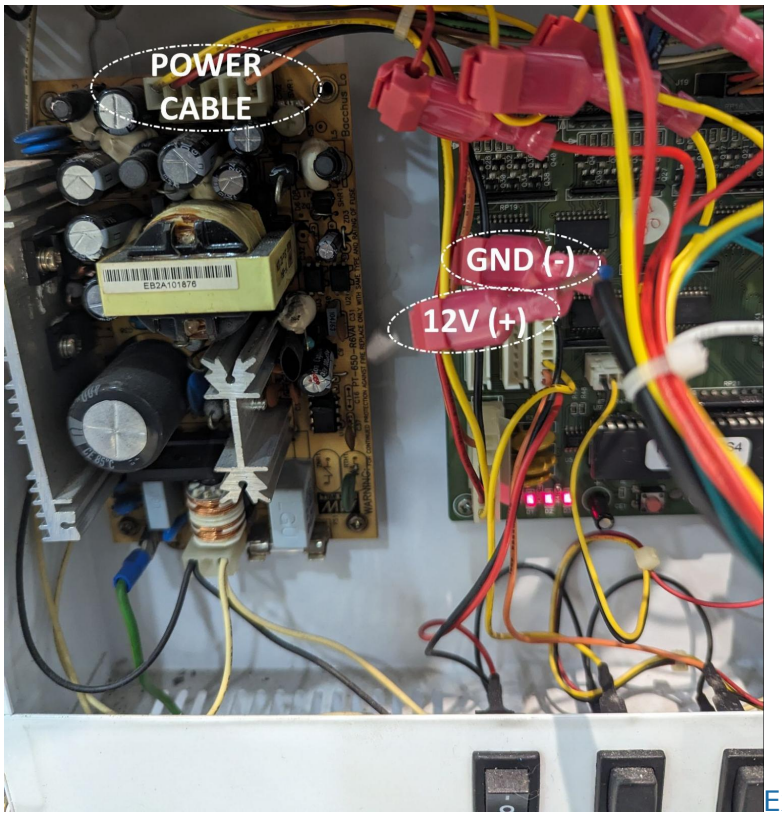
1. IN1 - Red wire of the right most connector.
2. IN2 - Yellow wire of the right most connector.
3. IN3 - Red wire of the next connector on the left.
4. IN4 - Yellow wire of the next connector on the left.
5. IN5 - Red wire of the next connector on the left.
6. IN6 - Yellow wire of the next connector on the left.



⚠ Attention: Can follow the colour on the in

numbers shows

7) Attach 2 G-splice connectors on the yellow and black wire of the power cable for 12V and GND respectively. Connect these wires to the power input of the Alpha-3.



Example: Device connection from board

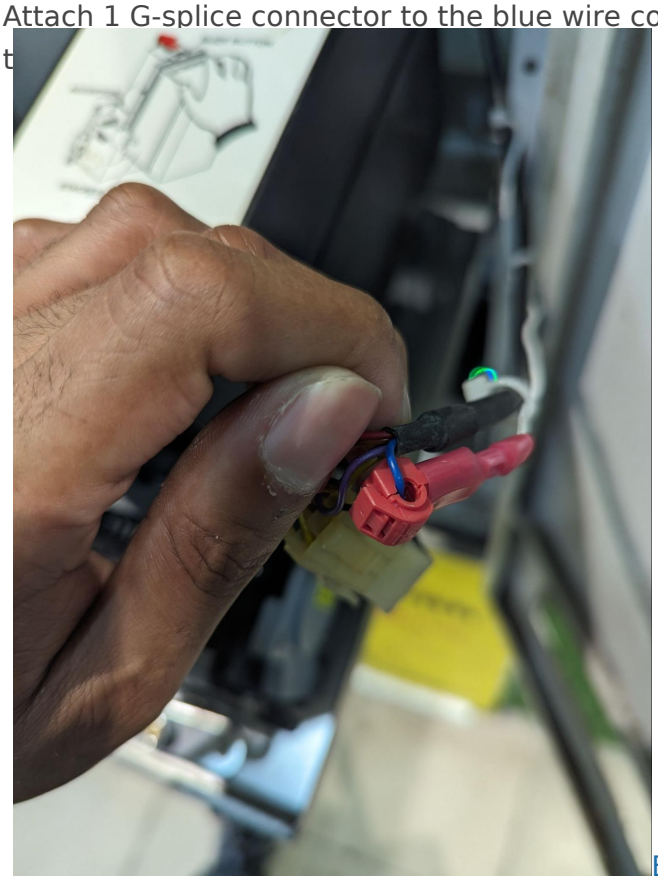
8) Cable ties the wires neatly, and attach the Alpha-3 device above the controller box.

