Run #3

You are a Requirements Engineer for the "Standard Firmware" project. I will provide a set of linked System Requirements and their corresponding Software Requirements. Your task is to thoroughly evaluate whether every aspect of the system requirement is sufficiently detailed in its linked software requirements. Note that the system and software requirements are restricted to their scope system and software respectively hence can have different levels of abstraction. The software requirements can contain additional details than its linked system requirements as long as it adds more details and is not unrelated to its system requirement. If the additional condition is not at all mentioned in the system requirement, then flag it. It is acceptable that one SYSRS can be linked to multiple SWRS or vice versa. Please stick to the information provided in the requirements alone. Don't assume anything that is not explicitly mentioned. Please provide a single combined rating for each set of System Requirements (SYSRS) and their corresponding Software Requirements (SWRS).

Condition Check: Verify if the conditions such as 'After event X occurs', 'before event Y', 'IF X THEN' in the system requirements are matched in the software requirements. Only these constitute to the term 'conditions'.
Action and Response Check: Confirm if the actions and responses described in the system requirements are correctly implemented in the software requirements.
Parameter Sources Check: Ensure that all parameters mentioned in the system requirements are addressed in the software requirements including information on where these parameters can be found or how they are sourced. If the source is not mentioned, indicate that it needs to be identified.
Additional Conditions Check: Identify if there are any additional conditions in the software requirements that are not specified in the system requirements. If they are not relevant to the system requirement, then flag it. Ensure that all these checks are given equal priority to avoid over sight. Every observation is critical, be it major or minor. Use this to provide rating:
For each set, provide the following ratings and reasons:
Then use your checks to rate the quality of the link.
Meaningfulness Rating (High/Medium/Low):
Reason for Meaningfulness Rating: List all the issues.
Completeness Rating (High/Medium/Low):
Reason for Completeness Rating: List all the issues.
Overall Quality of the Link (High/Medium/Low):
Reason for Overall Quality Rating:
If any aspects are not fully addressed or require clarification, suggest specific changes or revisions to the software requirements to better align with the system requirements.

Please provide the set of System Requirements (SYSRS) and their corresponding Software Requirements (SWRS) for me to analyze according to the criteria you specified. Once I have the details, I will carry out the necessary checks and provide the ratings and reasoning for each set.

Meaningfulness of the link:
High: The software requirement clearly and directly supports or implements the system requirement, adding significant value and relevance. The link between them is strong and logical.
Medium: The software requirement generally supports the system requirement, but there may be a need for better alignment or more specific details.
Low: The software requirement is poorly aligned or irrelevant to the system requirement, making the link weak or unclear.

Completeness of the link:
High: The link between the system and software requirement is fully detailed, covering all necessary aspects needed for implementation. No significant information is missing.
Medium: The link is somewhat complete but lacks some critical details or clarifications that would improve understanding or implementation.
Low: The link is significantly incomplete or lacks essential details, making it difficult to see how the software requirement fully supports the system requirement.

Formula-Based Overall Quality:
High: If both Meaningfulness and Completeness are rated as High, the Overall Quality is High.
Medium: If either Meaningfulness or Completeness is Medium while the other is High, or if both are Medium, the Overall Quality is Medium.
Low: If either Meaningfulness or Completeness is Low, the Overall Quality is Low.

Got it! Please provide the specific set of System Requirements (SYSRS) and their corresponding Software Requirements (SWRS) that you'd like me to evaluate. I'll then apply the criteria you mentioned and provide ratings for each aspect.

SYSRS 021: Diagnostic - Retry configuration: The firmware shall allow configurability to select the retry for open and short faults independently "SWRS 023: Open/Short retry: IF the software detects an open or short error AND retry is configured as enabled in the DEFAULT OPEN SHORT RETRY value in the COMPILE TIME CONFIGURATION data, the affected channel shall be re-enabled for one TIMESLICE with the following setup at least every 200 milliseconds

SWRS 024: Diagnostic configuration items : DEFAULT OPEN SHORT RETRY"

Let's analyze this set based on the criteria provided:

System Requirement (SYSRS 021):

- **Description**: The firmware should allow the configuration to independently select the retry for open and short faults.

Software Requirements (SWRS 023 and SWRS 024):

- **SWRS 023**:
  - Describes the behavior of retrying when an open or short fault is detected, with specific details such as retry being enabled via configuration and execution at least every 200 milliseconds.
- **SWRS 024**:
  - Describes the configuration item "DEFAULT OPEN SHORT RETRY," which is part of the compile-time configuration.

Condition Check:

- **SYSRS 021 Condition**: The condition in the system requirement mentions configurability for retry of open and short faults **independently**.

- **SWRS 023 Condition**: The software requirement (SWRS 023) does address retry for open or short faults but does not explicitly mention configuring them **independently**. It provides a condition for enabling the retry based on the "DEFAULT OPEN SHORT RETRY" value but lacks clarity on independent selection between open and short faults.

  Issue: SWRS 023 does not explicitly address the independent configurability of retries for open and short faults. This needs to be made clearer, especially if separate configurations are needed for each type of fault.

Action and Response Check:

- **SYSRS 021 Action/Response**: The system requirement demands that the retry configuration be adjustable for open and short faults.

- **SWRS 023 Action/Response**: The software requirement specifies the action taken when a retry is configured (re-enabling the affected channel for one timeslice). However, it doesn't clearly define how independent retry configurations for open and short faults are managed or applied.

  Issue: The software requirement does describe a retry mechanism but doesn't address how the system handles separate configurations for open and short faults.

Parameter Sources Check:

- **SYSRS 021 Parameters**: The system requirement implies a parameter for retry configurability (no explicit parameter is mentioned in the system requirement but inferred from the context).

- **SWRS 023 Parameters**: The "DEFAULT OPEN SHORT RETRY" is specified as a configuration item, which suggests a parameter source. However, it lacks clarity on whether this configuration applies to both open and short faults independently.

  Issue: The software requirement provides a parameter source ("DEFAULT OPEN SHORT RETRY") but does not clearly explain how it handles independent configuration for open and short faults. This needs to be clarified.

Additional Conditions Check:

- **SWRS 023 Additional Condition**: SWRS 023 introduces a retry interval (every 200 milliseconds), which is not specified in SYSRS 021.
- **Relevance**: The 200-millisecond retry interval seems to be an implementation detail that adds to the system requirement, rather than being unrelated. Therefore, this is acceptable and relevant.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirements (SWRS 023 and SWRS 024) generally support the system requirement by describing a retry mechanism for open and short faults. However, the system requirement's condition of **independent configurability** is not clearly addressed, weakening the meaningfulness of the link.

- **Completeness Rating**: **Medium**
  - **Reason**: The software requirements describe a retry mechanism and a configuration parameter ("DEFAULT OPEN SHORT RETRY"), but they do not fully explain how the independent configurability for open and short faults is achieved. The lack of clarity on independent configuration reduces completeness.

- **Overall Quality of the Link**: **Medium**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Medium, the overall quality of the link is also Medium. To improve the quality, the software requirements should clarify how retries for open and short faults can be configured **independently** and make sure that the retry parameter is detailed for both types of faults.

Suggested Revisions:

- **SWRS 023**: Include explicit details on how the retry configuration for open and short faults can be selected independently. Clarify whether the "DEFAULT OPEN SHORT RETRY" value applies to both or whether separate values are used.
- **SWRS 024**: If independent configuration is required, define separate configuration parameters for open and short faults or clearly explain how the single configuration parameter handles both faults independently.

"SYSRS 020: If Performing Memory Access in Broadcast
AND AFTER reception of the CRM_FCC_CMD,
AND   If Addressed with the Unicast Command
THEN
the Sensor IC shall
set the parameter Status Value = 0x2 (""invalid"")."   "SWRS 022: If Performing Memory Access in Broadcast
AND AFTER reception of the CRM_FCC_CMD,
AND If Addressed with the Broadcast Command
THEN

the Software shall
set the parameter Status Value = 0x2 (""invalid"")."

System Requirement (SYSRS 020):

- **Description**: If performing memory access in broadcast mode and after receiving the CRM_FCC_CMD, if addressed with a **unicast command**, the Sensor IC shall set the status value to 0x2 (indicating "invalid").

Software Requirement (SWRS 022):

- **Description**: If performing memory access in broadcast mode and after receiving the CRM_FCC_CMD, if addressed with a **broadcast command**, the software shall set the status value to 0x2 (indicating "invalid").

Condition Check:

- **SYSRS 020 Conditions**:
  - Performing memory access in broadcast mode.
  - After receiving the CRM_FCC_CMD.
  - If addressed with a **unicast command**.
  - Action: Set the status value to 0x2.

- **SWRS 022 Conditions**:
  - Performing memory access in broadcast mode.
  - After receiving the CRM_FCC_CMD.
  - **If addressed with a broadcast command**.
  - Action: Set the status value to 0x2.

  Issue: There is a mismatch in the conditions: SYSRS 020 specifies setting the status value in response to a **unicast command**, while SWRS 022 specifies it in response to a **broadcast command**. These are two different conditions, and the software requirement does not match the system requirement.

Action and Response Check:

- **SYSRS 020 Action/Response**: The system requirement specifies that the Sensor IC should set the status value to 0x2 when the conditions are met.

- **SWRS 022 Action/Response**: The software requirement also specifies setting the status value to 0x2 when the conditions are met.

  Issue: While both the system and software requirements specify the same action (setting the status to 0x2), the conditions under which this action should occur are not aligned (unicast vs. broadcast).

Parameter Sources Check:

- **SYSRS 020 Parameters**: The key parameter is the status value, which should be set to 0x2 (invalid).
- **SWRS 022 Parameters**: The software requirement also refers to the status value being set to 0x2.

  Issue: The status value parameter is addressed in both requirements. However, the parameter sources are dependent on the correct conditions (unicast vs. broadcast), which are mismatched.

Additional Conditions Check:

- **SWRS 022 Additional Condition**: SWRS 022 introduces the condition of being addressed by a **broadcast** command, which is not mentioned in SYSRS 020. This is an additional condition not relevant to the system requirement.

  Issue: The software requirement introduces an additional condition (broadcast command) that is not specified in the system requirement, and it changes the meaning of the action, making it a potential error or misalignment.

Ratings and Reasons:

- **Meaningfulness Rating**: **Low**
  - **Reason**: The software requirement introduces a different condition (broadcast command) than the system requirement (unicast command), which makes the link between the two weak and unclear. The software requirement does not directly support the system requirement, as the condition is entirely different.
- **Completeness Rating**: **Low**
  - **Reason**: The link is incomplete because the software requirement does not address the unicast command condition specified in the system requirement. Instead, it references a broadcast command, which leads to significant misalignment between the system and software requirements.
- **Overall Quality of the Link**: **Low**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Low, the overall quality of the link is also Low. The software requirement does not match the system requirement and introduces an unrelated condition, making the link fundamentally incorrect.

Suggested Revisions:

- **SWRS 022**: The software requirement should be revised to match the condition specified in the system requirement. It should reference the **unicast command** instead of the broadcast command. The correct condition should be:
  - "If Performing Memory Access in Broadcast AND AFTER reception of the CRM_FCC_CMD, AND If Addressed with the **Unicast Command** THEN the Software shall set the parameter

Status Value = 0x2 ('invalid')."

This revision would ensure alignment with the system requirement (SYSRS 020).

"SYSRS 019: If Performing Memory Access in Unicast
AND AFTER reception of the CRM_FCC_CMD,
AND If Addressed with the Broadcast Command
THEN
the Sensor IC shall
set the parameter Status Value = 0x2 (""invalid"")." "SWRS 021: If Performing Memory Access in Unicast
AND AFTER reception of the CRM_FCC_CMD,
AND If Addressed with the Broadcast Command
THEN
the Software shall
set the parameter Status Value = 0x2 (""invalid"")."

System Requirement (SYSRS 019):

- **Description**: If performing memory access in unicast mode and after receiving the CRM_FCC_CMD, if addressed with a **broadcast command**, the Sensor IC shall set the status value to 0x2 (indicating "invalid").

Software Requirement (SWRS 021):

- **Description**: If performing memory access in unicast mode and after receiving the CRM_FCC_CMD, if addressed with a **broadcast command**, the software shall set the status value to 0x2 (indicating "invalid").

Condition Check:

- **SYSRS 019 Conditions**:
  - Performing memory access in unicast mode.
  - After receiving the CRM_FCC_CMD.
  - If addressed with a **broadcast command**.
  - Action: Set the status value to 0x2.

- **SWRS 021 Conditions**:
  - Performing memory access in unicast mode.
  - After receiving the CRM_FCC_CMD.
  - If addressed with a **broadcast command**.
  - Action: Set the status value to 0x2.

Result: The conditions in both the system and software requirements are identical. Both specify that when memory access is in unicast mode and the CRM_FCC_CMD is received, and the device is addressed with a broadcast command, the status value should be set to 0x2.

Action and Response Check:

- **SYSRS 019 Action/Response**: The system requirement specifies that the Sensor IC should set the status value to 0x2 when the conditions are met.
- **SWRS 021 Action/Response**: The software requirement also specifies setting the status value to 0x2 when the conditions are met.

