MOST

Media **O**riented **S**ystems **T**ransport

Multimedia and Control Networking Technology

Evaluation of Diagnostic Information by a Central Component

Rev. 1.0

09/2009

MOSTCO CONFIDENTIAL

See page 3 for the terms of disclosure

Legal Notice

COPYRIGHT

LICENSE DISCLAIMER

Nothing on any MOST Cooperation Web Site, or in any MOST Cooperation document, shall be construed as conferring any license under any of the MOST Cooperation or its members or any third party's intellectual property rights, whether by estoppel, implication, or otherwise.

CONTENT AND LIABILITY DISCLAIMER

MOST Cooperation or its members shall not be responsible for any errors or omissions contained at any MOST Cooperation Web Site, or in any MOST Cooperation document, and reserves the right to make changes without notice. Accordingly, all MOST Cooperation and third party information is provided "AS IS". In addition, MOST Cooperation or its members are not responsible for the content of any other Web Site linked to any MOST Cooperation Web Site. Links are provided as Internet navigation tools only.

MOST COOPERATION AND ITS MEMBERS DISCLAIM ALL WARRANTIES WITH REGARD TO THE INFORMATION (INCLUDING ANY SOFTWARE) PROVIDED, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to you.

In no event shall MOST Cooperation or its members be liable for any damages whatsoever, and in particular MOST Cooperation or its members shall not be liable for special, indirect, consequential, or incidental damages, or damages for lost profits, loss of revenue, or loss of use, arising out of or related to any MOST Cooperation Web Site, any MOST Cooperation document, or the information contained in it, whether such damages arise in contract, negligence, tort, under statute, in equity, at law or otherwise.

FEEDBACK INFORMATION

Any information provided to MOST Cooperation in connection with any MOST Cooperation Web Site, or any MOST Cooperation document, shall be provided by the submitter and received by MOST Cooperation on a non-confidential basis. MOST Cooperation shall be free to use such information on an unrestricted basis.

TRADEMARKS

MOST Cooperation and its members prohibit the unauthorized use of any of their trademarks. MOST Cooperation specifically prohibits the use of the MOST Cooperation LOGO unless the use is approved by the Steering Committee of MOST Cooperation.

SUPPORT AND FURTHER INFORMATION

For more information on the MOST technology, please contact:

MOST Cooperation
Administration
D-76185 Karlsruhe
Germany

Tel: (+49) (0) 721 966 50 00
Fax: (+49) (0) 721 966 50 01
E-mail: contact@mostcooperation.com
Web: www.mostcooperation.com

This Specification is Confidential Information of the MOST Cooperation. It may only be disclosed to member companies. Member companies wishing to discuss these Specifications with suppliers or other third parties must ensure that a commercially standard form of non-disclosure agreement has been previously executed by the party receiving such Specifications. Use of these Specifications may only be for purposes for which they are intended by the MOST Cooperation. Unauthorized use or disclosure is a violation of law.

Contents

Bibliography

All documents, which are referenced by this MOST document, are listed here along with their versions.

Document		Revision
[1]	MOST Specification	3.0
[2]	FBlock Diagnosis	3.0
[3]	FBlock NetBlock	3.0
[4]	MOST Physical Layer Basic Specification	1.0

Document History

Evaluation of Diagnostic Information by a Central Component Rev. 1.0

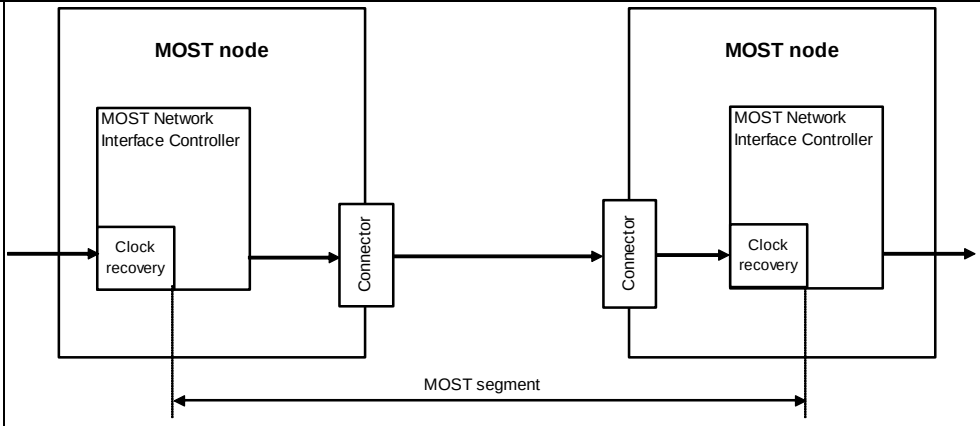
Change Ref.	Section	Changes
1V0_001	General	Initial Version.

