LK-29 Software Specification

Geiger Counter & Environment Monitor

Version 1.4.0 | July 2026 | dedeideas.eu | Lubomir Karlik sr.

This document describes the LK-29 firmware architecture as implemented in the current MicroPython project. Version 1.3.5 adds station altitude handling and converts the BME688 absolute pressure reading to sea-level pressure immediately after each sensor read.

1. Scope and Features

- Ionizing radiation measurement with a GM tube, high-voltage regulation, CPM and dose-rate calculation.
- BME688 environmental sensing: temperature, humidity, absolute station pressure, gas resistance and IAQ.
- Sea-level pressure calculation using STATION_ELEVATION_M; the raw sensor value remains available as PRESSURE_RAW.
- Altitude lookup from OpenTopoData SRTM90m when the configured station coordinates change or when altitude is missing.
- Responsive web dashboard with metric tiles, two draggable dual-axis charts, 24-hour in-RAM history and weather forecast.
- 2.9 inch three-color e-paper display, rotated 180 degrees, with SK/EN localization and screenshot export.
- OTA ZIP firmware upgrade, live web console, Wi-Fi fallback AP, Basic Auth configuration panel and InfluxDB logging.

2. Hardware Platform

Item	Value
MCU	ESP32-S3 Zero, 240 MHz dual-core, 8 MB PSRAM, 4 MB flash
Display	Waveshare 2.9 inch e-Paper B V3, 128 x 296 px, black/red/white
Radiation detector	XERAM 3G8B GM tube pulse input on GPIO1 (wide-range industrial tube)
Environmental sensor	BME688 on I2C address 0x77; fallback detection also supports SHT3x class sensors
High voltage	PWM controlled boost converter, target output typically 400-450 V

Item	Value
Buzzer	Passive buzzer on GPIO10, 3 kHz PWM tick per GM pulse plus alarm sequences
Power	USB-C 5 V or Li-Ion battery plus boost converter
Network	Wi-Fi 802.11 b/g/n, 2.4 GHz

2.1 Pin Assignment

GPIO	Function
1	GM tube pulse input
2	Battery voltage divider ADC
3	SPI MOSI for e-paper
4	SPI CLK for e-paper
5	Display reset
6	HV ADC feedback
7	PWM output for HV regulator
8	I2C SCL for BME688
9	I2C SDA for BME688
10	Buzzer PWM output
11	Display BUSY input
12	Display DC
13	Display CS
16	SPI MISO, currently unused

3. Runtime Architecture

The firmware runs on MicroPython. main.py performs bootstrapping, Wi-Fi, time sync, module initialization and the main measurement loop. Sensor and GM data are held in config.py runtime variables and then exposed through the dashboard, API, display, history buffer and InfluxDB writer.

Module	Responsibility
main.py	Boot sequence, Wi-Fi connection, NTP, GeoIP, HTTP server and main loop orchestration.

Module	Responsibility
config.py	Persistent configuration and live runtime values, including PRESSURE_RAW, PRESSURE and STATION_ELEVATION_M.
reg.py	GM IRQ handling, rolling CPM, dose-rate conversion and HV PI regulation.
buzzer.py	Per-pulse tick and alarm signaling on GPIO10.
bme680.py	BME688 low-level driver, forced mode reads and gas register access.
i2c.py	I2C scan, BME688/SHT reads, IAQ calculation and pressure sea-level conversion.
elevation.py	OpenTopoData SRTM90m altitude lookup and station-to-sea-level pressure conversion.
display.py	E-paper composition, localization, rotated rendering and BMP screenshot buffer.
history.py	24-hour RAM circular buffer for chart series.
http_server.py	HTTP routing, static assets, authentication decisions and file transfer.
http_api.py	REST handlers, configuration save/set operations, OTA endpoints and console endpoints.
http_cfg.py	Runtime mirror of configuration fields shown in the web configuration panel.
influx.py	Buffered InfluxDB 1.x/2.x line protocol writer with id and ip tags.
openweather.py	OpenWeatherMap current weather and forecast client.
openmeteo.py	Open-Meteo current weather and forecast client.
geoip.py	Optional IP-based location detection.
ota.py	ZIP extraction, CRC validation support and atomic firmware file replacement.
console.py	Live console stream via client-side polling (/api/console/recent); the server optionally also exposes an SSE stream (/api/console/stream) that the current client does not use.