  Result: The action described in the software requirement matches the system requirement exactly. The response to the conditions is consistent across both requirements.

Parameter Sources Check:

- **SYSRS 019 Parameters**: The key parameter is the status value, which should be set to 0x2 (invalid).
- **SWRS 021 Parameters**: The software requirement also references the status value being set to 0x2.

  Result: Both the system and software requirements reference the same parameter (status value set to 0x2). There is no issue with parameter sources, as they are correctly addressed.

Additional Conditions Check:

- **SWRS 021 Additional Condition**: No additional conditions are introduced in the software requirement. It matches the system requirement exactly.

  Result: There are no extraneous conditions in the software requirement, ensuring that the link remains clear and relevant.

Ratings and Reasons:

- **Meaningfulness Rating**: **High**
  - **Reason**: The software requirement directly supports the system requirement. The conditions and actions described in both the system and software requirements are identical, making the link strong and logical.
- **Completeness Rating**: **High**
  - **Reason**: The software requirement fully covers the system requirement, including all the necessary conditions, actions, and parameters. There is no missing or unclear information.
- **Overall Quality of the Link**: **High**
  - **Reason**: Since both Meaningfulness and Completeness are rated as High, the overall quality of the link is also High. The software requirement correctly and thoroughly implements the

system requirement.

Suggested Revisions:

- **None**: The software requirement (SWRS 021) perfectly aligns with the system requirement (SYSRS 019), so no revisions are needed. The link is both meaningful and complete, addressing all necessary aspects for successful implementation.

"SYSRS 018: - Unlocked, If No Key Is Set
If Performing Memory Access
AND If the Key is NOT set
THEN
the Sensor IC shall
set the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea to the state Memory Area State: Unlocked." "SWRS 020: - locked, If Key Is Set
If Performing Memory Access
AND If the Key is set
THEN
the Software shall
set the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea to the state Memory Area State: locked."

System Requirement (SYSRS 018):

- **Description**: If performing memory access and the key is **not set**, the Sensor IC shall set the state of the memory area (identified by CRM_CMD 0x8) to **unlocked**.

Software Requirement (SWRS 020):

- **Description**: If performing memory access and the key **is set**, the software shall set the state of the memory area (identified by CRM_CMD 0x8) to **locked**.

Condition Check:

- **SYSRS 018 Conditions**:
  - Performing memory access.
  - If the key is **not set**.
  - Action: Set the memory area state to **unlocked**.
- **SWRS 020 Conditions**:
  - Performing memory access.
  - If the key **is set**.
  - Action: Set the memory area state to **locked**.

Issue: The conditions in SYSRS 018 and SWRS 020 are opposite. SYSRS 018 focuses on the scenario when the key is **not set**, resulting in an **unlocked** memory area, while SWRS 020 focuses on the scenario when the key **is set**, resulting in a **locked** memory area. SWRS 020 does not directly implement or support the condition of SYSRS 018 (unlocked when the key is not set).

Action and Response Check:

- **SYSRS 018 Action/Response**: The system requirement specifies that when the conditions are met (key is **not set**), the memory area state should be set to **unlocked**.
- **SWRS 020 Action/Response**: The software requirement specifies that when the key **is set**, the memory area state should be set to **locked**.

  Issue: The actions are different. SYSRS 018 expects the memory area to be **unlocked** when the key is not set, but SWRS 020 describes an action that applies when the key is set (locking the memory area). The software requirement does not fulfill the action required by SYSRS 018, which is unlocking the memory area when the key is not set.

Parameter Sources Check:

- **SYSRS 018 Parameters**: The key parameter here is the **MemoryArea** identified by CRM_CMD 0x8, with the state being set to **unlocked** if the key is not set.
- **SWRS 020 Parameters**: The software requirement also refers to the **MemoryArea** identified by CRM_CMD 0x8, but the state is set to **locked** if the key is set.

  Result: Both requirements refer to the same **MemoryArea** parameter, but the resulting states are different. While the parameter itself is correctly addressed, the state (unlocked vs. locked) depends on the conditions (key set vs. not set), and they are not aligned.

Additional Conditions Check:

- **SWRS 020 Additional Condition**: SWRS 020 introduces the condition of the key **being set** rather than **not being set**, which is not addressed in SYSRS 018. This is an additional condition that does not directly align with the system requirement's focus on the "key not set" condition.

  Issue: The software requirement introduces a condition (key is set) that is not mentioned in SYSRS 018, and this causes misalignment between the two requirements. The system requirement expects behavior when the key is **not set**, but the software requirement handles the opposite scenario.

Ratings and Reasons:

- **Meaningfulness Rating**: **Low**
  - **Reason**: The software requirement focuses on locking the memory area when the key is set, which is the opposite condition of the system requirement, where the memory area should be unlocked when the key is not set. This makes the link between them weak and irrelevant for SYSRS 018.

- **Completeness Rating**: **Low**
  - **Reason**: The software requirement does not cover the conditions or actions outlined in the system requirement. It does not address what should happen when the key is not set, which is critical to fulfilling the system requirement. This results in an incomplete link.

- **Overall Quality of the Link**: **Low**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Low, the overall quality of the link is also Low. The software requirement handles a completely different scenario (locking instead of unlocking), making it unsuitable as a match for the system requirement.

Suggested Revisions:

- **SWRS 020** should be revised to handle both scenarios:
  - The software requirement needs to explicitly cover the case where the key is **not set**, and the memory area should be **unlocked**. This can either be added as a separate software requirement or integrated into SWRS 020 by handling both the "key set" and "key not set" conditions.

For example:

- "If Performing Memory Access AND If the Key is **not set**, THEN the Software shall set the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea to the state Memory Area State: **unlocked**."

This would ensure that both locking and unlocking behavior is covered, fulfilling both conditions (key set and key not set).

"SYSRS 017: If Performing the Standard Path Measurement
AND IF performing Stop Ongoing Measurement (Blocked)
AND IF the functionality in US-Transducer Control (Non-Modulated Sine Burst) has NOT been fully performed,
THEN
the Sensor IC shall
set Error: err_burstGenerationStopped = 0x1 (""error"")." "SWRS 019: IF in Main-Measurement
AND IF Break Burst Generation occurs
THEN
the software shall set
Error: err_burstGenerationStopped = 0x1 (""error"")."

System Requirement (SYSRS 017):

- **Description**: If performing the standard path measurement and stop ongoing measurement (blocked) is triggered, and if the non-modulated sine burst generation in the US-Transducer

Control has not been fully performed, the Sensor IC shall set the error `err_burstGenerationStopped = 0x1` (indicating an error).

Software Requirement (SWRS 019):

- **Description**: If the system is in the main measurement phase and burst generation is interrupted, the software shall set the error `err_burstGenerationStopped = 0x1` (indicating an error).

Condition Check:

- **SYSRS 017 Conditions**:
  - Performing standard path measurement.
  - Stop ongoing measurement (blocked) is triggered.
  - The non-modulated sine burst in the US-Transducer Control has **not been fully performed**.
  - Action: Set the error `err_burstGenerationStopped = 0x1`.

- **SWRS 019 Conditions**:
  - In the main measurement phase.
  - Burst generation is interrupted.
  - Action: Set the error `err_burstGenerationStopped = 0x1`.

  Issue: The software requirement (SWRS 019) mentions burst generation interruption, but it does not specifically cover all the conditions in the system requirement. In SYSRS 017, the error is triggered if the **stop ongoing measurement (blocked)** command is issued and the sine burst generation is **not fully performed**. SWRS 019, however, mentions "break burst generation" without referencing the conditions about stopping ongoing measurement or whether the sine burst was fully performed. This makes the conditions in SWRS 019 broader and less specific than those in SYSRS 017.

Action and Response Check:

- **SYSRS 017 Action/Response**: The system requirement specifies that the Sensor IC should set the error `err_burstGenerationStopped = 0x1` when the defined conditions are met.

- **SWRS 019 Action/Response**: The software requirement also specifies setting the error `err_burstGenerationStopped = 0x1` when burst generation is interrupted.

  Issue: While both requirements specify setting the same error, the software requirement does not provide sufficient detail about when the burst generation is interrupted, specifically missing the check about whether the sine burst was **not fully performed** and whether the "stop ongoing measurement" condition applies.

Parameter Sources Check:

- **SYSRS 017 Parameters**: The key parameter here is the error `err_burstGenerationStopped`, which should be set to 0x1 if the conditions are met.

- **SWRS 019 Parameters**: The software requirement also references the same error (`err_burstGenerationStopped = 0x1`).

  Result: Both requirements reference the same error parameter, so there is no issue regarding parameter sources. However, the conditions triggering the error differ between the system and software requirements, impacting the appropriateness of setting the parameter.

Additional Conditions Check:

- **SWRS 019 Additional Condition**: SWRS 019 introduces the condition of "burst generation interruption," which is similar but broader than the specific case described in SYSRS 017. However, SWRS 019 lacks important conditions such as stopping the ongoing measurement and whether the sine burst was fully performed.

  Issue: SWRS 019 does not fully match the conditions described in SYSRS 017. It introduces a more general "burst generation interruption" condition, while SYSRS 017 requires a more specific context (stopping ongoing measurement and incomplete sine burst performance).

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement generally supports the system requirement by referencing the same error (`err_burstGenerationStopped = 0x1`), but it lacks key conditions that are specifically called out in SYSRS 017. The lack of reference to "stop ongoing measurement" and the "non-modulated sine burst not fully performed" conditions weakens the meaningfulness of the link.

- **Completeness Rating**: **Medium**
  - **Reason**: The software requirement is somewhat complete but omits important conditions described in the system requirement, such as checking whether the sine burst was fully performed and whether the "stop ongoing measurement" condition is met. Without these details, the link is incomplete.

- **Overall Quality of the Link**: **Medium**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Medium, the overall quality of the link is also Medium. To improve this, the software requirement should include more specific conditions to match the system requirement fully.

Suggested Revisions:

- **SWRS 019** should be revised to include the specific conditions from SYSRS 017. It should explicitly address the following:
  - "If performing Stop Ongoing Measurement (Blocked)"
  - "If the non-modulated sine burst in US-Transducer Control has **not been fully performed**"

A revised version could be:

- "If in Main-Measurement AND If performing Stop Ongoing Measurement (Blocked) AND If the non-modulated sine burst in US-Transducer Control has NOT been fully performed, THEN the software shall set Error: err_burstGenerationStopped = 0x1 ('error')."

This would bring the software requirement in line with the system requirement and improve the overall quality of the link.

"SYSRS 016: After Reception of the Valid CRM_FCC_CMD,
THEN
the Sensor IC shall
set the PDCM_RSP Parameter KAC = 0x0." "SWRS 018: Increment KAC
AFTER Sending PDCM Packet After Reception of the BRC
THEN
the Software shall
increment the PDCM_RSP Parameter KAC by 1;

After Reception of the Valid CRM_FCC_CMD,
THEN
the Software shall
set the PDCM_RSP Parameter KAC = 0x3."

System Requirement (SYSRS 016):

- **Description**: After the reception of a valid CRM_FCC_CMD, the Sensor IC shall set the `PDCM_RSP` parameter `KAC = 0x0`.

Software Requirement (SWRS 018):

- **Description**:
  1. After sending the PDCM packet following the reception of the BRC, the software shall increment the `PDCM_RSP` parameter `KAC` by 1.
  2. After the reception of a valid CRM_FCC_CMD, the software shall set the `PDCM_RSP` parameter `KAC = 0x3`.

Condition Check:

- **SYSRS 016 Conditions**:
  - After reception of a valid **CRM_FCC_CMD**.
  - Action: Set `KAC = 0x0`.

- **SWRS 018 Conditions**:
  1. After sending the PDCM packet after the reception of the **BRC** command.
  2. After reception of a valid **CRM_FCC_CMD**, the software sets `KAC = 0x3`.

Issue: There is a mismatch between the conditions in SYSRS 016 and SWRS 018. SYSRS 016 specifies that after receiving a valid CRM_FCC_CMD, the `KAC` parameter should be set to `0x0`, while SWRS 018 specifies that after the same event, the `KAC` parameter is set to `0x3`. Additionally, SWRS 018 introduces an unrelated condition (incrementing `KAC` after receiving the BRC and sending the PDCM packet), which is not referenced in SYSRS 016.

Action and Response Check:

- **SYSRS 016 Action/Response**: After receiving a valid CRM_FCC_CMD, the system should set `KAC = 0x0`.

- **SWRS 018 Action/Response**: After receiving a valid CRM_FCC_CMD, the software sets `KAC = 0x3`, and in a separate case (after sending the PDCM packet following a BRC), it increments `KAC` by 1.

  Issue: The actions between the system and software requirements do not match. SYSRS 016 requires `KAC` to be set to `0x0`, whereas SWRS 018 specifies that `KAC` should be set to `0x3` in response to the same event. This is a major misalignment, as the required action differs entirely between the two requirements.

