You are the functional safety manager of a multicore controller development. Your engineers have split the Software on 2 cores, implemented a Real-Time Operating System, and tested all the system features. You have so far not been aware of the multi-core implementation. What are you Top 5 concerns and why?

As the functional safety manager, the multi-core implementation, especially when previously unaware of it, raises several critical concerns regarding ISO 26262 compliance and the safety of the system. My top 5 concerns would be:

1. **Freedom from Interference (FFI) and Dependent Failures:** The splitting of software across two cores, particularly if they handle functions with different ASILs, immediately triggers concerns about freedom from interference. ISO 26262-11:2018, 5.4.2.2 explicitly states that if multiple software elements with different ASIL ratings coexist in a multi-core context, a freedom from interference analysis according to ISO 26262-9:2018, Clause 6, is required. The exemplary faults in ISO 26262-6:2018, Annex D, concerning memory, timing, and execution interferences, are a starting point for this analysis (ISO 26262-11:2018, 5.4.2.2). A dependent failure analysis is crucial for identifying common cause failures, especially relating to shared hardware resources (ISO 26262-11:2018, 5.4.2.2 Note 1; ISO 26262-9:2018, Clause 7) and shared software resources (ISO 26262-11:2018, 5.4.2.2 Example 4; ISO 26262-9:2018, Clause 6). If the Real-Time Operating System (RTOS) is intended as a safety mechanism to ensure independence between software elements, it must be developed compliant with the highest ASIL of the separated elements (ISO 26262-11:2018, 5.4.2.2 Example 4).
2. **Timing Requirements and Faults in Multi-core Environment:** Multi-core components are known to be potentially subject to timing faults (ISO 26262-11:2018, 5.4.2.3). The software safety requirements must specify timing constraints (ISO 26262-6:2018, 6.4.2 e)), and an upper estimation of required resources, including execution time, is mandatory for embedded software (ISO 26262-6:2018, 7.4.13). The interactions between hardware and software in a multi-core setup can significantly influence performance and execution times (ISO 26262-6:2018, Table 10 Note c)). Dedicated analyses for timing faults and the implementation of adequate countermeasures (e.g., watchdogs, timing supervision units) are essential to prevent violations of safety goals (ISO 26262-11:2018, 5.4.2.3 Example 1, 2).
3. **Completeness and Adequacy of Safety Analyses (Hardware and Software):** The development of hardware at the item level requires the analysis of potential hardware faults and their effects, along with quantitative evaluations of the overall hardware architecture (ISO 26262-5:2018, 5). This includes evaluating hardware architectural metrics (ISO 26262-5:2018, Clause 8) and assessing the residual risk of safety goal violations due to random hardware failures (ISO 26262-5:2018, Clause 9). For software, safety analyses and dependent failure analyses are applied at the software architecture level (ISO 26262-6:2018, E.2.1). The fact that "all system features" have been tested does not guarantee that specific safety analyses related to multi-core interactions (e.g., race conditions, deadlocks, shared resource conflicts) have been sufficiently performed or that their results have driven design choices and verification.
4. **Consistency of Hardware-Software Interface (HSI) and Architectural Design:** The technical safety requirements are allocated to both hardware and software, leading to the refinement of the Hardware-Software Interface (HSI) specification (ISO 26262-5:2018, 6.2, 6.3.1). The software architectural design must describe static and dynamic aspects, including control flow, concurrency, data flow, and communication relationships, and their allocation to hardware (ISO 26262-6:2018, 7.4.5). Furthermore, the software unit design and implementation must be consistent with the HSI specification (ISO 26262-6:2018, 8.4.2 c)). The introduction of a multi-core architecture significantly complicates these aspects, requiring a re-evaluation of how software interacts with hardware resources and how shared

resources are managed at the interface level.

5. **Verification and Validation of Functional Safety Properties in the Integrated Multi-core System:** While system features have been tested, the verification activities must specifically provide evidence that the embedded software fulfills its safety requirements when executed in the target environment and contains neither undesired functionalities nor undesired properties regarding functional safety (ISO 26262-6:2018, Table 10, Objective 11). The integration steps must ensure that safety measures resulting from safety analyses at the software architectural level are properly implemented (ISO 26262-6:2018, Table 10, Objective 10 b)). Without prior knowledge of the multi-core implementation, there is a risk that existing verification plans and test cases are insufficient to uncover safety-critical behaviors arising from concurrency, resource contention, and timing variability inherent to multi-core operations.

Additional list of clauses used:

- ISO 26262-5:2018, Clause 5 General topics for the product development at the hardware level
- ISO 26262-5:2018, Clause 6 Specification of hardware safety requirements
- ISO 26262-5:2018, 6.2 General
- ISO 26262-5:2018, 6.3.1 Prerequisites
- ISO 26262-5:2018, Clause 8 Evaluation of the hardware architectural metrics
- ISO 26262-5:2018, Clause 9 Quantitative analyses of random hardware failures
- ISO 26262-6:2018, 6.4.2 e)
- ISO 26262-6:2018, 7.4.5
- ISO 26262-6:2018, 7.4.13
- ISO 26262-6:2018, 8.4.2 c)
- ISO 26262-6:2018, Table 10 (Objectives 10 b and 11)
- ISO 26262-6:2018, Annex D
- ISO 26262-6:2018, E.2.1
- ISO 26262-9:2018, Clause 6 Coexistence of elements
- ISO 26262-9:2018, Clause 7 Dependent failure analysis
- ISO 26262-11:2018, 5.4.2.2 Clarifications on Freedom from interference (FFI) in multi-core components
- ISO 26262-11:2018, 5.4.2.2 Note 1
- ISO 26262-11:2018, 5.4.2.2 Example 1
- ISO 26262-11:2018, 5.4.2.2 Example 4
- ISO 26262-11:2018, 5.4.2.3 Timing requirements in multi-core component
- ISO 26262-11:2018, 5.4.2.3 Example 1
- ISO 26262-11:2018, 5.4.2.3 Example 2

Correctness = 5P
Completness = 4P, -1P Safety related problems with scheduling are not explicitly mentioned e.g.: "**Timing and Scheduling Safety:** Ensuring that the RTOS scheduler reliably prioritizes and allocates CPU time to safety-critical tasks across cores. Scheduling-related issues such as priority inversion, preemption latency, or task starvation can lead to missed deadlines and potential hazard exposure"