4. Configuration Model

Configuration is stored in config.py and edited through /config. The form layout is described by html/http_server.json, while http_cfg.py mirrors live read-only and editable values for the UI.

Variable	Meaning
FIRMWARE_VERSION	Current firmware string: ESP-32 S3 Zero v1.3.5.
OWM_LAT / OWM_LON	Station coordinates used by weather providers and altitude lookup.

Variable	Meaning
STATION_ELEVATION_M	Station altitude above sea level in meters; fetched from OpenTopoData SRTM90m when needed.
PRESSURE_RAW	Absolute pressure from the BME688 at station altitude, in hPa.
PRESSURE	BME688 pressure reduced to sea level, in hPa; this is the value shown and logged.
I2C_INTERVAL	Sensor read interval in seconds.
CPM_PER_USV	Tube calibration factor for dose-rate conversion.
TARGET_VOLTAGE / HV_DIVIDER_RATIO	High-voltage regulator target and feedback scaling.
OWM_ENABLED / OM_ENABLED	Weather provider selection. OpenWeatherMap requires an API key; Open-Meteo does not.
INFLUX_*	InfluxDB endpoint, bucket/database, token, measurement, interval and buffer settings.
AUTH_PASSWORD	HTTP Basic Auth password for protected pages and actions.
LANG	UI/display language key such as sk or en.

4.1 Altitude Refresh Policy

During /api/save, http_api.py checks whether OWM_LAT or OWM_LON changed, or whether no valid STATION_ELEVATION_M exists. If so, it calls elevation.refresh_config_elevation(). The lookup uses https://api.opentopodata.org/v1/srtm90m with the configured latitude and longitude. Save continues even if the lookup fails; the returned JSON contains the warning so the user can retry or enter altitude manually.

5. Sensor Data Flow

1. main.py periodically calls i2c.read_bme688_values() according to I2C_INTERVAL.
2. The BME688 is triggered in forced mode and returns temperature, humidity, pressure and gas resistance.
3. i2c.py stores the raw absolute pressure in config.PRESSURE_RAW.
4. i2c.py calls elevation.pressure_to_sea_level(raw_hPa, temp_C, STATION_ELEVATION_M).
5. The converted sea-level pressure is rounded to 0.1 hPa and stored in config.PRESSURE.
6. IAQ is calculated from gas resistance, adaptive baseline and mild outdoor temperature/humidity compensation.

Limitation: BME688 gas/IAQ sensing is intended primarily for indoor air-quality estimation. It is not a particulate-matter sensor and does not report PM1, PM2.5, PM10, dust, pollen or aerosol concentration. Because outdoor humidity, temperature swings, airflow and sensor ageing affect gas resistance, LK-29 treats IAQ as an indicative trend value only, not as a calibrated air-pollution or health-safety measurement.

7. The HTTP API, e-paper display, dashboard, history.py and influx.py read config.PRESSURE, not PRESSURE_RAW.

5.1 Sea-Level Pressure Formula

The station pressure is reduced to sea level with a barometric approximation using altitude in meters and the measured BME688 temperature. In code, the implementation lives in elevation.pressure_to_sea_level(). For the current station altitude of 268.0 m, a raw pressure near 983 hPa becomes approximately 1015 hPa.

Item	Value
Input	pressure_hpa from BME688, temp_c from BME688, elevation_m from STATION_ELEVATION_M
Fallbacks	If temperature is missing, 15 deg C is used. If altitude is missing, 0 m is used.
Output	Sea-level pressure in hPa, used as config.PRESSURE.
Rationale	Weather forecasts and public meteorological pressure values are normally reported at sea level.

5.2 IAQ Calculation

- Gas resistance is read from BME688 gas registers after a forced-mode gas measurement.
- IAQ_GAS_BASELINE is adjusted by controlled quiet-window normalization; cleaner-air samples are not accepted immediately unless the stability and humidity conditions are satisfied.
- The primary ratio is log10(baseline / gas); IAQ is mapped to roughly 10..500.
- Outdoor compensation clips humidity to 30-85 percent and temperature to 5-35 deg C.
- Compensation coefficients are intentionally mild and are applied only above the configured ratio threshold.
- IAQ_TEXT is localized from lang_sk.json or lang_en.json.