Parameter Sources Check:

- **SYSRS 016 Parameters**: The `PDCM_RSP` parameter `KAC` is central, and it should be set to `0x0` after receiving the CRM_FCC_CMD.

- **SWRS 018 Parameters**: The same `PDCM_RSP` parameter `KAC` is referenced but is set to `0x3` after the CRM_FCC_CMD. Additionally, it is incremented by 1 after the BRC event and sending the PDCM packet.

  Issue: The `KAC` parameter is correctly referenced in both requirements, but the value it should take (`0x0` vs. `0x3`) differs between SYSRS 016 and SWRS 018. This discrepancy means that the software requirement does not correctly implement the system requirement.

Additional Conditions Check:

- **SWRS 018 Additional Condition**: SWRS 018 introduces an additional condition, incrementing `KAC` after the PDCM packet is sent following a BRC command, which is not mentioned in SYSRS 016.

  Issue: This additional condition (incrementing after the BRC event) is unrelated to SYSRS 016 and seems unnecessary in this context. Since SYSRS 016 does not mention this event, it adds complexity without directly supporting the system requirement.

Ratings and Reasons:

- **Meaningfulness Rating**: **Low**
  - **Reason**: The software requirement does not align with the system requirement. After the reception of the CRM_FCC_CMD, SYSRS 016 requires `KAC` to be set to `0x0`, while SWRS

018 sets it to `0x3`. This fundamental mismatch makes the link weak and unclear. Furthermore, the additional BRC condition is unrelated to SYSRS 016.

- **Completeness Rating**: **Low**
  - **Reason**: The software requirement does not complete the system requirement, as it specifies a different value for the `KAC` parameter than what is required. Additionally, the extra condition (incrementing `KAC` after BRC) adds unnecessary complexity and does not address the system requirement.

- **Overall Quality of the Link**: **Low**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Low, the overall quality of the link is also Low. The software requirement does not correctly implement the system requirement and introduces unrelated actions and conditions.

Suggested Revisions:

- **SWRS 018** should be revised to match the system requirement:
  - After the reception of a valid CRM_FCC_CMD, the software should set `KAC = 0x0` as required by SYSRS 016. The value `0x3` should be replaced with `0x0` to align with the system requirement.
  - Remove the additional condition of incrementing `KAC` after sending the PDCM packet following the BRC command, as this is unrelated to the system requirement and could introduce unnecessary confusion.

The revised SWRS 018 would then be:

- "After Reception of the Valid CRM_FCC_CMD, THEN the Software shall set the PDCM_RSP Parameter KAC = 0x0."

This would ensure the software correctly implements the behavior required by the system.

"SYSRS 015: Increment KAC
AFTER Sending PDCM Packet After Reception of the BRC
THEN
the Sensor IC shall
increment the PDCM_RSP Parameter KAC by 1; the counter is reset to 0 at overflow" "SWRS 017: Increment KAC
AFTER Sending PDCM Packet After Reception of the BRC
THEN
the Software shall
increment the PDCM_RSP Parameter KAC by 1; "

System Requirement (SYSRS 015):

- **Description**: After sending the PDCM packet following the reception of the BRC, the Sensor IC shall increment the `PDCM_RSP` parameter `KAC` by 1. Additionally, the counter (`KAC`) should reset to 0 when it overflows.

Software Requirement (SWRS 017):

- **Description**: After sending the PDCM packet following the reception of the BRC, the software shall increment the `PDCM_RSP` parameter `KAC` by 1.

Condition Check:

- **SYSRS 015 Conditions**:
  - After sending the PDCM packet.
  - After reception of the BRC.
  - Action: Increment `KAC` by 1 and **reset the counter to 0 at overflow**.

- **SWRS 017 Conditions**:
  - After sending the PDCM packet.
  - After reception of the BRC.
  - Action: Increment `KAC` by 1.

  Issue: While both SYSRS 015 and SWRS 017 share the same conditions about sending the PDCM packet after receiving the BRC, **SWRS 017 lacks the detail about resetting the `KAC` counter at overflow**. This omission means that the software requirement is missing an important aspect of the condition specified in the system requirement.

Action and Response Check:

- **SYSRS 015 Action/Response**: The system requirement specifies that the `KAC` parameter should be incremented by 1, and it must reset to 0 when it overflows.

- **SWRS 017 Action/Response**: The software requirement also specifies incrementing the `KAC` parameter by 1 but does not mention the reset to 0 at overflow.

  Issue: While both requirements describe the increment action, **SWRS 017 omits the overflow reset behavior**, which is a crucial aspect of the system requirement's expected behavior.

Parameter Sources Check:

- **SYSRS 015 Parameters**: The `PDCM_RSP` parameter `KAC` is incremented by 1, and the counter is reset to 0 at overflow.

- **SWRS 017 Parameters**: The same parameter (`KAC`) is mentioned for the increment, but the overflow reset is not handled.

  Issue: The parameter (`KAC`) is correctly referenced in both requirements, but the missing reset at overflow in SWRS 017 means the parameter handling is incomplete in the software requirement.

Additional Conditions Check:

- **SWRS 017 Additional Condition**: SWRS 017 does not introduce any additional conditions, but it omits the overflow reset condition from SYSRS 015.

  Result: No irrelevant additional conditions are introduced, but the omission of the overflow behavior in SWRS 017 makes it incomplete in comparison to SYSRS 015.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement generally supports the system requirement by implementing the increment of the `KAC` parameter after the appropriate conditions (sending PDCM packet after receiving BRC). However, the omission of the overflow reset makes the software requirement less meaningful in fully supporting the intended behavior of the system.

- **Completeness Rating**: **Medium**
  - **Reason**: The software requirement covers the main increment action but does not fully address the system requirement because it omits the reset to 0 at overflow. This makes the link somewhat incomplete, as the full behavior expected by the system requirement is not fully captured.

- **Overall Quality of the Link**: **Medium**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Medium, the overall quality of the link is also Medium. The software requirement does address a significant part of the system requirement (incrementing `KAC`), but it lacks the overflow reset behavior, which reduces the quality of the link.

Suggested Revisions:

- **SWRS 017** should be revised to include the overflow reset behavior specified in SYSRS 015. The software requirement should ensure that after incrementing `KAC`, the counter is reset to 0 upon overflow.

A revised version of SWRS 017 could be:

- "After Sending PDCM Packet After Reception of the BRC, THEN the Software shall increment the PDCM_RSP Parameter KAC by 1; the counter is reset to 0 at overflow."

This revision would fully align the software requirement with the system requirement and ensure complete and correct implementation of the desired behavior.

"SYSRS 014: IF the Event_MAX is generated,
THEN
the Sensor IC shall

measure the time of the occurrence of the event condition relative to the start of the measurement and provide PDCM_RSP P12, P13 Parameter EventTimeStamp<i> for the <i>th Event transmitted in a DSI3 packet." SWRS 016: The software shall send the MAX events with the measured time of the occurrence of the event, condition relative to the start of the measurement, and provide the PDCM_RSP P12,P13 Parameter EventTimeStamp<i> for the <i>th Event transmitted in a PDCM frame.

System Requirement (SYSRS 014):

- **Description**: If the **Event_MAX** is generated, the Sensor IC shall measure the time of the event's occurrence relative to the start of the measurement and provide the `PDCM_RSP` parameters `P12`, `P13` as `EventTimeStamp<i>` for the `<i>th` event transmitted in a DSI3 packet.

Software Requirement (SWRS 016):

- **Description**: The software shall send the **MAX events** with the measured time of the event's occurrence relative to the start of the measurement and provide the `PDCM_RSP` parameters `P12`, `P13` as `EventTimeStamp<i>` for the `<i>th` event transmitted in a PDCM frame.

Condition Check:

- **SYSRS 014 Conditions**:
  - If the **Event_MAX** is generated.
  - Action: Measure the time of the event's occurrence relative to the start of the measurement and provide `EventTimeStamp<i>` for the `<i>th` event in a DSI3 packet.

- **SWRS 016 Conditions**:
  - If the **MAX event** occurs.
  - Action: Send the measured time of the event's occurrence relative to the start of the measurement and provide `EventTimeStamp<i>` for the `<i>th` event in a PDCM frame.

  Issue: There is a slight difference in how the event data is transmitted. SYSRS 014 specifies that the event timestamps are transmitted in a **DSI3 packet**, while SWRS 016 specifies that the event timestamps are transmitted in a **PDCM frame**. This is a key difference in the transport mechanism used, which needs clarification. If the intention was to have the same transport mechanism, SWRS 016 needs to align with the system requirement.

Action and Response Check:

- **SYSRS 014 Action/Response**: The system requirement specifies that when an Event_MAX is generated, the Sensor IC should measure the event's occurrence time and transmit it as `EventTimeStamp<i>` in a DSI3 packet.

- **SWRS 016 Action/Response**: The software requirement specifies that the MAX events should be sent with the measured time as `EventTimeStamp<i>` in a PDCM frame.

Issue: The action of measuring and sending the event timestamp is consistent between the system and software requirements. However, the system requirement explicitly mentions using a **DSI3 packet**, while the software requirement mentions a **PDCM frame**. This discrepancy may indicate different communication mechanisms, which needs clarification.

Parameter Sources Check:

- **SYSRS 014 Parameters**: The system requirement specifies using `P12` and `P13` as the parameters for `EventTimeStamp<i>`, corresponding to the `<i>th` event in a DSI3 packet.
- **SWRS 016 Parameters**: The software requirement also uses `P12` and `P13` for the same purpose (`EventTimeStamp<i>`) for the `<i>th` event but in a PDCM frame.

  Result: The parameters (`P12` and `P13`) are consistent in both the system and software requirements, meaning that the parameter usage is correct. The main issue lies in the type of packet or frame used to transmit these parameters.

Additional Conditions Check:

- **SWRS 016 Additional Condition**: No additional conditions are introduced. However, the transmission mechanism (PDCM frame instead of DSI3 packet) seems to be an alternate or conflicting condition rather than an additional one.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement generally supports the system requirement by implementing the measurement and transmission of the event timestamp. However, the difference in the transmission method (PDCM frame vs. DSI3 packet) weakens the meaningfulness of the link, as the system and software requirements seem to expect different communication mechanisms.
- **Completeness Rating**: **Medium**
  - **Reason**: The software requirement covers most aspects of the system requirement, including the measurement and parameter usage. However, the lack of clarity on the transport mechanism (DSI3 packet vs. PDCM frame) means that the link is somewhat incomplete, as this difference is not reconciled.
- **Overall Quality of the Link**: **Medium**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Medium, the overall quality of the link is also Medium. The software requirement does implement the core functionality described in the system requirement, but the discrepancy in the communication protocol (DSI3 vs. PDCM) needs clarification to ensure full alignment.

Suggested Revisions:

- **SWRS 016** should be revised to clarify or align the transmission mechanism with the system requirement. If the system requirement intends for the event timestamp to be transmitted via a

DSI3 packet, the software requirement should be updated to reflect this.

A revised version of SWRS 016 could be:

- "The software shall send the MAX events with the measured time of the occurrence of the event, condition relative to the start of the measurement, and provide the PDCM_RSP P12, P13 Parameter EventTimeStamp<i> for the <i>th Event transmitted in a **DSI3 packet**."

This revision would ensure consistency between the system and software requirements and remove any ambiguity regarding the transmission method. If using a **PDCM frame** is intentional, the system requirement should be updated accordingly.

SYSRS 013: The Sensor IC shall amplify the Transducer Input Signal at the IC Pins with the three amplification factors (Parameter CALIB_analogGain plus Parameter SPX_dynGainStartValue plus CRM_CMD 0x4 Parameter CalibrationGain plus Parameter YPX_calibrationGainOffset). "SWRS 013: Analog Amplification: AFTER reception of the command CRM_CMD 0x4: Start Calibration Measurement the software shall set the register the register ASP_CFG.g_ana to Parameter CALIB_analogGain .

SWRS 014: Digital Amplification: AFTER reception of the command CRM_CMD 0x4: Start Calibration Measurement the software shall set the register ENVP_AMPD_CFG.g_dig to Parameter SPX_dynGainStartValue .

SWRS 015: Calibration Gain: AFTER reception of the command CRM_CMD 0x4: Start Calibration Measurement
the software shall set the register ENVP_GCAL.g_cal to (CRM_CMD 0x4 Parameter CalibrationGain + Parameter YPX_calibrationGainOffset).
"

System Requirement (SYSRS 013):

- **Description**: The Sensor IC shall amplify the transducer input signal at the IC pins using the following combination of amplification factors:
  - **CALIB_analogGain**
  - **SPX_dynGainStartValue**
  - **CRM_CMD 0x4 Parameter CalibrationGain**
  - **YPX_calibrationGainOffset**

Software Requirements (SWRS 013, SWRS 014, SWRS 015):

1. **SWRS 013**: After reception of the command CRM_CMD 0x4 (Start Calibration Measurement), the software shall set the register `**ASP_CFG.g_ana**` to **Parameter CALIB_analogGain**.
2. **SWRS 014**: After reception of the command CRM_CMD 0x4 (Start Calibration Measurement), the software shall set the register `**ENVP_AMPD_CFG.g_dig**` to **Parameter SPX_dynGainStartValue**.
3. **SWRS 015**: After reception of the command CRM_CMD 0x4 (Start Calibration Measurement), the software shall set the register `**ENVP_GCAL.g_cal**` to **(CRM_CMD 0x4 Parameter CalibrationGain**

+ YPX_calibrationGainOffset).