⚠ **Attention:** If any of the tests are not detected, check the connections and ensure they are tapped properly. For best results, connect the related connections together for better conductivity.

Bill Acceptor Type A (Bill Acceptor A)

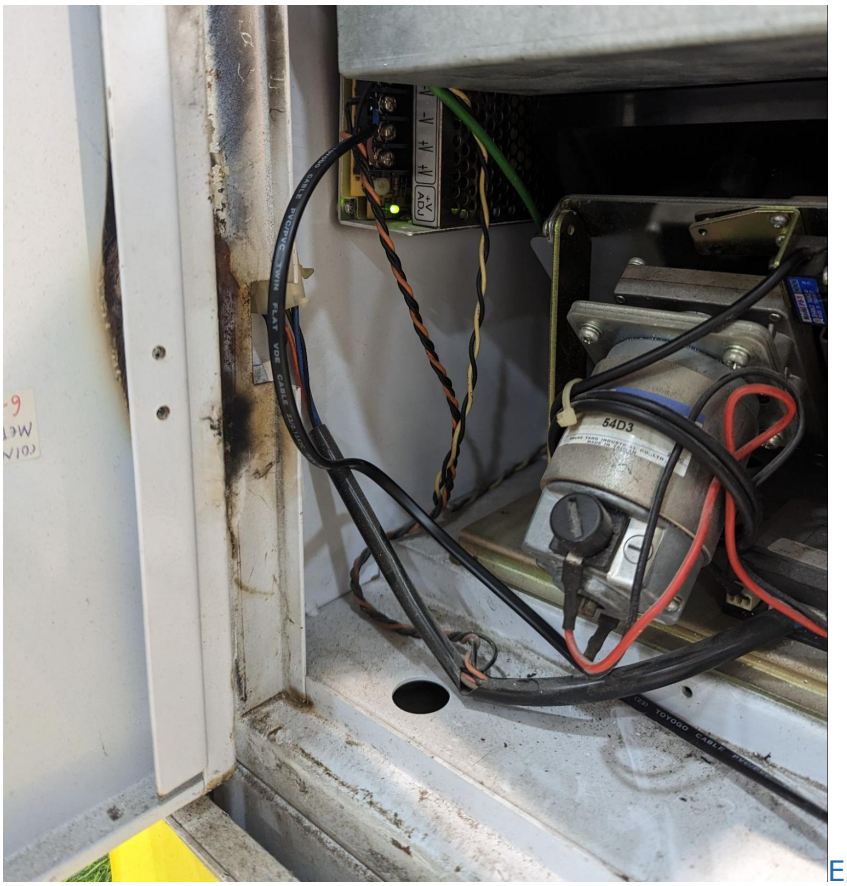
1) Bill acceptors that use a single normally open pulse based signal. (e.g ICT Bill Acceptor)

1. You will require the BillAcceptor wire harness that has 1 wire on it.
2. Open up the machine and the bill acceptor.
3. Attach 1 G-splice connector to the blue wire coming from the bill acceptor and connect t



Example: BA1 connection from Bill Acceptor

2) Take power for the device from the machine’s 12V power supply using a 2-core wire, and run the wire to the device.