The web UI and documentation therefore present IAQ as an orientation value. For particulate pollution or regulatory outdoor air-quality monitoring, an additional PM sensor is required.

6. Web Dashboard

The dashboard at / is implemented by html/dashboard.html and html/dashboard.css. It polls /api/data, loads /api/history once at startup, and renders metric tiles, a wind compass, forecast cards and two charts.

- Displayed pressure is sea-level pressure from config.PRESSURE.
- Metric tile backgrounds use value-dependent colors at about 20 percent intensity.
- Chart configuration is drag-and-drop: users drag metric tiles to the left or right axis of a chart.
- Dashboard drag-and-drop is implemented with Pointer Events and coordinate-based drop-zone detection. Native HTML5 drag is disabled for metric tiles to avoid unreliable browser drop delivery on touch browsers.
- A long-press timer (approximately 260 ms) starts drag mode on touch devices before the browser context-menu gesture can cancel the interaction. The context menu and WebKit touch callout are disabled on metric tiles.
- Clicking or tapping metric tiles displays an in-tile 24-hour min/max overlay for 7 seconds. The overlay uses the same history arrays as the charts and includes metric units.
- Forecast icon sizing is recalculated from actual tile geometry after resize/orientation changes. Resize work is debounced to avoid Chrome main-thread stalls during window resizing.
- Chart preferences are stored per browser, while history samples are stored in ESP32 RAM.
- The history buffer keeps up to 1440 samples, one point per minute, and is cleared after ESP32 restart.
- The forecast can come from OpenWeatherMap or Open-Meteo, based on configuration.

6.1 Dashboard Metrics

Metric tiles are clickable. For metrics backed by history arrays (temperature, humidity, pressure, radiation and IAQ), the tile can temporarily show the minimum and maximum value in the visible 24-hour interval. Wind is excluded because no wind history series is stored.

The min/max overlay is absolutely positioned, does not change tile layout, and dynamically fits label/value/unit typography to the tile dimensions.

Metric	Source
Temperature	config.TEMPERATURE from BME688/SHT sensor.
Humidity	config.HUMIDITY from BME688/SHT sensor.
Pressure	config.PRESSURE, sea-level pressure derived from BME688 raw pressure.
Radiation	Dose rate from reg.py, based on rolling CPM and CPM_PER_USV.
IAQ	config.IAQ and config.IAQ_TEXT from BME688 gas algorithm.

Metric	Source
Wind	Weather provider current wind speed and direction.

7. HTTP Interface

Wi-Fi configuration endpoints:

- /api/connect validates parameters, sends an immediate JSON response with status=connecting, closes the client socket, then performs the blocking STA connection attempt. The result is stored in config.CONNECT_STATUS.
- /api/connect_status returns connecting, success:<ip> or failed. On success or failure, the endpoint clears CONNECT_STATUS and restarts the device after responding.
- Saved Wi-Fi passwords are reused only when the requested SSID matches the saved SSID in wifi_config.json. For a different SSID, the API returns an error requiring a new password.
- If wifi_config.json is missing or contains no SSID, WiFiManager.connect() returns False and main.py starts the GeigerCounter fallback AP with the setup portal.

Endpoint	Purpose
/	Dashboard.
/config	Authenticated configuration panel.
/wifi	Wi-Fi setup and captive portal page.
/ota	Authenticated OTA upgrade page.
/console	Live console page.
/screenshot	BMP screenshot of the e-paper framebuffer.
/api/data	Current values, dashboard labels, status and weather summary.
/api/history	In-RAM chart history arrays: temp, hum, pres, rad, iaq and timestamps.
/api/set?KEY=VALUE	Authenticated single-field configuration update.
/api/save	Authenticated persistence of configuration; refreshes altitude when required.
/api/i2c_scan	Authenticated I2C bus scan.
/api/weather_fetch	Authenticated immediate weather refresh.
/api/db_buffer	InfluxDB buffer state and optional debug window.
/api/config/download	Download current config.py.
/api/ota/upload	POST ZIP firmware archive with CRC header.

