TECHNICAL REPORT TR-2026-014

Firmware Watchdog Timeout Root-Cause Analysis

M. Takahashi Embedded Systems Group

August 27, 2026

Abstract. A field fleet of controller units began reporting intermittent watchdog resets roughly six weeks after a firmware update. This report documents the investigation, identifies the root cause as a priority-inversion deadlock between the sensor polling task and the communications task, and recommends a fix already validated in a 30-unit pilot.

1 Background

Beginning in week 3 after the v2.4.0 firmware rollout, field telemetry showed an elevated rate of unplanned watchdog resets: 0.4% of units per day, against a historical baseline of under 0.02%. No single site or hardware batch was implicated — affected units were spread across all four manufacturing lots and all deployment regions, which pointed toward a firmware rather than hardware cause from the outset.

2 Investigation

2.1 Reproducing the fault

The reset could not be reproduced on the bench under normal load, but was reproduced reliably (9 of 10 attempts) by holding the unit under sustained high sensor-sampling load while simultaneously saturating the communications bus with retransmit traffic — a combination that occurs in the field during storms, when both sensor noise and radio retries spike together.

2.2 Root cause

Trace logs from a reproduced failure show the sensor polling task (priority 3) blocked waiting on a mutex held by the logging task (priority 1), while the communications task (priority 2) was runnable and continuously preempting the logging task before it could release the mutex. This is a classic priority-inversion pattern: the high-priority sensor task is indirectly blocked by a low-priority task, without the RTOS's priority-inheritance mechanism engaging, because the mutex in question was created without the inheritance flag set.

Table 1: Task priorities and the deadlock path

Task	Priority	Role in the deadlock
Sensor polling	3 (highest)	Blocked on mutex
Communications	2	Continuously preempts logging task
Logging	1 (lowest)	Holds mutex, cannot run to release it

Once the sensor polling task is blocked long enough, the watchdog timer (which expects a periodic pet from that task) expires and resets the unit — explaining both the intermittency (it requires the specific load combination) and the delayed onset (a firmware change in v2.4.0 moved the mutex acquisition from the logging task's old lock-free ring buffer to the new mutex-protected one, introducing the priority-inversion window that did not exist before).

3 Fix and Validation

The fix creates the mutex with the priority-inheritance attribute set, so the logging task temporarily inherits the sensor task's priority while holding the mutex, preventing the communications task from starving it out.

- Deployed to a 30-unit pilot fleet for three weeks under the same storm-load conditions that reproduced the original fault on the bench.
- Zero watchdog resets observed in the pilot fleet during a period when the unpatched fleet's expected rate would have produced approximately 6 resets across 30 units.
- No measurable change to communications task latency, ruling out a regression in the other direction.

4 Recommendation

Roll the fix out fleet-wide as an over-the-air update, prioritising the storm-prone regions where the triggering load pattern is most common. No hardware changes are required.