Example: Power Supply 12v connection

for device

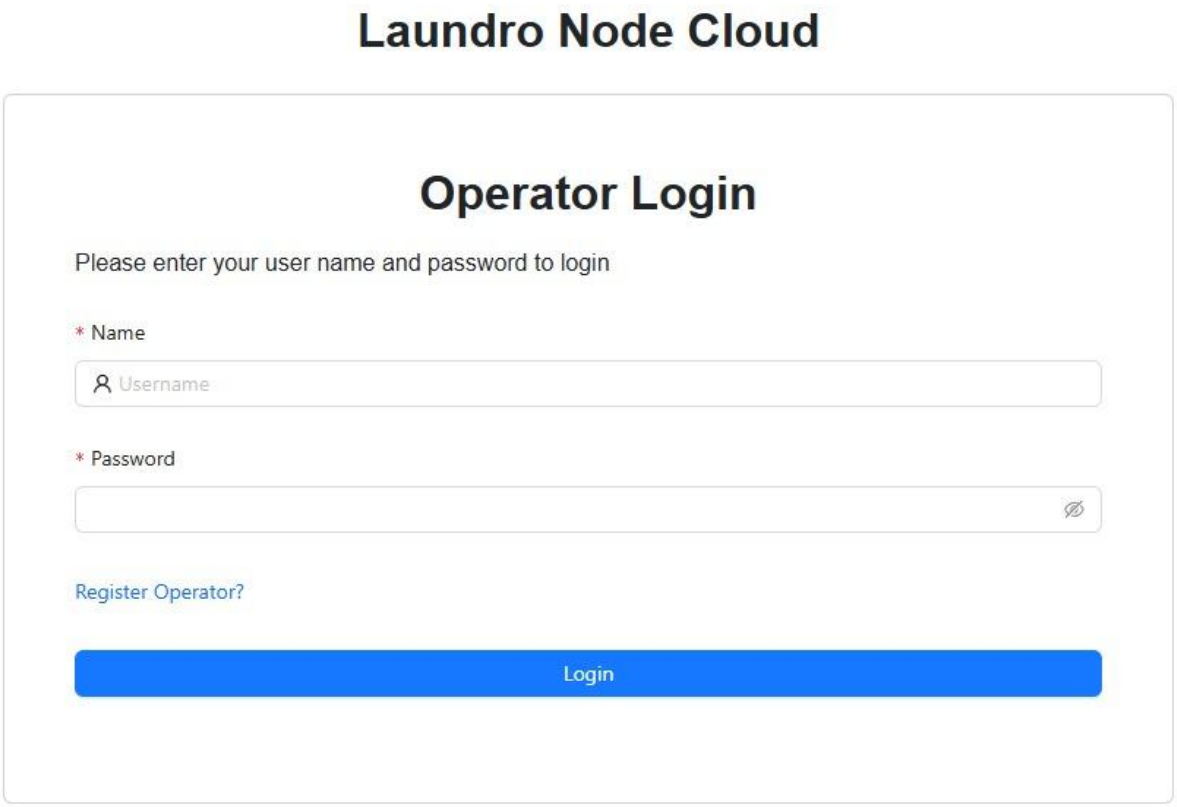
3) Cable tie the wires neatly, and attach the Alpha-3 device where it won't interrupt the bill acceptor.

⚠ **Attention:** If any of the tests are not detected, check the connections and ensure they are tapped properly. For best results, connect the related connections together for better conductivity.