Endpoint	Purpose
/api/ota/run	Run OTA extraction with streaming progress.
/api/ota/status	OTA progress snapshot.
/api/console/recent	Recent console lines since sequence number.
/api/console/stream	Streaming console output.
/api/set_password	Authenticated password update.

8. Configuration Panel

The panel is generated from html/http_server.json. It includes live measurements, Wi-Fi status, I2C bus tools, system information, weather settings, HV/regulator settings, localization, InfluxDB and actions. Version 1.3.5 adds an editable Altitude [m] field mapped to STATION_ELEVATION_M.

- Live Measurements shows Temperature, Humidity, Sea-level Pressure and IAQ.
- Weather Settings include latitude, longitude and altitude.
- Saving settings persists values into config.py through targeted rewriting.
- Chart setup is intentionally not part of the panel; chart axes are configured by drag-and-drop on the dashboard.

9. E-paper Display

display.py renders the Waveshare panel in three colors and rotates the final image by 180 degrees to match the physical installation. The pressure row uses config.PRESSURE, therefore it shows sea-level pressure.

- Header shows DEVICE_NAME and firmware version.
- Wi-Fi row shows SSID and IP address or a localized warning.
- Measurement rows show temperature, humidity, pressure, dose rate and IAQ when available.
- Fonts include ASCII plus degree and micro symbols.
- The /screenshot endpoint generates a BMP snapshot of the current display framebuffers.

10. Radiation and High-Voltage Control

- GPIO1 rising-edge IRQ increments the GM pulse counter.
- micropython.schedule() is used to trigger buzzer work outside IRQ context.
- CPM is calculated as a rolling count over the last 60 seconds.

- Dose rate in uSv/h is CPM divided by CPM_PER_USV.
- The HV regulator uses ADC feedback, TARGET_VOLTAGE and a PI controller with deadband and duty limits.
- When CPM_ALARM_THRESHOLD is exceeded, the acoustic alarm starts.

11. Weather Providers

Provider	Protocol	API Key	Forecast
OpenWeatherMap	HTTP/HTTPS depending on configured endpoint	Required	5 days / 3 hours
Open-Meteo	HTTPS	Not required	Up to 16 days / hourly data

Open-Meteo current pressure from the provider is kept as weather-provider pressure. The local dashboard pressure tile uses the locally measured BME688 sea-level pressure, not the forecast-provider pressure.

12. InfluxDB Logging

- InfluxDB writes contain measured values only: temperature, humidity, pressure, radiation and IAQ.
- pressure is config.PRESSURE, meaning sea-level pressure in hPa.
- Each record includes id, ip and location-related tags where configured.
- The RAM ring buffer capacity is INFLUX_BUF_SIZE; failed writes remain pending until a later flush.
- Both InfluxDB 1.x database/write endpoint and InfluxDB 2.x bucket/org/token configuration are supported.

13. OTA and Console

The OTA flow uploads a ZIP archive to /api/ota/upload, validates CRC32 when supplied, stores the selected diff/full mode, and extracts files through /api/ota/run with streaming progress. Differential upgrades use the saved /ota_manifest.json from the last successful OTA, matching the LK-30 approach. The web console exposes recent log lines and a live stream for diagnostics without a serial terminal.

14. Boot Sequence

8. ESP32-S3 boots MicroPython and runs main.py.
9. config.py is imported and runtime defaults are initialized.

10. Wi-Fi connects or starts AP fallback with captive portal.
11. NTP and optional GeoIP lookup run.
12. HTTP server starts and static pages/API routes become available.
13. E-paper display initializes and renders startup status.
14. I2C scan identifies BME688/SHT devices.
15. Main loop starts sensor reads, GM/HV updates, display refresh, weather updates and InfluxDB flushes.

