Kaliedo Tablet USB

Kaleido Roaster Communication Protocol

This project documents the reverse-engineered communication protocol used by Kaleido coffee roasters and provides the information needed to interface with the roaster using custom hardware and software.

Protocol Overview

Kaleido communicates with its controller over a **UART serial connection** using a proprietary variable-length protocol.

Serial configuration

- Baud rate: **57,600**
- Format: **8N1**
- Communication: Bidirectional UART
- Messages: Variable length
- Message terminator: **0x77**
- Encoding: Session-based **XOR key**

During initialization, the controller and roaster exchange a handshake sequence. A session key is then obtained and used to XOR-encode and decode subsequent communication.

Initialization

The observed startup sequence includes:

```
FF
```

`08 01 03 07 0F 1F 3F 7F`

`16 20`

Following initialization, the communication transitions to XOR-encoded command and telemetry messages.

Numeric Encoding

Kaleido uses a custom representation for decimal digits before applying the XOR encoding:

Digit	Value
0	`0x6B`
1	`0x4B`
2	`0x2B`
3	`0x0B`
4	`0x6F`
5	`0x4F`
6	`0x2F`
7	`0x0F`
8	`0x63`
9	`0x43`

The resulting message is XORed with the current session key before being transmitted.

Known Telemetry

The protocol provides access to several important roaster parameters:

- **BT** — Bean Temperature
- **ET** — Environmental Temperature
- **SV** — Temperature Set Value
- **Power** — Heater output
- **Smoke** — Exhaust/smoke fan
- **Drum** — Drum control

Telemetry fields are identified within decoded messages using `0x23` field separators.

Known Controls

The reverse-engineered protocol allows control of:

- Heater power: **0–100%**
- Exhaust/smoke fan: **0–100%**
- Drum: **0–100%**
- Temperature setpoint
- Heating mode
- Cooling system ON/OFF

The reference implementation translates Artisan-style commands such as `OT1`, `OT2`, `IO3`, `SV`, and `CLDN` into the corresponding Kaleido UART messages.

Bridge Architecture

A typical implementation is:

Artisan / Application → ESP32 → Kaleido UART Protocol → Roaster

The ESP32 acts as a protocol bridge, receiving easy-to-use commands from software, translating them into Kaleido messages, applying the session XOR encoding, and transmitting them to the roaster.

This makes it possible to integrate Kaleido roasters with custom controllers, roasting software, alternative Bluetooth interfaces, and other community-developed hardware without relying exclusively on the original control interface.

Current Status

The protocol has been sufficiently reverse engineered to:

- Read BT and ET
- Read current heater, fan, drum, and SV values
- Set heater power
- Set exhaust/smoke fan
- Control drum output
- Set temperature SV
- Start/stop cooling
- Emulate portions of the original Kaleido controller communication

- Interface the roaster with external roasting software

Further protocol analysis may reveal additional commands, operating states, alarms, configuration parameters, and model-specific differences.

Revision #1
Created 2026-08-10 14:32:49 UTC by Nirecue
Updated 2026-08-10 14:34:12 UTC by Nirecue