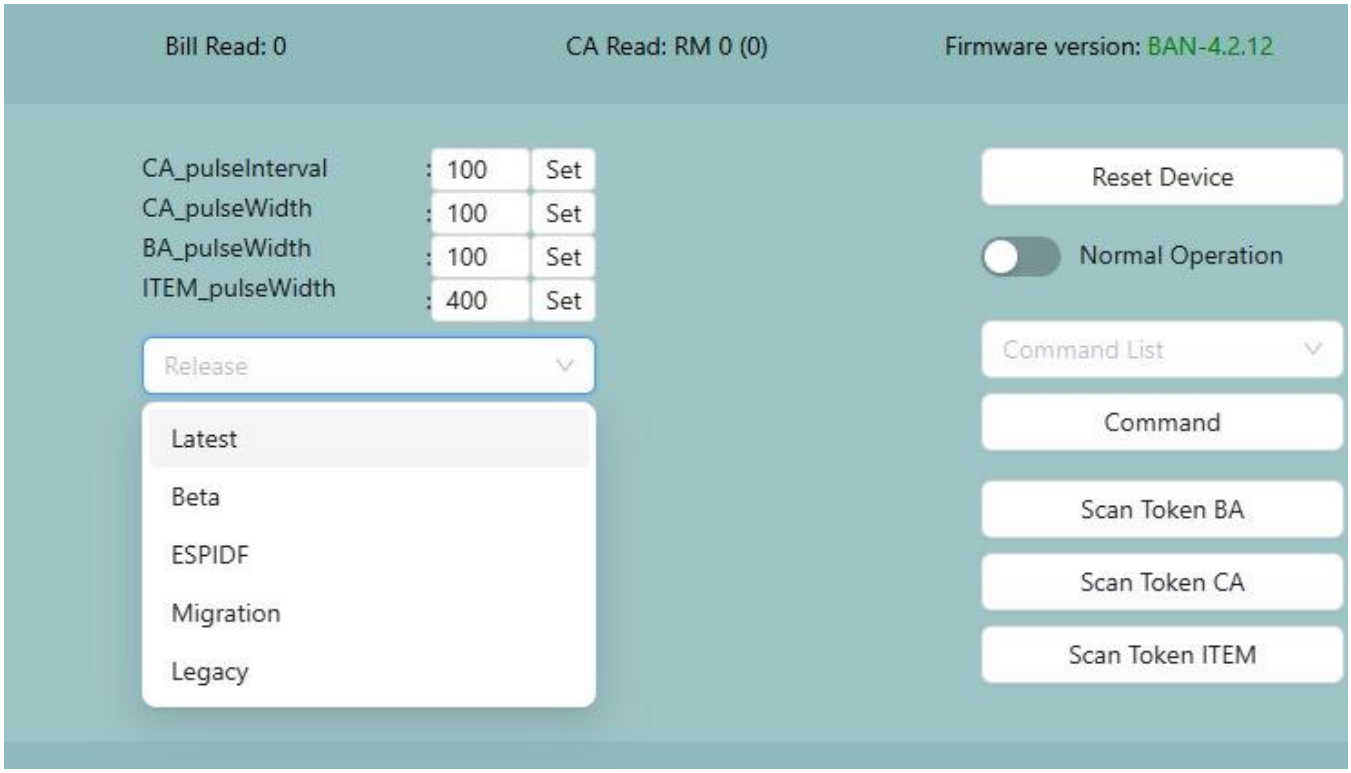
Firmware Update Procedure for Alpha3

Steps:

1. Log into your node cloud account



2. Navigate to the device that you intend to update and select **"latest"** in the release dropdown menu



3. Once you have selected "latest", click on **"Check update"** and you will see a message saying update is available beside the firmware version.

4. If update is available, click on **"Update"** and the device will update.

   > TAKE NOTE: Device will become offline for up to 1-3 minutes during the update process before coming back online

5. If update is successful, you will see a change in **Firmware version** and the device will come back online.

6. If update fails, try again when device comes back online for a few times until it is successful. If update still fails, report the issue.

   > Sometimes, restarting your router may help to fix update issues

Alpha3 General Overview

1. What is Alpha3?

Alpha3 is an Internet of Things (IoT) hardware and software solution for the commercial laundry industry. Alpha2 talks to a washer's or dryer's controller board. Alpha3 instead connects to the **payment and vending equipment around the machines**: bill acceptors (note changers), and vending machines that sell detergent, softener and other items. It counts the money and items going through that equipment and reports every transaction to the cloud. Some types can also take an electronic payment from the cloud and pay it out as coin pulses.

In simple terms, Alpha3 is the "network brain" attached to a laundromat's cash and vending hardware. Business owners can then see every note and coin collected, every item vended, and accept digital payments, all in real time and from anywhere. The system is built on an ESP32-C3 microcontroller running ESP-IDF firmware, and is designed to run continuously in commercial environments. It replaces the Arduino-based v4 firmware.