15. Version 1.3.5 Change Summary

- Web UI: /config, /wifi, /ota and /console share a unified blue palette (#04223d / #0a2f52 / #1565C0) with per-page dark/light theme; the dashboard theme is unchanged.
- Dashboard: robust pointer-based chart drag-and-drop with long-press support and context-menu suppression on metric tiles.
- Dashboard: metric tiles can show 24-hour min/max values with units for 7 seconds after tap/click.
- Dashboard: forecast icon and temperature sizing is recalculated after resize/orientation changes with debounced scheduling.
- Wi-Fi setup: /api/connect now returns immediately and the setup page polls /api/connect_status, avoiding browser fetch failures during STA connection attempts.
- Wi-Fi setup: saved passwords are only reused for the same SSID; ESP32-S3 remains limited to 2.4 GHz Wi-Fi networks.
- Added STATION_ELEVATION_M to config.py and the web configuration panel.
- Added elevation.py for OpenTopoData SRTM90m lookup and pressure conversion.
- Added PRESSURE_RAW as the absolute BME688 pressure at station altitude.
- Changed config.PRESSURE to store sea-level pressure after each BME688 read.
- Dashboard pressure tile, charts, e-paper display, history buffer and InfluxDB records now use sea-level pressure.
- The configuration save path refreshes altitude when coordinates change or altitude is not available.
- Firmware version string updated to v1.3.5.

16. Verification Notes

- Python syntax checks passed for the modified firmware modules.
- html/http_server.json was parsed successfully as JSON.
- At STATION_ELEVATION_M = 268.0 m, measured pressure around 983 hPa converts to approximately 1015 hPa.
- This specification is regenerated from the current project state for firmware 1.3.5.

v1.3.5 Addendum - IAQ Gas Normalization and 24 h Trend

Different BME688 modules can report gas resistance values that differ by orders of magnitude in the same clean air. The firmware therefore uses GAS_NORMALIZED so readings remain comparable after sensor replacement and between LK-29 and LK-30 devices.

The normalization formula is: GAS_NORMALIZED = GAS_RESISTANCE * GAS_NORM_TARGET / IAQ_CLEAN_BASELINE. The default GAS_NORM_TARGET is 250000 ohm. After 24 hours of stable clean air, IAQ_CLEAN_BASELINE is treated as the reference for the installed sensor.

A gas measurement is trusted only when BME688 reports gas_valid=True, heat_stab=True and the resistance is within 1000 to 50000000 ohm. Invalid or unstable readings must not update the IAQ baseline, clean-air vector, or trend state.

After IAQ_CLEAN_BASELINE changes, the firmware collects a new independent 24 h clean-air vector in iaq_clean_vector.json. The current 24 h normalized-gas window is then compared with that reference vector using both level change and curve shape.

A normalized-gas drop above 10% is treated as slow air-quality degradation. A smaller drop with a very similar curve shape may be marked as likely sensor drift. The status is available through IAQ_TREND_STATUS, IAQ_TREND_DELTA_PCT, IAQ_TREND_LEVEL_RATIO, and IAQ_TREND_SHAPE_ERROR_PCT.

Measurement history and diagnostics also include gas_normalized, gas_adc, gas_range, gas_valid, heat_stab, and gas_history_trusted. These fields help separate a real gas event from an unstable or incorrectly read BME688 measurement. Only trusted gas_normalized samples are allowed to update the clean-air vector and IAQ trend state.

BME688 is primarily intended for indoor environmental sensing. It does not measure particulate matter PM1/PM2.5/PM10, so IAQ must be treated as an indicative value, not as a certified air-quality measurement or a safety gas detector.

v1.3.5 Addendum - Bosch Gas Resistance Compensation and IAQ Curve Smoothing

Gas resistance conversion from the raw gas_adc/gas_range registers used a simplified estimate, var1 = 262144 >> gas_range, which ignored the per-chip range_switching_error calibration byte (register 0x04, already read into sensor._sw_err in bme688.py) and the Bosch-calibrated lookup tables for the 16 gas ranges. For certain gas_range values this simplified estimate deviated significantly from the chip's actual calibration, so every time the sensor's own autoranging switched range (a normal part of BME688 operation) the computed resistance jumped by a real, repeatable amount - visible as regular "teeth" on the IAQ/Gas history chart. This was not sensor noise but an algorithmic error.

Fix: gas resistance is now computed with the official Bosch formula using the LOOKUP_TABLE_1/LOOKUP_TABLE_2 tables and sensor._sw_err - the same formula already present (but never called, due to its blocking read) in the gas property in bme688.py.

Related second fix: the IAQ curve interpolation in _iaq_from_gas_ratio() now runs over log10(baseline/gas ratio) instead of the raw ratio. The first curve segment (ratio 1.0 to IAQ_RATIO_100, default 2.0) previously had a several-times steeper slope than later segments, amplifying ordinary sensor noise in that region into visible "teeth" and an uneven IAQ decline. IAQ values at the configured breakpoints (IAQ_RATIO_100..500) remain unchanged; only the shape of the curve between them changes.

