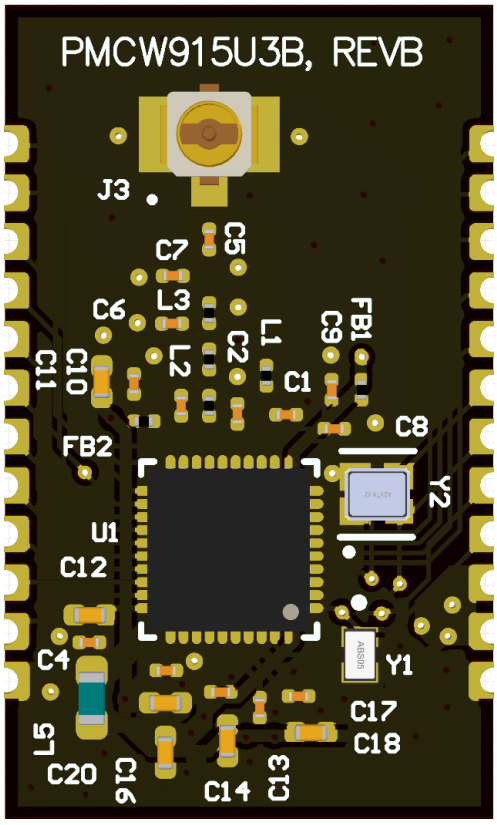
PMCW915U3B

915 MHz Wireless Sensor Gateway Module

Configuration Parameter Reference

UART protocol 1.3.0 · Document D3 · Issue 2.1 · May 2026

Power Micro Controls



Product: PMCW915U3B Radio Gateway Module **Protocol version:** 1.3.0 **Audience:** integrators configuring the module over UART

1. How configuration works

- Every setting is read with **`CFG_GET`** and written with **`CFG_SET`** (see D2). A write updates a **RAM working copy** and is validated immediately.
- **`CFG_COMMIT`** validates the whole set, writes it atomically to **NVM3** (on-chip non-volatile storage), and **applies it live**. Committed values **survive reboot**.
- Committed values do **not** survive a firmware update that changes the config store layout. The store is versioned, and adding the GPIO parameters in 1.3.0 bumped it to **layout version 2**: a module upgraded into this release comes up at factory defaults and must be reconfigured.
- **`CFG_RELOAD`** discards uncommitted edits; **`CFG_FACTORY_RESET`** restores the defaults below and commits them.

Persistence

`Persist = nvm3` → value is retained across power cycles once committed. `Persist = none` → read-only / derived values that are never stored.

Certification bounds

Parameters marked **Cert-bound = yes** are constrained to the frozen RF envelope, including the channel plan, the +20.0 dBm power ceiling and the fixed PHY frame length that the module is intended to be certified with. A value outside the allowed range is **rejected** (`ERR_RANGE` or `ERR_CERT_LIMIT`) and never stored or applied, so no host command can move the radio outside that envelope. The hard caps are compile-time constants in `include/host_protocol.h` (`SILADS_CH_MAX`, `SILADS_TXPOWER_MAX_DDBM`, `SILADS_PHY_FRAME_SIZE`).

The envelope is enforced today, but the module is **not yet certified in any country** and no emissions measurements exist.

2. Parameters

ID	Name	Acc	Range	Default	Units	Persist	Cert-bound	Description
0x0001	role	rw	0..1	1	0=slave,1=master	nvm3	-	Radio role. Was compile-time #define HOP_MASTER 1.
0x0002	operating_channel	rw	0..25	10	channel (ch N = 902 + N MHz)	nvm3	yes	Base operating channel. Was set_e3_channel(10). Bounded to the certified 902-927 MHz plan.
0x0003	tx_power_ddbm	rw	-100..200	200	deci-dBm	nvm3	yes	TX power. Clamped to the +20.0 dBm certified ceiling (rf_envelope.tx_power_max_ddbm).
0x0010	rx_forward_enable	rw	0..1	1	0/1	nvm3	-	Forward received sensor traffic to the host. When enabled the module emits SENSOR_DATA_IND / SENSOR_STATUS_IND for ANY valid sensor frame it demodulates on the operating channel — including replies to polls initiated by OTHER gateways. This is the multi-gateway redundancy mechanism: overlapping gateways all report what they hear; prioritization and dedup happen upstream (GD32/AWS). Was volatile rx_requested.
0x0011	decode_records	rw	0..1	1	0=raw page,1=decoded records	nvm3	-	SENSOR_DATA_IND payload format: raw 2048B page vs decoded sensor_record_t[].
0x0020	tx_delay_ms	rw	0..65535	500	ms	nvm3	-	Inter-transmit pacing delay. Was #define TX_DELAY 500.
0x0021	tx_timeout_ms	rw	0..65535	7000	ms	nvm3	-	Response timeout. Was #define TX_TIMEOUT 7000.
0x0022	no_response_retries	rw	0..65535	5	count	nvm3	-	Unanswered TX before SENSOR_NO_RESPONSE_IND. Was the 'no_response > 5' magic.
0x0040	hop_enable	rw	0..1	1	0/1	nvm3	-	Stored hopping flag; the hop engine is dormant in this release. The radio stays on operating_channel, and MOD_STATUS echoes this flag, so a value of 1 does not mean hopping is active.
0x0041	hop_table_len	rw	0..25	25	count	nvm3	yes	Number of valid entries in hop_table.
0x0042	hop_table	rw	0..25	0	channel per hop slot	nvm3	yes	Hop sequence; each entry a channel 0..25. Was hop_freq_table[25]. Bounded to the certified plan.
0x0043	hop_dwell_ms	rw	0..65535	400	ms	nvm3	yes	Dwell time per channel. Certification-relevant (FHSS).