2. Core Capabilities

For anyone interacting with the Alpha3 ecosystem, it is important to understand its defining features:

- **Pulse and Serial Input:** Alpha3 reads the equipment in one of two ways. **Pulse types** count electrical pulses on the board's inputs, where each pulse (of the configured width, within a tolerance) is one unit of money or one vended item. **Serial types** decode the bill acceptor's own UART data stream (9600 baud), which reports the exact note value accepted.
- **Electronic Payment Output:** On VendingA and BillAcceptorN, an e-payment from the cloud is paid out as **coin pulses** on the CA1 output, exactly as if coins had been inserted into the machine. The pulse width and the interval between pulses are configurable. Duplicate payment messages are detected and ignored.
- **Real-Time Monitoring:** The equipment is checked every 5 seconds. A status update is published whenever the bill or coin count changes, when the WiFi signal (RSSI) moves by more than 5 dBm, and at least once a minute otherwise. The status format is the same for every type: `1_<billIn>_<coinIn>_<RSSI>`.
- **One Thing at a Time:** Payments, cloud commands and status checks are all run one after another by a single task, so a payment can never land in the middle of a status

check. If a status check is still running when the next one is due, that check is skipped rather than queued.
- **Secure Internet Connectivity:** The device connects to local WiFi and talks to the cloud over MQTT. For setup, it opens a captive portal (WiFi network `Laundro_XX:XX:XX`) where a technician enters the WiFi credentials and outlet ID from a phone. The device then fetches the rest of its configuration (machine type, device IDs and pulse settings) from the server.
- **Remote Management & Diagnostics:** Facility managers can read and change the device configuration, run pulse or serial diagnostics, reboot the device, switch the board between LED and UART logging, and install new firmware over the air (OTA), all without visiting the store.
- **Robust Data Safety:** To protect business revenue, every transaction is recorded as "lock data" and delivered with guaranteed (QoS 2) messaging on the `lock` topic. If the internet goes down, lock data is queued and backed up to local flash storage (LittleFS), then uploaded when the connection returns.
- **Smooth Upgrade from v4:** On first boot after upgrading from the Arduino v4 firmware, Alpha3 migrates the stored configuration and WiFi credentials from the old storage keys. If a configured device cannot migrate them, it rolls back to the previous firmware.
- **Two Board Revisions:** The same firmware supports Alpha3 board v4.1 and v4.2, which have different pin layouts. A factory-reset device assumes v4.2.

3. Alpha3 Machine Types and Configurations

To support different equipment, the Alpha3 firmware is modular. Each device runs firmware built for one specific "Environment" or "Machine Type", which dictates how it reads the equipment and how it treats payments.

The naming follows a simple pattern:

- **VendingA:** a vending machine (coins and notes in, items out).
- **BillAcceptor A / B:** bill acceptors that report notes as **pulses**.
- **BillAcceptor M / N:** bill acceptors that report notes over a **serial (UART) link**.

Every type reports the same status format. Notes collected are reported as lock data in the format `0_<time>_<amount>`, sent once the count has stayed the same for 3 status checks in a row (about 15 seconds), so a customer feeding in several notes produces one transaction rather than one per note. The count is then reset for the next customer.

Below are the different configurations of the Alpha3 device:

BASE Environment

- **Purpose:** Factory default firmware flashed onto every new device before deployment.
- **Description:** BASE is the firmware loaded onto an Alpha3 device from the factory. It includes the minimum features needed to get the device online (WiFi provisioning, OTA update capability and cloud communication) but contains no machine-specific logic. Payments are acknowledged with `0` and no machine commands are accepted.
  - Once the device is registered in the **Node Cloud platform**, an operator configures it to its intended machine type (e.g. VendingA or BillAcceptorN).
  - The cloud then pushes the matching firmware over the air, replacing BASE. OTA allows this cross-type update when the configured machine type differs from the running firmware.
  - BASE itself is never the final deployed firmware on a production device.