Condition Check:

- **SYSRS 013 Conditions**:
  - Amplification is to be performed using a combination of **four parameters**: `CALIB_analogGain`, `SPX_dynGainStartValue`, `CRM_CMD 0x4 Parameter CalibrationGain`, and `YPX_calibrationGainOffset`.
- **SWRS 013 Conditions**:
  - After receiving the CRM_CMD 0x4 (Start Calibration Measurement), set the analog gain register to `CALIB_analogGain`.
- **SWRS 014 Conditions**:
  - After receiving CRM_CMD 0x4 (Start Calibration Measurement), set the digital gain register to `SPX_dynGainStartValue`.
- **SWRS 015 Conditions**:
  - After receiving CRM_CMD 0x4 (Start Calibration Measurement), set the calibration gain register to the combination of `CalibrationGain` and `YPX_calibrationGainOffset`.

  Result: The condition in SYSRS 013 (amplification using four parameters) is matched in the software requirements (SWRS 013, 014, 015), which handle each parameter separately after the reception of CRM_CMD 0x4. All the necessary amplification factors are addressed in the software requirements.

Action and Response Check:

- **SYSRS 013 Action/Response**: The system requirement specifies that the Sensor IC shall amplify the input signal using the four specified parameters.
- **SWRS 013, 014, 015 Action/Response**: The software requirements detail the configuration of the amplification parameters in separate registers:
  - **SWRS 013**: Sets `ASP_CFG.g_ana` to `CALIB_analogGain`.
  - **SWRS 014**: Sets `ENVP_AMPD_CFG.g_dig` to `SPX_dynGainStartValue`.
  - **SWRS 015**: Sets `ENVP_GCAL.g_cal` to the sum of `CalibrationGain` and `YPX_calibrationGainOffset`.

  Result: The software requirements correctly implement the actions described in the system requirement, with each parameter being applied appropriately to specific registers. However, it is important to note that **SYSRS 013 refers to all four parameters collectively** for amplification, while the software splits this into separate register configurations.

Parameter Sources Check:

- **SYSRS 013 Parameters**: Four amplification factors are involved: `CALIB_analogGain`, `SPX_dynGainStartValue`, `CRM_CMD 0x4 CalibrationGain`, and `YPX_calibrationGainOffset`.

- **SWRS 013, 014, 015 Parameters**:
  - SWRS 013: `CALIB_analogGain`.
  - SWRS 014: `SPX_dynGainStartValue`.
  - SWRS 015: `CalibrationGain + YPX_calibrationGainOffset`.

  Result: All the necessary parameters from the system requirement are addressed in the software requirements, with the correct mapping of parameters to their respective registers.

Additional Conditions Check:

- **SWRS 013, 014, 015 Additional Conditions**: There are no additional conditions introduced in the software requirements. The reception of CRM_CMD 0x4 (Start Calibration Measurement) triggers all the relevant actions in each of the software requirements, which aligns with the system requirement.

  Result: No additional or unrelated conditions are introduced, so the software requirements remain relevant to the system requirement.

Ratings and Reasons:

- **Meaningfulness Rating**: **High**
  - **Reason**: The software requirements clearly and directly implement the system requirement. Each amplification factor is handled in a separate software requirement, and all factors specified in SYSRS 013 are accounted for. While the system requirement describes them collectively, the software requirements break them down into individual actions, which makes the link meaningful and relevant.

- **Completeness Rating**: **High**
  - **Reason**: The software requirements fully cover the system requirement by configuring the registers for each of the specified amplification parameters. The condition of applying all four parameters after receiving CRM_CMD 0x4 is fully addressed in SWRS 013, 014, and 015.

- **Overall Quality of the Link**: **High**
  - **Reason**: Both the Meaningfulness and Completeness are rated as High, so the overall quality of the link is also High. The software requirements effectively implement the system requirement and ensure that all necessary parameters are applied to the amplification process.

Suggested Revisions:

- **None**: The software requirements (SWRS 013, 014, 015) align well with the system requirement (SYSRS 013) and address all necessary aspects. No revisions are needed. The software requirements provide the required detail for each amplification factor and ensure that the system behavior is fully implemented.

"SYSRS 014: IF the Event_MAX is generated,
THEN
the Sensor IC shall
measure the time of the occurrence of the event condition relative to the start of the measurement and provide PDCM_RSP P12, P13 Parameter EventTimeStamp<i> for the <i>th Event transmitted in a DSI3 packet." SWRS 016: The software shall send the MAX events with the measured time of the occurrence of the event, condition relative to the start of the measurement, and provide the PDCM_RSP P12,P13 Parameter EventTimeStamp<i> for the <i>th Event transmitted in a PDCM frame.

System Requirement (SYSRS 014):

- **Description**: If the **Event_MAX** is generated, the Sensor IC shall measure the time of the event's occurrence relative to the start of the measurement and provide the `PDCM_RSP` parameters `P12` and `P13` as `EventTimeStamp<i>` for the `<i>th` event transmitted in a **DSI3 packet**.

Software Requirement (SWRS 016):

- **Description**: The software shall send the MAX events with the measured time of the event's occurrence relative to the start of the measurement and provide the `PDCM_RSP` parameters `P12` and `P13` as `EventTimeStamp<i>` for the `<i>th` event transmitted in a **PDCM frame**.

Condition Check:

- **SYSRS 014 Conditions**:
  - When the **Event_MAX** is generated.
  - The time of the event's occurrence is measured relative to the start of the measurement.
  - The timestamp (`EventTimeStamp<i>`) is transmitted in a **DSI3 packet**.

- **SWRS 016 Conditions**:
  - When the **MAX event** occurs.
  - The time of the event's occurrence is measured relative to the start of the measurement.
  - The timestamp (`EventTimeStamp<i>`) is transmitted in a **PDCM frame**.

  Issue: The major difference lies in the transmission medium. SYSRS 014 specifies that the timestamp is sent in a **DSI3 packet**, while SWRS 016 specifies a **PDCM frame**. These are two different types of communication packets, which creates a discrepancy between the system and software requirements. If the intention is to use the same communication method, the software requirement must be aligned with the system requirement.

Action and Response Check:

- **SYSRS 014 Action/Response**: The Sensor IC shall measure the event's timestamp and provide the `EventTimeStamp<i>` in a **DSI3 packet**.

- **SWRS 016 Action/Response**: The software shall send the `EventTimeStamp<i>` for MAX events in a **PDCM frame**.

  Issue: The action described in SWRS 016 is similar to SYSRS 014 in terms of measuring the timestamp, but it refers to sending it in a **PDCM frame** rather than a **DSI3 packet**. This creates a potential conflict in implementation if different transmission mechanisms are expected.

Parameter Sources Check:

- **SYSRS 014 Parameters**: The `EventTimeStamp<i>` is constructed using the `PDCM_RSP` parameters `P12` and `P13` and applies to the `<i>th` event.

- **SWRS 016 Parameters**: The software requirement correctly references the `PDCM_RSP` parameters `P12` and `P13` for the same purpose of providing the `EventTimeStamp<i>`.

  Result: The parameters (`P12` and `P13`) are consistently referenced in both requirements. However, the differing packet transmission methods (DSI3 vs. PDCM) affect how these parameters are transmitted.

Additional Conditions Check:

- **SWRS 016 Additional Condition**: No additional conditions are introduced in the software requirement, apart from the difference in transmission (PDCM frame vs. DSI3 packet).

  Result: No extraneous conditions are introduced, but the change in the transmission method is a notable difference between the system and software requirements.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement generally supports the system requirement by implementing the measurement and transmission of the event timestamp. However, the discrepancy in the packet type used for transmission (PDCM frame vs. DSI3 packet) weakens the meaningfulness of the link, as it suggests a different communication method than expected in the system requirement.

- **Completeness Rating**: **Medium**
  - **Reason**: The software requirement covers most aspects of the system requirement, including the measurement of the event timestamp and correct usage of the `P12` and `P13` parameters. However, the mismatch in the transmission medium (PDCM frame vs. DSI3 packet) means that the link is not fully complete, as the expected communication method is unclear.

- **Overall Quality of the Link**: **Medium**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Medium, the overall quality of the link is also Medium. The software requirement implements the core functionality but deviates from the system requirement in terms of the transmission method, which reduces the overall quality.

Suggested Revisions:

- **SWRS 016** should be revised to clarify or align the transmission mechanism with the system requirement. If the system requires the event timestamp to be transmitted via a **DSI3 packet**, the software requirement should be updated accordingly.

A revised version of SWRS 016 could be:

- "The software shall send the MAX events with the measured time of the occurrence of the event, condition relative to the start of the measurement, and provide the PDCM_RSP P12, P13 Parameter EventTimeStamp<i> for the <i>th Event transmitted in a **DSI3 packet**."

This would ensure that the communication method is consistent between the system and software requirements. If a **PDCM frame** is intended, the system requirement should be updated to reflect that.

"SYSRS 012: BEFORE Leaving the Discovery Mode
AND IF the effective value of the parameter DSI3_physicalAddress is NOT equal to the default value of the Parameter DSI3_physicalAddress
THEN
the Sensor IC shall
set Error: err_discPhysAddrMismatch = 0x1 (""error detected"").
" "SWRS 012 : BEFORE Leaving the Discovery Mode
AND IF the effective value of the Parameter DSI3_physicalAddress (RAM) is NOT equal to the default value of the Parameter DSI3_physicalAddress (NVM)
THEN
the Software shall
set Error: err_discPhysAddrMismatch = 0x1 (""error detected"").
OTHERWISE
the Software shall
set Error: err_discPhysAddrMismatch = 0x0 (""no error"").
"

System Requirement (SYSRS 012):

- **Description**: Before leaving Discovery Mode, if the effective value of the parameter **DSI3_physicalAddress** is not equal to its default value, the Sensor IC shall set `**Error: err_discPhysAddrMismatch = 0x1**` (indicating an error).

Software Requirement (SWRS 012):

- **Description**:
  - Before leaving Discovery Mode, if the effective value of the **DSI3_physicalAddress** in **RAM** is not equal to its default value in **NVM**, the software shall set `**Error: err_discPhysAddrMismatch = 0x1**` (indicating an error).

- Otherwise, the software shall set `Error: err_discPhysAddrMismatch = 0x0` (indicating no error).

Condition Check:

- **SYSRS 012 Conditions**:
  - Before leaving Discovery Mode.
  - If the effective value of **DSI3_physicalAddress** is not equal to its default value.
  - Action: Set `err_discPhysAddrMismatch = 0x1`.

- **SWRS 012 Conditions**:
  - Before leaving Discovery Mode.
  - If the effective value of **DSI3_physicalAddress (RAM)** is not equal to its default value in **NVM**.
  - Action: Set `err_discPhysAddrMismatch = 0x1`. If the addresses match, set `err_discPhysAddrMismatch = 0x0`.

  Result: SWRS 012 adds more specificity by checking the difference between the value in **RAM** and the default value in **NVM**, whereas SYSRS 012 simply checks if the current value differs from the default. Additionally, SWRS 012 introduces an **else condition**, which sets `err_discPhysAddrMismatch = 0x0` if no mismatch is detected, which is not mentioned in SYSRS 012.

Action and Response Check:

- **SYSRS 012 Action/Response**: The system requirement specifies setting the error `err_discPhysAddrMismatch = 0x1` if a mismatch in the physical address is detected before leaving Discovery Mode.

- **SWRS 012 Action/Response**: The software requirement specifies the same action if a mismatch is detected but also includes an action for when no mismatch is found, setting the error to `0x0`.

  Result: The error setting in the event of a mismatch aligns between both requirements. However, SWRS 012 adds an action for setting the error to `0x0` when there is no mismatch, which is not mentioned in SYSRS 012.

Parameter Sources Check:

- **SYSRS 012 Parameters**: The parameter in question is **DSI3_physicalAddress**, with a comparison against its default value.

- **SWRS 012 Parameters**: The software requirement specifies comparing **DSI3_physicalAddress** in **RAM** against its default value in **NVM**.

  Result: SWRS 012 adds more detail by distinguishing between the **RAM** and **NVM** values of the **DSI3_physicalAddress**, which is a valid clarification of how the comparison should be performed. This additional detail is useful and relevant but not present in the system requirement.

Additional Conditions Check:

- **SWRS 012 Additional Condition**: SWRS 012 adds an additional condition where, if no mismatch is detected, the error is set to `0x0`. This behavior is not specified in SYSRS 012 but is a reasonable extension to cover the case where no error is present.

  Result: The added condition does not conflict with SYSRS 012 but extends it by specifying what happens if there is no error (setting the error to `0x0`). This is an enhancement, not an unrelated addition.