Consequence for existing devices: the stored IAQ_GAS_BASELINE and IAQ_CLEAN_BASELINE values were calibrated under the old formula, so after this fix they temporarily no longer match the new (more accurate) resistance scale until they are re-settled by automatic normalization or a manual service baseline calibration.

Both fixes were originally identified and deployed on the parallel LK-30 airQ project (same BME688 fingerprint/IAQ foundation) and subsequently ported to LK-29.

v1.3.5 Addendum - OTA Manifest and Trusted Gas History

OTA differential upgrade now follows the LK-30 model. After a successful upgrade, the firmware stores /ota_manifest.json with the complete file list of the installed release. The /api/ota/manifest endpoint returns this saved manifest instead of walking the flash filesystem and calculating CRC32 for every file before each upload. If no manifest exists yet, the browser automatically uses a full upload; subsequent upgrades can upload only changed files.

The browser sends X-OTA-Mode: diff or full during /api/ota/upload, and the selected mode is kept in the OTA status for /api/ota/run. OTA_CLIENT_IDLE_TIMEOUT_MS and OTA_RUN_TIMEOUT_MS define the browser-side idle tolerance and server-side run timeout. Persistent local files, Wi-Fi settings, password, InfluxDB/OpenWeatherMap configuration and IAQ calibration values remain preserved across the upgrade.

IAQ trend handling was tightened. The runtime flag GAS_HISTORY_TRUSTED marks whether the current BME688 gas sample may feed gas_normalized, the clean-air vector and long-term trend detection. Clean-pulse samples, post-pulse settling samples, ADC-saturated values and readings with invalid gas_valid/heat_stab status remain visible as diagnostics, but they do not update the IAQ baseline, clean-air vector or trend state. /api/data exposes gas_history_trusted for diagnostics.

v1.3.6 Addendum — reactive clean-pulse, XERAM 3G8B tube calibration, and UI updates

The periodic clean-pulse (fixed schedule every N cycles) was replaced with a reactive one: the sensor now stays at a single constant heater temperature (BME688_IAQ_HEATER_TEMP_C) all the time, and a higher 400°C pulse only fires after BME688_CLEAN_PULSE_SATURATION_STREAK (default 3) consecutive saturated readings (gas_adc ≥ 1023), not on a timer. The fixed schedule caused a periodic “tooth” in the IAQ graph at every pulse even though the pulse itself was excluded from history, because the sensor needs several cycles to thermally/chemically resettle afterwards.

The post-pulse exclusion window (BME688_CLEAN_PULSE_SETTLE_CYCLES) was increased from 4 to 8 cycles based on real operating data showing a variable 4-to-7-cycle recovery time.

The IAQ_TEXT classification was changed to: ≤100 Excellent, ≤200 Good, ≤250 Fair, ≤300 Poor, ≤400 Bad, >400 Very bad.

The Radiation tile and chart color thresholds were rescaled: green (safe) up to 1 µSv/h, orange (elevated) up to 10 µSv/h, red (high) above 10 µSv/h.

The actually installed GM tube was identified as XERAM 3G8B (the document and config.py had previously assumed J305/STS-5/SBM-20). From the tube's datasheet (“1 stroke/sec per mR/h”) the calibration constant CPM_PER_USV = 60 / 8.76 ≈ 6.85 (1 R = 8.76 mSv) was derived, replacing the previous value of 114 carried over from J305. The editable range of the CPM per 1 uSv/h field on the configuration page was widened from 50–200 to 1–200.

The web console (/console) had a duplicate-log-line bug caused by overlapping concurrent requests to /api/console/recent during a brief device stall. A guard against overlapping requests was added.