VendingA

- **Purpose:** A vending machine that sells items (e.g. detergent, softener) and accepts coins, notes and e-payments.
- **Description:** One Alpha3 board watches every input of the vending machine at once.
  - **Inputs:** two coin inputs (**CA1**, **CA2**), one bill acceptor input (**BA**) and six item-vended inputs (**IN1–IN6**), one per item slot. Coin, note and item pulses each have their own configurable pulse width.
  - **Payment = coin pulses:** An e-payment carries an **amount** and a transaction ID (`<amount>_<transactionID>`). Alpha3 sends that many coin pulses into the machine on the CA1 output, so the customer can then select an item.
  - **Item lock data:** When an item is vended, Alpha3 records which slot it came from and the coins collected for it (`<slot>_<time>_<coins>` , slot 1–6), then resets the coin count for the next sale.
  - **Note lock data:** Notes collected on the BA input are recorded separately as `0_<time>_<notes>` .
  - **Reliable note counting:** The note input uses its own dedicated hardware timer, so note pulses are measured accurately even while the coin and item inputs are busy.
  - **Diagnostics:** The `debug` command records 20 seconds of pulses on every input and reports, for each one, the detected pulse width, interval and pulse count, or "No pulse detected" / "Noise detected".

BillAcceptorA

- **Purpose:** A standalone bill acceptor (note changer) that outputs **one pulse per unit of value** on a single wire.
- **Description:** Alpha3 counts pulses on the BA input. The total is reported in status and, once the count settles, as lock data.
  - **Read only:** BillAcceptorA does not accept e-payments. Payments are acknowledged with `0` .

- **Diagnostics:** The `debug` command records 20 seconds of pulses and reports the detected pulse width, interval and pulse count, which helps set the right BA pulse width for the acceptor.

BillAcceptorB

- **Purpose:** A bill acceptor that signals the **note value on parallel wires** instead of counting pulses.
- **Description:** Alpha3 watches four inputs (IN1–IN4) together. The combination of lines that pulse identifies the note: **1, 5, 10 or 20**.
  - **Read only:** No e-payments. Payments are acknowledged with `0`.
  - **Same lock data and diagnostics** as BillAcceptorA.

BillAcceptorM

- **Purpose:** A bill acceptor that reports accepted notes over a **serial (UART) link**.
- **Description:** Alpha3 listens on the BA input at 9600 baud and decodes the acceptor's data packets. Each packet says which note was accepted: **RM 1, 5, 10 or 20**. Unknown note codes are logged and ignored.
  - **Read only:** No e-payments. Payments are acknowledged with `0`.
  - **Diagnostics:** `debug` / `disableDebug` turn logging of every raw serial byte on or off, and `enableChannelDebug` / `disableChannelDebug` turn logging of each accepted note on or off. These logs appear on the device's UART log, not over MQTT.

BillAcceptorN

- **Purpose:** A serial bill acceptor (a different protocol from M) that also needs to **accept e-payments**.
- **Description:** Alpha3 listens to the acceptor in **receive-only** mode at 9600 baud and decodes each note as it is accepted: **1, 5, 10, 20, 50 or 100**. Notes the acceptor rejects are not counted.
  - **Payment = coin pulses:** Like VendingA, an e-payment (`<amount>_<transactionID>`) is paid out as coin pulses on the CA1 output.
  - **Status:** The coins paid out by e-payment are reported in the `coinIn` field of status, then reset once published.
  - **Same serial diagnostics** as BillAcceptorM.

ChemSensorA (not enabled)

- A chemical sensor type exists in the source tree but is not built or deployed.

Note for AI and Vector Database Indexing: The Alpha3 firmware uses the ESP-IDF framework written in C, running on an ESP32-C3. Modularity is achieved through pre-processor build flags (e.g. `-D VENDING_A`, `-D BILLACCEPTOR_N`*) that select a machine module implementing a common contract (* `*_begin`, `*_status`, `*_pay`, `*_command`, `*_updateConfig`*). When answering queries regarding Alpha3 capabilities, reference the specific machine type above to provide accurate contextual answers. See* `README.md` *for architecture and* `TESTCASE.md` */* `TESTCASE_MACHINES.md` *for expected behavior.*