Ratings and Reasons:

- **Meaningfulness Rating**: **High**
  - **Reason**: The software requirement meaningfully supports the system requirement by implementing the same condition for setting the error when a mismatch is detected. The additional clarification of comparing the **RAM** and **NVM** values of the physical address enhances the meaning of the requirement without deviating from the system's intent.

- **Completeness Rating**: **Medium**
  - **Reason**: While the software requirement extends the system requirement by covering both error and no-error cases, SYSRS 012 only focuses on the error condition. The added detail in SWRS 012 (comparing **RAM** and **NVM** values and specifying the no-error case) is valuable, but the system requirement could be more explicit in defining both error and no-error behaviors. Therefore, SWRS 012 completes the requirement in one sense but adds content that SYSRS 012 did not cover explicitly.

- **Overall Quality of the Link**: **High**
  - **Reason**: Despite the additional condition for no error, the software requirement aligns well with the system requirement in terms of handling the mismatch scenario and amplifying the details. The clarifications around **RAM** and **NVM** comparisons and handling both error and no-error states improve the implementation without diverging from the system requirement's intent.

Suggested Revisions:

- **SYSRS 012** could be enhanced to include the behavior when there is **no mismatch**, similar to how SWRS 012 defines it:
  - Add a condition that specifies setting the error to `0x0` if the physical addresses match.
  - Clarify whether the system requirement intends the comparison to be between **RAM** and **NVM** values, as detailed in SWRS 012.

A possible revision to SYSRS 012 could be:

- "Before leaving Discovery Mode, if the effective value of the DSI3_physicalAddress (RAM) is not equal to the default value of the DSI3_physicalAddress (NVM), the Sensor IC shall set `Error: err_discPhysAddrMismatch = 0x1`. Otherwise, the Sensor IC shall set `Error: err_discPhysAddrMismatch = 0x0`."

This would ensure full alignment between the system and software requirements.

"SYSRS 010: If Performing Memory Area Modification
AND AFTER reception of CRM_CMD 0x9: Take Data or Send Data
THEN
the Sensor IC shall
cache the CRM_CMD 0x9 Parameter Data to the Data Cache.

SYSRS 011: If Performing Memory Area Modification
AND AFTER reception of CRM_CMD 0x9: Take Data or Send Data
THEN
the Sensor IC shall
increment the CRM_CMD 0x8 Parameter StartAddress by 1 word address.

" "SWRS 011: If Performing Memory Area Modification
AND AFTER reception of CRM_CMD 0x9: Take Data or Send Data
THEN
the Software shall store the CRM_CMD 0x9 Parameter Data to RAM until the command CRM_CMD 0xA: Perform Memory Operation with CRM_CMD 0xA Parameter MemoryOperation == 0x2 ""Write to NVM"" is received."

System Requirements (SYSRS 010 and SYSRS 011):

- **SYSRS 010**: If performing memory area modification and after receiving CRM_CMD 0x9 (Take Data or Send Data), the Sensor IC shall cache the CRM_CMD 0x9 parameter data to the Data Cache.
- **SYSRS 011**: If performing memory area modification and after receiving CRM_CMD 0x9 (Take Data or Send Data), the Sensor IC shall increment the CRM_CMD 0x8 parameter StartAddress by 1 word address.

Software Requirement (SWRS 011):

- **Description**: If performing memory area modification and after receiving CRM_CMD 0x9 (Take Data or Send Data), the software shall store the CRM_CMD 0x9 parameter data to RAM until the CRM_CMD 0xA (Perform Memory Operation) is received with CRM_CMD 0xA parameter MemoryOperation == 0x2 ("Write to NVM").

Condition Check:

- **SYSRS 010 and SYSRS 011 Conditions**:
  - The condition in both system requirements involves performing memory area modification **after** the reception of CRM_CMD 0x9 (Take Data or Send Data).
  - **SYSRS 010** requires caching the data from CRM_CMD 0x9.

- **SYSRS 011** requires incrementing the StartAddress by 1 word address after receiving CRM_CMD 0x9.

- **SWRS 011 Conditions**:
  - The software requirement similarly involves memory area modification **after** receiving CRM_CMD 0x9, but the software stores the data to **RAM**, not the Data Cache.
  - Additionally, SWRS 011 introduces a condition where the data is kept in RAM until CRM_CMD 0xA is received, with a specific parameter (MemoryOperation == 0x2) indicating a write to NVM.

Issue: SWRS 011 does not explicitly address the caching of data in the Data Cache (as required by SYSRS 010) or the incrementing of the StartAddress by 1 word (as required by SYSRS 011). Instead, SWRS 011 focuses on storing data to RAM and waiting for a subsequent command (CRM_CMD 0xA) to write to NVM. The conditions for storing data to RAM and waiting for the CRM_CMD 0xA operation are additional steps not mentioned in SYSRS 010 or SYSRS 011.

Action and Response Check:

- **SYSRS 010 Action/Response**: The system requirement specifies that the Sensor IC must cache the CRM_CMD 0x9 parameter data to the Data Cache.

- **SYSRS 011 Action/Response**: The system requirement specifies that the StartAddress must be incremented by 1 word after receiving CRM_CMD 0x9.

- **SWRS 011 Action/Response**: The software stores the CRM_CMD 0x9 parameter data in RAM until CRM_CMD 0xA is received to trigger a write to NVM, which is not explicitly mentioned in SYSRS 010 or SYSRS 011. SWRS 011 does not mention the increment of the StartAddress or the caching of data into the Data Cache as required by SYSRS 010 and SYSRS 011.

Issue: The actions described in SWRS 011 do not fully match the system requirements. SWRS 011 focuses on storing data in RAM and awaiting a memory operation, whereas the system requirements specify caching data in the Data Cache and incrementing the StartAddress.

Parameter Sources Check:

- **SYSRS 010 and SYSRS 011 Parameters**:
  - **SYSRS 010** involves the CRM_CMD 0x9 Parameter Data, which needs to be cached in the Data Cache.
  - **SYSRS 011** involves incrementing the CRM_CMD 0x8 Parameter StartAddress by 1 word.

- **SWRS 011 Parameters**:
  - SWRS 011 also references the CRM_CMD 0x9 Parameter Data but stores it in RAM rather than caching it.
  - The CRM_CMD 0x8 StartAddress parameter is not addressed in SWRS 011.

Issue: SWRS 011 references the same CRM_CMD 0x9 Parameter Data, but it is stored in RAM, not the Data Cache. Additionally, SWRS 011 does not address the CRM_CMD 0x8 StartAddress increment specified in SYSRS 011.

Additional Conditions Check:

- **SWRS 011 Additional Condition**: SWRS 011 introduces an additional condition where the data is stored in RAM until CRM_CMD 0xA is received, with MemoryOperation == 0x2 ("Write to NVM"). This condition is not present in SYSRS 010 or SYSRS 011 and adds complexity that the system requirements do not mention.

  Issue: The additional condition of waiting for CRM_CMD 0xA to write data to NVM introduces an extra step that is not specified in the system requirements.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement generally addresses storing the CRM_CMD 0x9 parameter data, but it deviates from the system requirements by focusing on RAM instead of the Data Cache. Additionally, it introduces extra conditions (waiting for CRM_CMD 0xA) that are not covered in SYSRS 010 or SYSRS 011. SWRS 011 is partially meaningful but does not directly support the actions required by the system requirements.

- **Completeness Rating**: **Low**
  - **Reason**: The software requirement does not address the complete set of actions specified in the system requirements. It omits the caching to the Data Cache required by SYSRS 010 and the incrementing of the StartAddress required by SYSRS 011. Instead, it adds an additional step involving CRM_CMD 0xA, which reduces its completeness relative to the system requirements.

- **Overall Quality of the Link**: **Low**
  - **Reason**: The software requirement diverges from the system requirements in significant ways, particularly by omitting key actions (Data Cache usage and StartAddress increment) and introducing additional conditions (waiting for CRM_CMD 0xA). As a result, the overall quality of the link is rated as Low.

Suggested Revisions:

- **SWRS 011** should be revised to align more closely with the system requirements. Specifically:
  - The software should cache the CRM_CMD 0x9 parameter data in the **Data Cache** (as specified in SYSRS 010) rather than storing it only in RAM.
  - The software should increment the **CRM_CMD 0x8 Parameter StartAddress by 1 word** (as specified in SYSRS 011).
  - If the additional behavior (waiting for CRM_CMD 0xA to write to NVM) is needed, this should be explicitly added to the system requirements.

A revised version of SWRS 011 could be:

- "If Performing Memory Area Modification AND AFTER reception of CRM_CMD 0x9: Take Data or Send Data THEN the Software shall cache the CRM_CMD 0x9 Parameter Data to the Data Cache and increment the CRM_CMD 0x8 Parameter StartAddress by 1 word. The software may store the

data in RAM until the command CRM_CMD 0xA is received for the MemoryOperation == 0x2 (Write to NVM)."

This would ensure the software requirement is fully aligned with the system requirements.

"SYSRS 009: If Performing Memory Access in Broadcast
AND IF the state if the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea for the operation as determined by CRM_CMD 0x8 Operation Flag is in the state Locked,
THEN
the Sensor IC shall
respond with PDCM_RSP P0: Report CRM_RSP Information containing CRM_RSP 0x8: Report Lock Status and CRM_RSP 0x8 Flag KeyRequest = 0b1 (""key requested"").

" "SWRS 010: If Performing Memory Access in Broadcast
AND IF the state if the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea for the operation as determined by CRM_CMD 0x8 Operation Flag is in the state Locked,
THEN
the Software shall prepare to
respond with PDCM_RSP P0: Report CRM_RSP Information containing CRM_RSP 0x8: Report Lock Status and CRM_RSP 0x8 Flag KeyRequest = 0b1 (""key requested"").
"

System Requirement (SYSRS 009):

- **Description**: If performing memory access in broadcast mode and the memory area identified by CRM_CMD 0x8 (MemoryArea) is in the **Locked** state (as determined by the CRM_CMD 0x8 Operation Flag), the Sensor IC shall respond with `PDCM_RSP P0: Report CRM_RSP Information` containing `CRM_RSP 0x8: Report Lock Status` and `CRM_RSP 0x8 Flag KeyRequest = 0b1` (indicating "key requested").

Software Requirement (SWRS 010):

- **Description**: If performing memory access in broadcast mode and the memory area identified by CRM_CMD 0x8 (MemoryArea) is in the **Locked** state (as determined by the CRM_CMD 0x8 Operation Flag), the software shall prepare to respond with `PDCM_RSP P0: Report CRM_RSP Information` containing `CRM_RSP 0x8: Report Lock Status` and `CRM_RSP 0x8 Flag KeyRequest = 0b1` (indicating "key requested").

Condition Check:

- **SYSRS 009 Conditions**:
  - If performing memory access in **broadcast mode**.
  - The memory area identified by **CRM_CMD 0x8 (MemoryArea)** is in the **Locked** state (determined by the Operation Flag).

- Action: The Sensor IC shall respond with `PDCM_RSP P0` containing `CRM_RSP 0x8: Report Lock Status` and `CRM_RSP 0x8 Flag KeyRequest = 0b1`.

- **SWRS 010 Conditions**:
  - If performing memory access in **broadcast mode**.
  - The memory area identified by **CRM_CMD 0x8 (MemoryArea)** is in the **Locked** state (determined by the Operation Flag).
  - Action: The software shall **prepare** to respond with `PDCM_RSP P0` containing `CRM_RSP 0x8: Report Lock Status` and `CRM_RSP 0x8 Flag KeyRequest = 0b1`.

  Result: The conditions for initiating the response are identical between SYSRS 009 and SWRS 010. The software requirement introduces the phrase "prepare to respond," which suggests an intermediate step before the actual response is sent. This may indicate that the software is setting up for the response but does not yet send it directly, which differs slightly from the system requirement, where the Sensor IC "shall respond."

Action and Response Check:

- **SYSRS 009 Action/Response**: The Sensor IC shall respond by sending the `PDCM_RSP P0` report containing the lock status and the key request flag.

- **SWRS 010 Action/Response**: The software prepares to send the same `PDCM_RSP P0` report but does not specify whether it actually completes the response.

  Issue: While both requirements specify the same report and content (including the lock status and key request flag), SWRS 010 stops short of saying the software will **send** the response. It only mentions that the software will "prepare to respond," which could imply that another step or condition is needed before the actual transmission. This discrepancy creates a small gap between the two requirements.

Parameter Sources Check:

- **SYSRS 009 Parameters**: The requirement references parameters from CRM_CMD 0x8 (MemoryArea and Operation Flag) and requires generating a `PDCM_RSP` containing the `CRM_RSP 0x8: Report Lock Status` with the `KeyRequest` flag set to `0b1`.

- **SWRS 010 Parameters**: The software requirement similarly references parameters from CRM_CMD 0x8 and specifies preparing the same response containing the `CRM_RSP 0x8: Report Lock Status` with the `KeyRequest` flag set to `0b1`.