0x0050	gpio_mode	rw	0..5	0	mode per pin (see GPIO_CONFIG)	nvm3	-	Power-on mode of each of the 12 header GPIOs, indexed by logical pin 0..11 (see gpio_map). 0=disabled/hi-Z is the factory default, so an unconfigured module never drives a customer net.
0x0051	gpio_out_default	rw	0..4095	0	bitmask, bit n = pin n	nvm3	-	Level applied at power-on to pins whose gpio_mode is an output. Commit this if a net must hold a defined level before the host can issue GPIO_WRITE.
0x0052	gpio_irq_mask	rw	0..4095	0	bitmask, bit n = pin n	nvm3	-	Input pins that raise GPIO_CHANGE_IND on an edge. 0 disables all change notification.
0x00F0	frame_size	ro	2096..2096	2096	bytes	none	yes	PHY fixed frame length. Read-only mirror of the certified PHY. Was #define PKT_SIZE 2096.
0x00F1	local_eui	ro	-		16 ASCII hex chars	none	-	This module's own 64-bit MCU unique ID as ASCII hex. Read-only (SYSTEM_GetUnique()).

3. Notes on specific parameters

- **`operating_channel`** — channel `N` maps to `902 + N` MHz (`N` = 0..25 → 902..927 MHz). Both the gateway and the sensor must agree on channel; when hopping is enabled this is the base/rendezvous channel.
- **`tx_power_ddbm`** — transmit power in **deci-dBm** (e.g. `200` = +20.0 dBm). Clamped to the `SILADS_TXPOWER_MAX_DDBM` ceiling. The realized EIRP also depends on the external PA and antenna gain, neither of which has been measured (see D1, where the RF and electrical characteristics are still TBD-CHAR). On a bench with two boards inches apart, set this well down — `-100` (−10.0 dBm) with the external PA off is the value used for bring-up.
- **Sensor addressing (no roster)** — as of protocol 1.2.0 the module holds no target roster. `SENSOR_*` opcodes carry the sensor's 16-character ASCII-hex EUI inline; enrollment/pairing and poll scheduling are host (GD32/AWS) responsibilities. Param ids `0x0030`/`0x0031` are retired and reserved.
- **`hop_enable` / `hop_table` / `hop_table_len` / `hop_dwell_ms`** — the frequency-hopping configuration. **The hop engine is not implemented in this release.** These parameters are accepted, validated and persisted, and `MOD_STATUS` reports `hop_enabled`, but the radio stays on `operating_channel` regardless. Every `hop_table` entry must still be within the channel plan (0..25), and `hop_dwell_ms` is certification-relevant (FHSS) — certify what actually ships, which today is a single-channel radio.
- **`decode_records`** — `1` makes `SENSOR_DATA_IND` carry decoded `silads_sensor_record_t` records (engineering-unit-ready per D7); `0` delivers the raw 2048-byte page for host-side decoding.
- **`gpio_mode` / `gpio_out_default` / `gpio_irq_mask`** — the persistent state of the twelve customer GPIOs on the module headers, all indexed by **logical pin 0..11** (the index map and the pin modes are in D2 §4.1). `gpio_mode` is a 12-byte array, one mode per pin, and defines what each pin *is* at power-on. `gpio_out_default` is a 12-bit mask giving the level applied at power-on to those pins whose mode is an output. `gpio_irq_mask` selects which input pins raise `GPIO_CHANGE_IND`; `0` disables change notification entirely.

These three behave like every other parameter: `CFG_SET` stages the value in the RAM working copy and validates it, and `CFG_COMMIT` persists it to NVM3 and applies it live. Committing an unrelated parameter does not disturb pin state — the pins are re-driven only when one of the three GPIO parameters actually changed in that commit, so live edits made with

`GPIO_CONFIG` / `GPIO_WRITE` are not silently overwritten. Conversely, those live opcodes are **not** persisted: only a commit of these parameters determines what the pins do after the next reset.

The factory default is `gpio_mode = 0` (disabled / high-impedance) on **every** pin, with `gpio_out_default = 0` and `gpio_irq_mask = 0`. An unconfigured or factory-reset module therefore never drives a customer net, and a net that must hold a defined level before the host is able to issue `GPIO_WRITE` needs those values committed. A stored blob carrying an out-of-range mode falls back to high-impedance rather than driving a pin from a bad value.

- **`frame_size` / `local_eui`** — read-only. `frame_size` mirrors the frozen PHY fixed length (2096) and cannot be changed. `local_eui` is this module's own identity from `SYSTEM_GetUnique()`.

4. Recommended bring-up sequence

```
MOD_INFO                          # confirm banner + protocol version 1.3.0
CFG_GET  local_eui                # note the module identity
CFG_SET  operating_channel = 10
CFG_SET  tx_power_ddbm     = 200  # use -100 on a bench with the units close together
CFG_SET  gpio_mode         = [...] # 12 bytes; leave 0 for any pin you do not use
CFG_COMMIT                        # persist + apply
MOD_STATUS                        # confirm operating_channel
SENSOR_STATUS "781C9DFFFE5CBB47"  # first poll of a watch (EUI inline, no roster)
```