In the OTA update log (/ota), the word “mtime” was replaced with “timestamp” in change-reason messages — a cosmetic text change only; the comparison logic (size + CRC32) is unchanged.

v1.3.7 Addendum — non-blocking (tick/flag) architecture rewrite

The firmware's core modules were rewritten from blocking calls to a cooperative tick/flag design, triggered by direct measurement: an e-paper refresh was found to block the single main loop (and with it, the HTTP server) for 2.5-3.9 seconds, with I2C and InfluxDB writes blocking it further. i2c.py (BME688 reads), display.py + epaper2in9bv3.py (e-paper refresh), http_server.py (the HTTP accept loop) and influx.py (InfluxDB writes) no longer run a single long blocking call - each advances through a handful of small, bounded steps (register read/write, deadline check) driven by a shared software timer.

Added soft_timer.py, ported from the sibling LK-30 airQ project: one hardware timer (Timer 2, the only one free - reg.py and buzzer.py already own the other three) dispatches many independent countdown tasks via micropython.schedule(), so no module needs its own hardware timer ID. A scheduling-queue-

full edge case that could silently and permanently stall every tick-driven module (i2c/display/influx/http all frozen with no error printed) was found and fixed during this work.

main.py's main loop now services HTTP only when a periodically-armed flag says it is due (HTTP_ACCEPT_POLL_MS=50ms, HTTP_ACCEPT_BURST=3 config keys, same pattern as LK-30 airQ) instead of every loop pass, and calls machine.idle() rather than a fixed sleep when nothing needs attention, letting the CPU rest between events instead of busy-polling.

Wi-Fi power-save (modem-sleep) is now explicitly disabled (WLAN.PM_NONE). It was buffering incoming packets until the next DTIM beacon, producing a sawtooth-shaped latency pattern (visible even in plain ICMP ping) once the main loop began actually idling between events instead of busy-looping.

Wi-Fi now sticks to the previously-associated access point (WIFI_CONNECT_BY_BSSID) across reboots, but automatically switches to a different AP broadcasting the same SSID if it is reachable with a clearly stronger signal (WIFI_BSSID_UPGRADE_MARGIN_DB, default 5 dB headroom to avoid flip-flopping) and remembers the new BSSID in wifi_config.json.

Added a lightweight /api/dashboard_data endpoint. The four dashboard pages (dashboard.html, dashboard1/2/3.html) previously polled the same /api/data endpoint used by the /config admin page every 15 seconds - a payload that also reflects every editable setting, RAM/flash diagnostics, Wi-Fi status and the I2C scan result. The new endpoint returns only the handful of values the dashboards actually display.

Fixed a server-clock synchronization bug: the browser's live clock and "next refresh" time could run about two hours fast. The device's RTC is deliberately kept at local time (UTC + TIMEZONE_DEV + DST_DEV), but the Unix timestamp sent to the browser for clock sync (server_ts) was read from that same local-shifted RTC and treated as if it were already UTC - when a browser in the same time zone combined it with its own UTC-based clock, the timezone/DST offset was counted twice.

Fixed a NameError in the BME688 gas-resistance debug log path (introduced while splitting the read into tick/flag steps) that silently aborted every measurement cycle into an error state whenever DBG_I2C was enabled, so no fresh sensor value was ever stored.

The one-time post-boot BME688 burn-in (400°C heater pulse) is now skipped on a software reset or OTA reboot if RTC memory still holds a marker from a previous, continuously-powered session - only a genuine power loss re-triggers it, since re-pulsing an already-conditioned sensor is unnecessary and was suspected of disturbing an otherwise stable gas reading.

The display's second boot-time refresh now waits at least DISPLAY_BOOT_DELAY_S (120 s default) after power-on before showing "real" sensor data, instead of refreshing on the very first successful BME688 read - the sensor's first read(s) right after burn-in are not representative (elevated temperature/IAQ).

Dashboard history charts now always show exactly 5 Y-axis value labels on both charts (previously an automatic 5-or-6 choice could differ between them), and a rounding fix prevents the top label from occasionally being clipped off the plot area.

Firmware version string updated to v1.3.7.

Addendum v1.3.8 — OTA package identity check and completion of the non-blocking architecture

config.py now defines the constants PROJECT_ID ("LK-29_meteo") and OTA_TARGET ("LK29"), which uniquely identify the project running on the device.

The packaging script make_LK29_zip.bat now automatically generates an ota_project.json file containing project_id/ota_target/firmware_version before creating the ZIP archive - following the same pattern used by the sibling LK-30 airQ project.