  Result: The parameters in both requirements are identical and correctly handled.

Additional Conditions Check:

- **SWRS 010 Additional Condition**: The only additional distinction is the notion of "preparing to respond," which is not mentioned in SYSRS 009. This intermediate step does not conflict with the system requirement but could suggest a staging step before the response is sent.

Result: The additional condition of "preparing" in SWRS 010 does not introduce irrelevant information but might indicate an incomplete implementation of the final response action.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement generally aligns with the system requirement by ensuring that the correct response (lock status and key request flag) is prepared. However, SWRS 010 does not confirm that the response is actually sent, which leaves some ambiguity. The use of "prepare to respond" suggests an incomplete action relative to the direct response required by SYSRS 009.

- **Completeness Rating**: **Medium**
  - **Reason**: The software requirement captures the key details of the system requirement, including the conditions and parameters. However, the step of actually **sending** the response is missing, as SWRS 010 only mentions preparation. The omission of this final action means that the link is incomplete.

- **Overall Quality of the Link**: **Medium**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Medium, the overall quality of the link is also Medium. The software requirement captures most of the necessary details but does not fully execute the action of sending the response as required by the system.

Suggested Revisions:

- **SWRS 010** should be revised to ensure that the software not only prepares the response but also **sends** it, aligning with the system requirement. The phrase "prepare to respond" could be modified to indicate that the response is both prepared and transmitted.

A revised version of SWRS 010 could be:

- "If Performing Memory Access in Broadcast AND IF the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea for the operation as determined by CRM_CMD 0x8 Operation Flag is in the state Locked, THEN the Software shall respond with PDCM_RSP P0: Report CRM_RSP Information containing CRM_RSP 0x8: Report Lock Status and CRM_RSP 0x8 Flag KeyRequest = 0b1 ('key requested')."

This revision ensures that the software executes the required action fully, matching the system requirement's behavior.

"SYSRS 008: If Performing Memory Access With Set Key
AND If Write Operation is Requested
AND If Memory Area Is Set Write Accessible
THEN
the Sensor IC shall

set the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea to the state Locked." "SWRS 009: If Performing Memory Access
AND If the Key is NOT Set
AND If Write Operation is Requested
AND If Memory Area Is Set Read Accessible
THEN
the Software shall
set the state of the Memory Area identified by CRM_CMD 0x10 Parameter MemoryArea to the state Locked.
"

System Requirement (SYSRS 008):

- **Description**: If performing memory access with a set key, and if a write operation is requested, and the memory area is set to be write accessible, the Sensor IC shall set the state of the memory area (identified by CRM_CMD 0x8 Parameter MemoryArea) to **Locked**.

Software Requirement (SWRS 009):

- **Description**: If performing memory access without a set key, and if a write operation is requested, and the memory area is set to be read accessible, the software shall set the state of the memory area (identified by CRM_CMD 0x10 Parameter MemoryArea) to **Locked**.

Condition Check:

- **SYSRS 008 Conditions**:
  - **Memory Access with Set Key**.
  - **Write Operation** is requested.
  - The **Memory Area is set to Write Accessible**.
  - **Action**: The Sensor IC shall set the memory area state to **Locked**.

- **SWRS 009 Conditions**:
  - **Memory Access without Set Key**.
  - **Write Operation** is requested.
  - The **Memory Area is set to Read Accessible**.
  - **Action**: The software shall set the memory area state to **Locked**.

  Issue: The conditions between SYSRS 008 and SWRS 009 differ significantly:
  - SYSRS 008 focuses on memory access with a **set key**, while SWRS 009 focuses on memory access **without a set key**.
  - SYSRS 008 addresses memory areas that are **write accessible**, whereas SWRS 009 deals with memory areas that are **read accessible**.
  - SYSRS 008 identifies the memory area using **CRM_CMD 0x8**, while SWRS 009 uses **CRM_CMD 0x10**.

These differences mean that SWRS 009 does not directly support SYSRS 008. The key conditions and memory access types do not align.

Action and Response Check:

- **SYSRS 008 Action/Response**: The system requirement specifies that the Sensor IC shall set the state of the memory area to **Locked** if the conditions are met.
- **SWRS 009 Action/Response**: The software requirement also sets the memory area state to **Locked** but under different conditions (no set key and read accessible memory).

  Issue: Although both SYSRS 008 and SWRS 009 result in setting the memory area state to **Locked**, the conditions leading to this action are different. SYSRS 008 requires a set key and write accessibility, while SWRS 009 handles the case without a key and read accessibility.

Parameter Sources Check:

- **SYSRS 008 Parameters**: The memory area is identified by **CRM_CMD 0x8 Parameter MemoryArea**.
- **SWRS 009 Parameters**: The memory area is identified by **CRM_CMD 0x10 Parameter MemoryArea**.

  Issue: The memory area is referenced by different commands in SYSRS 008 and SWRS 009, which introduces a further inconsistency between the two requirements.

Additional Conditions Check:

- **SWRS 009 Additional Conditions**: The software requirement introduces conditions that are not present in SYSRS 008, such as memory access without a set key and the memory area being read accessible. These additional conditions do not align with the system requirement and point to different use cases.

  Issue: SWRS 009 introduces unrelated conditions that do not match the system requirement, making it an inappropriate implementation for SYSRS 008.

Ratings and Reasons:

- **Meaningfulness Rating**: **Low**
  - **Reason**: SWRS 009 does not meaningfully align with SYSRS 008 due to several discrepancies:
    - Memory access with a set key vs. without a set key.
    - Write accessible memory vs. read accessible memory.
    - Different commands used to identify the memory area.

    These differences indicate that SWRS 009 addresses a different scenario and does not directly support SYSRS 008.
- **Completeness Rating**: **Low**

- **Reason**: SWRS 009 does not complete the system requirement because it handles different conditions and parameters. It does not cover the scenario outlined in SYSRS 008, making the link between the system and software requirements incomplete.

- **Overall Quality of the Link**: **Low**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Low, the overall quality of the link is also Low. SWRS 009 does not provide an appropriate or relevant implementation for the scenario described in SYSRS 008.

Suggested Revisions:

- **SWRS 009** should be revised to match the conditions outlined in SYSRS 008. Specifically, it should:
  - Handle memory access **with a set key**.
  - Focus on memory areas that are **write accessible**, not read accessible.
  - Use **CRM_CMD 0x8 Parameter MemoryArea** to identify the memory area, as specified in SYSRS 008.

A revised version of SWRS 009 could be:

- "If Performing Memory Access With Set Key AND If Write Operation is Requested AND If the Memory Area Is Set Write Accessible THEN the Software shall set the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea to the state Locked."

This revision would ensure the software requirement aligns with the system requirement and addresses the correct conditions.

"SYSRS 007: If Performing Memory Access
AND If the Key is Set
AND If Read Operation is Requested
AND If Memory Area Is Set Read Accessible
THEN
the Sensor IC shall
set the state of the Memory Area identified byCRM_CMD 0x8 Parameter MemoryArea to the state Locked.
" SWRS 008: If Performing Memory Access AND If the Key is Set AND If Read Operation is Requested AND If Memory Area Is Set Read Accessible THEN the Software shall set the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea to the state Locked.

System Requirement (SYSRS 007):

- **Description**: If performing memory access, and if the key is set, and a read operation is requested, and the memory area is set to be read accessible, the Sensor IC shall set the state of the memory area (identified by CRM_CMD 0x8 Parameter MemoryArea) to **Locked**.

Software Requirement (SWRS 008):

- **Description**: If performing memory access, and if the key is set, and a read operation is requested, and the memory area is set to be read accessible, the software shall set the state of the memory area (identified by CRM_CMD 0x8 Parameter MemoryArea) to **Locked**.

Condition Check:

- **SYSRS 007 Conditions**:
  - **Memory Access**.
  - **Key is Set**.
  - **Read Operation** is requested.
  - **Memory Area is Set to Read Accessible**.
  - **Action**: The Sensor IC shall set the memory area state to **Locked**.

- **SWRS 008 Conditions**:
  - The conditions are identical to SYSRS 007:
    - **Memory Access**.
    - **Key is Set**.
    - **Read Operation** is requested.
    - **Memory Area is Set to Read Accessible**.
    - **Action**: The software shall set the memory area state to **Locked**.

  Result: The conditions in both SYSRS 007 and SWRS 008 are identical. Both requirements specify the same conditions for memory access, key setting, read operation, and memory area state. There are no differences in the conditions described.

Action and Response Check:

- **SYSRS 007 Action/Response**: The system requirement specifies that the Sensor IC must set the memory area state to **Locked** if the conditions are met.

- **SWRS 008 Action/Response**: The software requirement specifies that the software must set the memory area state to **Locked** under the same conditions.

  Result: The actions described in both requirements are identical. Both require setting the memory area state to **Locked** if the specified conditions are met.

Parameter Sources Check:

- **SYSRS 007 Parameters**: The system requirement specifies that the memory area is identified by **CRM_CMD 0x8 Parameter MemoryArea**.

- **SWRS 008 Parameters**: The software requirement references the same parameter (**CRM_CMD 0x8 Parameter MemoryArea**) to identify the memory area.

  Result: The parameters are consistent between the system and software requirements, ensuring that both reference the same memory area for the locking operation.

Additional Conditions Check:

- **SWRS 008 Additional Condition**: SWRS 008 does not introduce any additional conditions. It strictly follows the conditions set by SYSRS 007 without adding any irrelevant steps.

  Result: No additional or unrelated conditions are introduced in SWRS 008, maintaining alignment with SYSRS 007.

Ratings and Reasons:

- **Meaningfulness Rating**: **High**
  - **Reason**: The software requirement (SWRS 008) fully supports the system requirement (SYSRS 007) by directly implementing the same conditions and actions. The link between the two is strong and logical, as the software requirement clearly implements the behavior specified by the system requirement.

- **Completeness Rating**: **High**
  - **Reason**: The software requirement covers all aspects of the system requirement. It includes all the necessary conditions (memory access, key set, read operation, and read accessibility) and specifies the correct action (setting the memory area state to **Locked**). There is no missing information, making the link complete.

- **Overall Quality of the Link**: **High**
  - **Reason**: Since both Meaningfulness and Completeness are rated as High, the overall quality of the link is also High. The software requirement is a complete and accurate implementation of the system requirement.

Suggested Revisions:

- **None**: The software requirement (SWRS 008) aligns perfectly with the system requirement (SYSRS 007) and covers all necessary conditions and actions. No revisions are needed, as the software requirement successfully implements the system requirement without any discrepancies.

"SYSRS 006: If Performing Memory Access in Unicast
AND IF the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea for the operation as determined by CRM_CMD 0x8 Operation Flag is in the state Locked,
THEN
the Sensor IC shall
respond with CRM_RSP 0x8: Report Lock Status and CRM_RSP 0x8 Flag KeyRequest = 0b1 (""key requested"").
" "SWRS 007: If Performing Memory Access in Unicast
AND IF the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea for the operation as determined by CRM_CMD 0x8 Operation Flag is in the state Locked,
THEN
the Software shall prepare to
respond with CRM_RSP 0x8: Report Lock Status and CRM_RSP 0x8 Flag KeyRequest = 0b1 (""key

```
requested"").
"
```

System Requirement (SYSRS 006):

- **Description**: If performing memory access in unicast mode, and if the state of the memory area (identified by CRM_CMD 0x8 Parameter MemoryArea for the operation as determined by CRM_CMD 0x8 Operation Flag) is in the **Locked** state, the Sensor IC shall respond with `CRM_RSP 0x8: Report Lock Status` and set the `CRM_RSP 0x8 Flag KeyRequest = 0b1` (indicating "key requested").

Software Requirement (SWRS 007):

- **Description**: If performing memory access in unicast mode, and if the state of the memory area (identified by CRM_CMD 0x8 Parameter MemoryArea for the operation as determined by CRM_CMD 0x8 Operation Flag) is in the **Locked** state, the software shall prepare to respond with `CRM_RSP 0x8: Report Lock Status` and set the `CRM_RSP 0x8 Flag KeyRequest = 0b1` (indicating "key requested").

Condition Check:

- **SYSRS 006 Conditions**:
  - Memory access is performed in **unicast mode**.
  - The memory area (identified by **CRM_CMD 0x8**) is in the **Locked** state (determined by the Operation Flag).
  - **Action**: The Sensor IC shall respond with `CRM_RSP 0x8: Report Lock Status` and set the `KeyRequest` flag to `0b1`.

- **SWRS 007 Conditions**:
  - The conditions are identical to SYSRS 006:
    - Memory access is performed in **unicast mode**.
    - The memory area (identified by **CRM_CMD 0x8**) is in the **Locked** state.
    - **Action**: The software shall **prepare to respond** with `CRM_RSP 0x8: Report Lock Status` and set the `KeyRequest` flag to `0b1`.

  Result: The conditions described in both SYSRS 006 and SWRS 007 are the same, with one minor difference: SWRS 007 states that the software shall "prepare to respond" rather than directly stating that the software shall **respond** as described in SYSRS 006. This slight difference suggests an intermediate step before the actual response is sent, while SYSRS 006 implies that the response is immediate.

Action and Response Check:

- **SYSRS 006 Action/Response**: The system requirement specifies that the Sensor IC shall **respond** with the lock status and the key request flag set to `0b1`.

- **SWRS 007 Action/Response**: The software shall **prepare to respond** with the same information (lock status and key request flag set to `**0b1**`).

  Issue: The phrase "prepare to respond" in SWRS 007 introduces ambiguity. It implies the software is getting ready to send the response but does not explicitly state whether the response is actually sent, whereas SYSRS 006 mandates that the response must be sent directly. This creates a slight difference in how the action is executed, with the software requirement potentially leaving out the final step of actually sending the response.

Parameter Sources Check:

- **SYSRS 006 Parameters**: The system requirement references the memory area using **CRM_CMD 0x8** and requires responding with the `**CRM_RSP 0x8**` response, including the lock status and key request flag (`**KeyRequest = 0b1**`).

- **SWRS 007 Parameters**: The software requirement uses the same parameters, referring to **CRM_CMD 0x8** and the `**CRM_RSP 0x8**` response, including the lock status and key request flag (`**KeyRequest = 0b1**`).

  Result: Both requirements refer to the same parameters and maintain consistency in terms of the content of the response.

Additional Conditions Check:

- **SWRS 007 Additional Condition**: The software requirement introduces the idea of "preparing" the response, which is not mentioned in SYSRS 006. This intermediate step does not conflict with SYSRS 006 but introduces ambiguity as to whether the final action of sending the response is completed.

  Result: The additional notion of "preparing" does not add a completely unrelated condition but suggests that SWRS 007 may not fully cover the action required by SYSRS 006.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement generally supports the system requirement by preparing the correct response (lock status and key request flag). However, the phrase "prepare to respond" introduces ambiguity about whether the software actually sends the response, which reduces the meaningfulness of the link. The system requirement expects the response to be sent, not just prepared.

- **Completeness Rating**: **Medium**
  - **Reason**: The software requirement captures the necessary conditions and parameters from the system requirement, but it does not fully complete the expected action of sending the response. The notion of "preparing" to respond rather than actually responding leaves the link incomplete.

- **Overall Quality of the Link**: **Medium**

- **Reason**: Since both Meaningfulness and Completeness are rated as Medium, the overall quality of the link is also Medium. While SWRS 007 implements most of the system requirement, it does not fully execute the final action of responding as required by SYSRS 006.

Suggested Revisions:

- **SWRS 007** should be revised to ensure that the software not only **prepares** the response but also **sends** it, as specified in the system requirement. The phrase "prepare to respond" should be modified to indicate that the response is both prepared and transmitted.

A revised version of SWRS 007 could be:

- "If Performing Memory Access in Unicast AND IF the state of the Memory Area identified by CRM_CMD 0x8 Parameter MemoryArea for the operation as determined by CRM_CMD 0x8 Operation Flag is in the state Locked, THEN the Software shall respond with CRM_RSP 0x8: Report Lock Status and CRM_RSP 0x8 Flag KeyRequest = 0b1 ('key requested')."

This revision ensures that the software requirement fully aligns with the system requirement and removes any ambiguity about the action being completed.

"SYSRS 005: IF the Sensor IC generates more information than possible to store in Event Buffer or Time Series Data Buffer
THEN
the Sensor IC shall forget the according information." "SWRS 005: - The software shall store information for up to 5 Echo Events.

SWRS 006: - The software shall hold at least the information for up to 100 time series data samples (time stamp and value)
IF NOT possible regarding memory constraints could be reduced up to 50 time series data samples. "

System Requirement (SYSRS 005):

- **Description**: If the Sensor IC generates more information than can be stored in the Event Buffer or Time Series Data Buffer, the Sensor IC shall "forget" (discard) the corresponding information.

Software Requirements (SWRS 005 and SWRS 006):

1. **SWRS 005**: The software shall store information for up to **5 Echo Events**.
2. **SWRS 006**: The software shall hold information for at least **100 time series data samples** (time stamp and value). If memory constraints arise, the software may reduce this to **50 time series data samples**.

Condition Check:

- **SYSRS 005 Conditions**:
  - If the Sensor IC generates more information than the Event Buffer or Time Series Data Buffer can store, the excess information must be discarded (forgotten).

- **SWRS 005 and 006 Conditions**:
  - SWRS 005 specifies storing information for **up to 5 Echo Events**.
  - SWRS 006 specifies holding information for at least **100 time series data samples**, with a possible reduction to **50 samples** due to memory constraints.

  Result: The conditions in SWRS 005 and SWRS 006 are somewhat related to SYSRS 005, as they describe limits on how much information can be stored. However, **neither SWRS 005 nor SWRS 006 directly addresses the action of discarding information** when those limits are exceeded, as specified in SYSRS 005. There is no mention of what happens when the event or time series data buffers are full.

Action and Response Check:

- **SYSRS 005 Action/Response**: When the event or time series data buffers are full, the system must discard any additional information that cannot be stored.

- **SWRS 005 and 006 Action/Response**:
  - SWRS 005 describes storing a maximum of **5 Echo Events**, but it does not specify what happens when more than 5 events are generated.
  - SWRS 006 describes storing **100 time series data samples** (or reducing to 50 if constrained) but does not address what the software should do when more data is generated than the buffer can handle.

  Issue: The software requirements describe limits for storing echo events and time series data but do not specify the action of **discarding excess information** as required by SYSRS 005.

Parameter Sources Check:

- **SYSRS 005 Parameters**: The system requirement implicitly refers to the "Event Buffer" and "Time Series Data Buffer" as the locations where data is stored.

- **SWRS 005 and 006 Parameters**:
  - SWRS 005 refers to storing up to **5 Echo Events**.
  - SWRS 006 refers to storing up to **100 time series data samples** (with the possibility of reducing to 50).

  Result: The parameters used in SWRS 005 and SWRS 006 are consistent with the data types (echo events and time series data) referenced in SYSRS 005. However, the software requirements lack explicit references to **buffers** and the handling of buffer overflows, which is critical in SYSRS 005.

Additional Conditions Check:

- **SWRS 005 and 006 Additional Conditions**: The software requirements introduce storage limits (5 echo events, 100 or 50 time series data samples) without mentioning what happens when

these limits are exceeded.

Issue: While the storage limits are relevant to the system requirement, the **lack of an action plan for buffer overflows** (i.e., discarding excess data) leaves SWRS 005 and SWRS 006 incomplete relative to SYSRS 005.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirements partially support the system requirement by setting limits on how much data (echo events and time series data) can be stored. However, they do not directly address the key action required by SYSRS 005, which is discarding information when the buffers are full. This reduces the meaningfulness of the link between the system and software requirements.

- **Completeness Rating**: **Low**
  - **Reason**: The software requirements fail to mention what happens when the buffers are full. While SWRS 005 and SWRS 006 define storage limits, they do not describe how the system should handle the overflow condition specified in SYSRS 005. As a result, the software requirements are incomplete with respect to the system requirement's core action.

- **Overall Quality of the Link**: **Low**
  - **Reason**: Since the software requirements do not fully implement the key behavior of discarding excess information when buffers are full, the overall quality of the link is Low. The software requirements partially address the storage capacity but omit the critical overflow handling described in SYSRS 005.

Suggested Revisions:

- **SWRS 005 and SWRS 006** should be revised to include explicit handling for buffer overflow situations, in line with SYSRS 005. Specifically:
  - SWRS 005 should state what happens if more than 5 Echo Events are generated and how excess events are handled (e.g., discarded).
  - SWRS 006 should specify how the software should behave when more than the allotted number of time series data samples (100 or 50) are generated. For example, it could state that excess data samples are discarded when the buffer is full.

A possible revision could be:

- **SWRS 005**: "The software shall store information for up to 5 Echo Events. If more than 5 Echo Events are generated, the software shall discard the oldest event to store new data."
- **SWRS 006**: "The software shall hold at least 100 time series data samples (time stamp and value). If memory constraints arise, this could be reduced to 50 time series data samples. If more data samples are generated than can be stored, the software shall discard the oldest samples to store new data."

These changes would ensure that the software requirements handle buffer overflows as required by the system requirement.

"SYSRS 004: If performing Erase Backup
AND If Addressed with the Unicast Command
THEN
the Sensor IC shall
respond with CRM_RSP 0x0: Report Acknowledgement." "SWRS 004: If performing Erase Backup
AND If Addressed with the Unicast Command
THEN
the Software shall prepare to
respond with CRM_RSP 0x0: Report Acknowledgement."

System Requirement (SYSRS 004):

- **Description**: If performing **Erase Backup** and addressed with a **Unicast Command**, the Sensor IC shall respond with `CRM_RSP 0x0: Report Acknowledgement`.

Software Requirement (SWRS 004):

- **Description**: If performing **Erase Backup** and addressed with a **Unicast Command**, the software shall prepare to respond with `CRM_RSP 0x0: Report Acknowledgement`.

Condition Check:

- **SYSRS 004 Conditions**:
  - Performing **Erase Backup**.
  - Addressed with a **Unicast Command**.
  - Action: The Sensor IC shall respond with `CRM_RSP 0x0: Report Acknowledgement`.

- **SWRS 004 Conditions**:
  - The conditions are identical:
    - Performing **Erase Backup**.
    - Addressed with a **Unicast Command**.
    - Action: The software shall **prepare to respond** with `CRM_RSP 0x0: Report Acknowledgement`.

  Result: The conditions in both SYSRS 004 and SWRS 004 are the same, with the only difference being that SWRS 004 mentions "preparing to respond" rather than actually **sending** the response. This suggests an intermediate step or potential delay before the response is sent, which is not mentioned in SYSRS 004.

Action and Response Check:

- **SYSRS 004 Action/Response**: The system requirement specifies that the Sensor IC shall directly respond with `CRM_RSP 0x0: Report Acknowledgement`.

- **SWRS 004 Action/Response**: The software prepares to send the same acknowledgement (`CRM_RSP 0x0`), but does not explicitly state that it **sends** the response.

  Issue: The phrase "prepare to respond" in SWRS 004 introduces ambiguity. It implies that the software is getting ready to send the response but does not confirm that the response is actually sent. SYSRS 004 requires that the response be sent immediately, so SWRS 004 does not fully implement the required action.

Parameter Sources Check:

- **SYSRS 004 Parameters**: The system requirement references the `CRM_RSP 0x0` response, which is an acknowledgement.

- **SWRS 004 Parameters**: The software requirement also references the same `CRM_RSP 0x0` acknowledgement response.

  Result: Both requirements use the same parameter, ensuring consistency in the content of the response. However, SWRS 004 leaves out the confirmation of actually sending the response.

Additional Conditions Check:

- **SWRS 004 Additional Condition**: The software requirement adds the condition of "preparing to respond," which is not mentioned in SYSRS 004. This additional condition does not conflict with the system requirement but adds ambiguity as to whether the final action of sending the response is completed.

  Result: The additional notion of "preparing" does not add a completely unrelated condition but suggests that SWRS 004 may not fully cover the action required by SYSRS 004.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement generally aligns with the system requirement by preparing the correct response (`CRM_RSP 0x0: Report Acknowledgement`). However, the phrase "prepare to respond" introduces ambiguity about whether the software actually sends the response, which reduces the meaningfulness of the link. The system requirement expects the response to be sent, not just prepared.

- **Completeness Rating**: **Medium**
  - **Reason**: The software requirement captures the necessary conditions and parameters from the system requirement, but it does not fully complete the expected action of **sending** the response. The notion of "preparing to respond" rather than actually responding leaves the link incomplete.

- **Overall Quality of the Link**: **Medium**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Medium, the overall quality of the link is also Medium. While SWRS 004 implements most of the system

requirement, it does not fully execute the final action of responding as required by SYSRS 004.

Suggested Revisions:

- **SWRS 004** should be revised to ensure that the software not only **prepares** the response but also **sends** it, as specified in the system requirement. The phrase "prepare to respond" should be modified to indicate that the response is both prepared and transmitted.

A revised version of SWRS 004 could be:

- "If performing Erase Backup AND If Addressed with the Unicast Command THEN the Software shall respond with CRM_RSP 0x0: Report Acknowledgement."

This revision ensures that the software requirement fully aligns with the system requirement and removes any ambiguity about the action being completed.

"SYSRS 003: BEFORE Sending CRM_RCC After Reception of the CRM_FCC,
THEN
the Sensor IC shall
set CRM_RSP Parameter CrmStatus = Status Value.
" "SWRS 003: BEFORE responding with the CRM_RSP,
the Software shall include the CRM_RSP Parameter CrmStatus into the CRM_RSP.

"

System Requirement (SYSRS 003):

- **Description**: **Before** sending the **CRM_RCC** (response command) after receiving the **CRM_FCC** (command), the Sensor IC shall set the **CRM_RSP** parameter **CrmStatus** to the **Status Value**.