Before writing even a single file from an uploaded package, ota.py verifies that the embedded ota_project.json matches the running device's PROJECT_ID/OTA_TARGET. On a mismatch (e.g. an LK-30 airQ package accidentally uploaded to an LK-29 device) the upgrade is rejected with a message such as "OTA rejected: package target LK30 does not match device target LK29" and no file is written.

If the package does not contain ota_project.json at all (e.g. an older package build), the update is likewise rejected for safety - there is no way to bypass this check.

The browser-side differential (diff) upload logic (ota.html) was updated to always include ota_project.json among the uploaded files (even when unchanged), while excluding it from the file list stored in ota_diff_manifest.json.

The fifth and final phase of the non-blocking (tick/flag) architecture rewrite was completed: a review of http_api.py confirmed that every remaining blocking wait (sleep) belongs exclusively to the explicitly out-of-scope paths - Wi-Fi connection, device restart, and OTA upgrade - and requires no further changes.

During this review, unused code was removed: a Server-Sent-Events console-streaming endpoint that was never wired into the HTTP server's routing - the web console (/console) actually works exclusively via periodic polling (/api/console/recent).

Firmware version updated to v1.3.8.

Addendum v1.3.9 — smoother charts and more accurate temperature/pressure/humidity readings

Dashboard chart line rendering was changed from a Catmull-Rom spline to monotone cubic Hermite interpolation (Fritsch-Carlson). The curve still passes exactly through every real data point, but a

segment between two neighbouring points can no longer overshoot past their values - the previous approach of raising the spline's tension (up to 0.8) actually made sharp real fluctuations (e.g. radiation) look even more overshot, which showed up as spiky, jagged peaks.

The Y axis on every dashboard chart now always renders exactly 5 equal parts (6 grid lines/values) instead of the previous 5 values (4 parts).

The radiation chart's Y axis now has a minimum range of 5 µSv/h instead of the previous 0.04 µSv/h - at low real readings (typically 0.1-0.3 µSv/h) the axis no longer zooms in so far that it visually exaggerates the GM tube's statistical noise. A related bug was also fixed where this wider minimum range could previously push the axis into negative values - it now always starts at 0 and only expands upward.

BME688: temperature, pressure and humidity are now measured in a separate step BEFORE the chip is heated for the gas measurement, instead of together with it as before. Each read cycle first triggers a short forced-mode measurement with the heater off (config.BME688_AMBIENT_WAIT_MS, 160 ms by default), stores T/P/H from that pass, and only then starts the heating cycle for the gas measurement - that cycle's own T/P/H readings are no longer used. This removes the temperature bias (and the resulting bias in humidity and sea-level pressure) caused by the sensor's own gas-heater self-heating.

As a side benefit for accuracy: the res_heat register calculation for the heater now uses the freshly measured ambient temperature instead of the last published config.TEMPERATURE value (which could itself still be biased by a previous heating cycle).

Firmware version updated to v1.3.9.

Addendum v1.4.0 — chart averages, history persistence across restarts, and a clear-history action

Dashboard charts now show the average of the collected history samples (same 24h window as the chart) under the title, formatted as "x̄: value unit" for both displayed metrics - the text color follows whichever metric is currently assigned, even after dragging it to the other side of the chart.

The OTA update confirmation dialog (/ota) now has the title "Firmware upgrade".

On a controlled restart (the /config Restart button, an OTA update, or a Wi-Fi reconfiguration), the RAM measurement history is now saved to history.json together with a one-shot history.json.restore marker. On the next boot the history is automatically restored into RAM and the marker is removed - so a later uncontrolled reboot (crash, power loss) never picks up stale data again.

The /config page has a new "Maintenance" section with a button to permanently delete the saved history (both the RAM buffer and the file on disk), with a confirmation dialog before it runs.

Verified (no code change needed): the existing mechanism that skips the BME688 burn-in heating on a software restart (RTC marker, Addendum v1.3.8/v1.3.9) already guarantees that a controlled restart

(button or OTA) never triggers an unnecessary 400C heat-up - measurement resumes immediately at the ambient step. The time gap between the end of the last heating pulse and the next ambient (heater-off) reading was also verified - during normal 60s-cycle operation the gap is about 59 seconds, far more than the sensor's MEMS hotplate needs to cool down.

Firmware version updated to v1.4.0.