Software Requirement (SWRS 003):

- **Description**: **Before** responding with the **CRM_RSP** (response), the software shall include the **CRM_RSP** parameter **CrmStatus** in the response.

Condition Check:

- **SYSRS 003 Conditions**:
  - **Before sending CRM_RCC** after receiving CRM_FCC.
  - Action: The Sensor IC shall set the **CrmStatus** parameter to **Status Value**.

- **SWRS 003 Conditions**:
  - **Before responding with CRM_RSP**.

- Action: The software shall include the **CrmStatus** parameter in the response.

Issue: Both SYSRS 003 and SWRS 003 involve actions that occur **before responding** with CRM_RSP. However, SYSRS 003 specifies that the **CrmStatus** must be set to a specific **Status Value**, while SWRS 003 simply mentions that the **CrmStatus** parameter is included in the response. The critical detail of setting **CrmStatus** to a **Status Value** is missing in SWRS 003.

Action and Response Check:

- **SYSRS 003 Action/Response**: The Sensor IC shall set the **CrmStatus** to a specific **Status Value** before sending the response.
- **SWRS 003 Action/Response**: The software includes the **CrmStatus** parameter in the response but does not mention that it is set to a specific value.

  Issue: The action described in SWRS 003 does not fully align with SYSRS 003. SYSRS 003 requires the **CrmStatus** to be set to a **specific Status Value**, while SWRS 003 only indicates that the parameter is included without specifying any particular value being assigned.

Parameter Sources Check:

- **SYSRS 003 Parameters**: The system requirement explicitly references the **CrmStatus** parameter, which must be set to a **Status Value**.
- **SWRS 003 Parameters**: The software requirement also references the **CrmStatus** parameter but does not mention assigning it a specific **Status Value**.

  Issue: SWRS 003 refers to the **CrmStatus** parameter but lacks the detail of setting it to a specific value, which is required by SYSRS 003.

Additional Conditions Check:

- **SWRS 003 Additional Condition**: SWRS 003 introduces no additional conditions beyond including the **CrmStatus** in the response.

  Result: No unrelated additional conditions are introduced, but SWRS 003 omits a key detail of assigning the **Status Value** to **CrmStatus** before responding.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement partially supports the system requirement by including the **CrmStatus** parameter in the response. However, it does not mention the critical step of setting the **CrmStatus** to a **specific Status Value**, which is required by SYSRS 003. This reduces the meaningfulness of the link.
- **Completeness Rating**: **Low**

  - **Reason**: SWRS 003 does not complete the system requirement, as it omits the necessary action of setting the **CrmStatus** to a **specific Status Value**. The software requirement only mentions including the parameter, but the critical assignment of the status value is missing.

- **Overall Quality of the Link**: **Medium**
  - **Reason**: The overall quality is rated as Medium because, although the software requirement refers to the **CrmStatus** parameter, it omits the essential detail of setting it to a specific value. This omission results in a partial implementation of the system requirement.

Suggested Revisions:

- **SWRS 003** should be revised to ensure that the software not only includes the **CrmStatus** parameter but also sets it to a specific **Status Value** before responding.

A revised version of SWRS 003 could be:

- "Before responding with the CRM_RSP, the Software shall set the CRM_RSP Parameter CrmStatus to the Status Value and include it in the response."

This revision would ensure that the software requirement fully aligns with the system requirement and implements the necessary step of setting the **CrmStatus** parameter to a specific value.

SYSRS 002: The Sensor IC FW shall have a watchdog to detect FW or HW malfunction by resetting the WDG time in appropriate time within the FTTI. SWRS 002: The software shall enable a watchdog to detect software or hardware malfunction AND IF a malfunction is present, perform a reset in an appropriate time within the Fault Tolerant Time Interval of 20ms

System Requirement (SYSRS 002):

- **Description**: The Sensor IC firmware (FW) shall have a watchdog to detect firmware (FW) or hardware (HW) malfunctions by resetting the watchdog (WDG) timer within the Fault Tolerant Time Interval (FTTI).

Software Requirement (SWRS 002):

- **Description**: The software shall enable a watchdog to detect software or hardware malfunctions. If a malfunction is detected, the software shall perform a reset within the appropriate time, specifically within the **Fault Tolerant Time Interval (FTTI)** of **20ms**.

Condition Check:

- **SYSRS 002 Conditions**:

  - A watchdog is required to detect **FW or HW malfunctions**.
  - The watchdog timer must be reset within the FTTI to ensure functionality.

- **SWRS 002 Conditions**:
  - The software must enable a watchdog to detect **software or hardware malfunctions**.
  - If a malfunction is present, the software performs a **reset** within the FTTI, which is specified as **20ms**.

  Result: Both requirements refer to using a watchdog to detect **FW/HW malfunctions**. However, SYSRS 002 emphasizes **resetting the watchdog timer** within the FTTI, while SWRS 002 introduces the condition of performing a **reset** in the case of a malfunction, with the FTTI explicitly defined as **20ms**. There is a subtle difference in focus: SYSRS 002 highlights **resetting the timer** (prevention), while SWRS 002 emphasizes performing a reset (recovery).

Action and Response Check:

- **SYSRS 002 Action/Response**: The system requirement specifies that the watchdog timer must be reset within the FTTI to prevent malfunction detection.

- **SWRS 002 Action/Response**: The software requirement focuses on enabling the watchdog and performing a **reset** if a malfunction occurs within the FTTI of 20ms.

  Issue: SWRS 002 does not explicitly mention the act of **resetting the watchdog timer** within the FTTI to prevent malfunctions, as specified in SYSRS 002. Instead, SWRS 002 focuses on performing a system reset in response to a detected malfunction. This suggests that the software requirement is addressing a recovery mechanism rather than the preventive mechanism described in SYSRS 002.

Parameter Sources Check:

- **SYSRS 002 Parameters**: The system requirement refers to the FTTI, which is implied but not explicitly defined.

- **SWRS 002 Parameters**: The software requirement explicitly defines the FTTI as **20ms**, which aligns with the implied FTTI reference in SYSRS 002.

  Result: SWRS 002 adds helpful detail by specifying the FTTI as **20ms**, which is not mentioned explicitly in SYSRS 002 but is a reasonable extension.

Additional Conditions Check:

- **SWRS 002 Additional Condition**: SWRS 002 introduces the condition of performing a system reset when a malfunction is detected. This condition is not directly mentioned in SYSRS 002, which focuses more on the watchdog timer resetting within the FTTI to prevent malfunction detection.

  Issue: SWRS 002 adds a response to malfunction (reset), which, while useful, differs slightly from the intent of SYSRS 002, which focuses more on prevention (resetting the watchdog timer). The

additional condition of **resetting the system** is a reactive step that is not mentioned in SYSRS 002.

Ratings and Reasons:

- **Meaningfulness Rating**: **Medium**
  - **Reason**: The software requirement (SWRS 002) supports the system requirement (SYSRS 002) by enabling a watchdog to detect malfunctions. However, SWRS 002 focuses on performing a **reset** when a malfunction is detected, rather than emphasizing the preventive action of **resetting the watchdog timer** within the FTTI. This shift in focus from prevention to recovery reduces the meaningfulness of the link.

- **Completeness Rating**: **Medium**
  - **Reason**: SWRS 002 covers most of the intent behind SYSRS 002 by introducing a watchdog mechanism and specifying a response within the FTTI. However, the preventive action of **resetting the watchdog timer** within the FTTI is missing from the software requirement, leaving it incomplete.

- **Overall Quality of the Link**: **Medium**
  - **Reason**: Since both Meaningfulness and Completeness are rated as Medium, the overall quality of the link is also Medium. SWRS 002 captures part of the intent of SYSRS 002 but focuses on recovery (reset) rather than prevention (watchdog timer reset), which leaves room for improvement in aligning the requirements.

Suggested Revisions:

- **SWRS 002** should be revised to incorporate the notion of **resetting the watchdog timer** to prevent malfunctions within the FTTI, in addition to the system reset action.

A possible revision of SWRS 002 could be:

- "The software shall enable a watchdog to detect software or hardware malfunction. The software shall reset the watchdog timer in an appropriate time within the Fault Tolerant Time Interval (FTTI) of 20ms to prevent malfunction detection. If a malfunction is detected, the software shall perform a reset within the same interval."

This revision would ensure that both the **prevention** (resetting the watchdog timer) and **recovery** (resetting the system) actions are covered, aligning SWRS 002 more closely with SYSRS 002.

SYSRS 001: The Sensor IC shall digitize the Transducer Input Signal at the IC Pins with sampling frequency of (CALIB_samplingFreq plus YPX_samplingFreqOffset). "SWRS 001: AFTER reception of the command CRM_CMD 0x1: Start Standard Measurement
AND
BEFORE the start of the Measurement the software shall set the register BRG.FS.f_s to (CALIB_samplingFreq plus YPX_samplingFreqOffset). "

System Requirement (SYSRS 001):

- **Description**: The Sensor IC shall digitize the transducer input signal at the IC pins with a sampling frequency of **(CALIB_samplingFreq + YPX_samplingFreqOffset)**.

Software Requirement (SWRS 001):

- **Description**: After receiving the command **CRM_CMD 0x1: Start Standard Measurement**, and **before** the start of the measurement, the software shall set the register **BRG.FS.f_s** to **(CALIB_samplingFreq + YPX_samplingFreqOffset)**.

Condition Check:

- **SYSRS 001 Conditions**:
  - The Sensor IC must digitize the input signal with a sampling frequency of **(CALIB_samplingFreq + YPX_samplingFreqOffset)**.
  - The sampling frequency is applied directly to the digitization process.

- **SWRS 001 Conditions**:
  - The software requirement specifies that after receiving **CRM_CMD 0x1** (Start Standard Measurement) and **before** the measurement begins, the software shall set the register **BRG.FS.f_s** to **(CALIB_samplingFreq + YPX_samplingFreqOffset)**.

  Result: Both requirements mention setting the sampling frequency to **(CALIB_samplingFreq + YPX_samplingFreqOffset)**. However, **SWRS 001 introduces additional conditions** that are not specified in SYSRS 001:
  - SWRS 001 refers to the **Start Standard Measurement** command (CRM_CMD 0x1).
  - It also specifies setting the register **before** the measurement starts, which is a procedural step that SYSRS 001 does not detail.

  These additional conditions are related to the operational context but are not explicitly mentioned in SYSRS 001.

Action and Response Check:

- **SYSRS 001 Action/Response**: The system requirement specifies that the Sensor IC shall digitize the input signal using the specified sampling frequency formula.

- **SWRS 001 Action/Response**: The software requirement specifies that the software shall set the sampling frequency register **BRG.FS.f_s** to the same formula before starting the measurement.

  Result: The core action—applying the sampling frequency based on the formula **(CALIB_samplingFreq + YPX_samplingFreqOffset)**—is consistent across both requirements. SWRS 001 provides more detail on when and how this setting is applied (before the measurement starts, after receiving CRM_CMD 0x1).

Parameter Sources Check:

- **SYSRS 001 Parameters**: The system requirement involves the parameters **CALIB_samplingFreq** and **YPX_samplingFreqOffset**.

- **SWRS 001 Parameters**: The software requirement also involves the same parameters, **CALIB_samplingFreq** and **YPX_samplingFreqOffset**, and applies them to the register **BRG.FS.f_s**.

  Result: The parameters are consistent across both requirements, and SWRS 001 ensures that the appropriate register is set with the correct sampling frequency.

Additional Conditions Check:

- **SWRS 001 Additional Conditions**: SWRS 001 introduces additional conditions such as receiving **CRM_CMD 0x1** (Start Standard Measurement) and setting the frequency before the measurement begins. These conditions are not mentioned in SYSRS 001 but are relevant to the overall process.

  Result: The additional conditions in SWRS 001 are operational and procedural details that support the execution of the system requirement but are not contradictory to SYSRS 001. However, they add more steps to the process that are not explicitly required by SYSRS 001.

Ratings and Reasons:

- **Meaningfulness Rating**: **High**
  - **Reason**: The software requirement (SWRS 001) meaningfully supports the system requirement (SYSRS 001) by setting the appropriate sampling frequency in the register before starting the measurement. The additional conditions in SWRS 001 (CRM_CMD 0x1 and before measurement) add operational context but do not detract from the overall meaning.

- **Completeness Rating**: **High**
  - **Reason**: The software requirement fully implements the system requirement by setting the required sampling frequency formula. Although SWRS 001 introduces additional steps (CRM_CMD 0x1 and timing details), it does not omit any critical aspects of SYSRS 001. The link is complete.

- **Overall Quality of the Link**: **High**
  - **Reason**: Since both Meaningfulness and Completeness are rated as High, the overall quality of the link is also High. SWRS 001 provides a full and detailed implementation of SYSRS 001, ensuring that the correct sampling frequency is set and digitization proceeds as expected.

Suggested Revisions:

- **No revisions are necessary**. SWRS 001 aligns well with SYSRS 001 and provides additional useful context (CRM_CMD 0x1 and timing), which does not contradict the system requirement but

rather enhances the overall process.

This conversation is archived. To continue, please unarchive it first.

Unarchive