1 Introduction

1.1 Scope

This document defines mechanisms for evaluation of diagnostic information, which is stored in all MOST nodes, by a central component in a MOST network [1].

1.2 Definitions

Term	Description
CU	Critical Unlock, according to the MOST Specification [1].
Initial CU	The initial CU that occurred in the MOST system; it has to be localized by the central component.
Detected CU	The CU that is detected locally by a MOST node and the occurrence of which is stored by the node.
Reported CU	A MOST node sends *NetBlock.ShutDownReason.Status* where parameter SSOCUStatus equals “Critical Unlock”.
MOST segment	 One MOST segment starts after the clock recovery [4] of a MOST node and ends after the clock recovery of the next MOST node in direction of the MOST signal.
No fault report	A MOST node sends *NetBlock.ShutDownReason.Status* where parameter SSOCUStatus equals “No fault saved“.
SSO	Sudden Signal Off, according to the MOST Specification [1].
Initial SSO	The initial SSO that occurred in the MOST system; it has to be localized by the central component.
Detected SSO	The SSO that is detected locally by a MOST node and the occurrence of which is stored by the node.
Reported SSO	A MOST node sends *NetBlock.ShutDownReason.Status* where parameter SSOCUStatus equals “Sudden Signal Off”.
System State	State of the MOST network (refer to the MOST Specification [1])

2 Overview

The MOST Specification [1] requires from each MOST node the storage of certain MOST network errors (e.g., Sudden Signal Off and Coding Errors). When checking the MOST network, these locally stored errors are gathered by a central component. To localize the root cause of a MOST network error, these gathered errors are evaluated. After evaluation, the locally stored values are reset to a default value.

Ring Break Diagnosis

Ring Break Diagnosis, which is specified in the MOST Specification [1], can only localize defective MOST segments if Stable Lock cannot be obtained. Ring Break Diagnosis is outside the scope of this document and mentioned solely for reasons of differentiation.

Evaluation of SSO and CU reports

Through the evaluation of reported Sudden Signal Off and Critical Unlock errors (refer to chapter 3), the defective MOST segment can be localized; the successful evaluation of SSO and CU reports requires a Stable Lock and a change from System State NotOK to System State OK.

Coding Error Counter Evaluation

To localize MOST segments with disturbed communication, this document specifies a mechanism for the evaluation of the Coding Error Counter that is implemented in every MOST node; this requires a Stable Lock and a System State OK.

3 Evaluation of Sudden Signal Off and Critical Unlock Reports

The purpose of SSO and CU report evaluation is to locate the origin of those errors; the origin, or in other words the defective MOST segment, is where the “initial SSO” or “initial CU” exists.

To succeed in locating the defective segment, information about the “detected SSO” or “detected CU” is needed. Therefore, individual MOST nodes locally store the occurrence—or rather detection—of SSO and CU errors. This information is retrieved when the MOST nodes answer to requests from the central component that performs the evaluation.

The central component uses the MOST property “NetBlock.ShutDownReason” to collect the information that is stored locally in the nodes of the MOST network about these errors. When a MOST node replies to a ShutDownReason request and reports that an SSO or CU error occurred, this is referred to as the “reported SSO” or “reported CU”.
Through evaluation of the reports, the central component is able to localize the “initial” defective MOST segment.

It is important to distinguish between “initial” and “detected” errors because in terms of availability of the MOST signal, every MOST node only sees its part of the MOST network and therefore the local view of every node, the detected SSO or CU, might not match what the evaluation is focusing on, namely the initial SSO or CU. This differentiation is especially important when more than one node reports an error.
It is further required to distinguish between “detected” and “reported” errors because the detection and the report in many cases do not take place in close succession. The central component has to rely on the reported SSO or CU because it cannot directly access the information about the detected SSO or CU.

3.1 Query ShutDownReasons

In the following state chart, the evaluation by the central component is described.

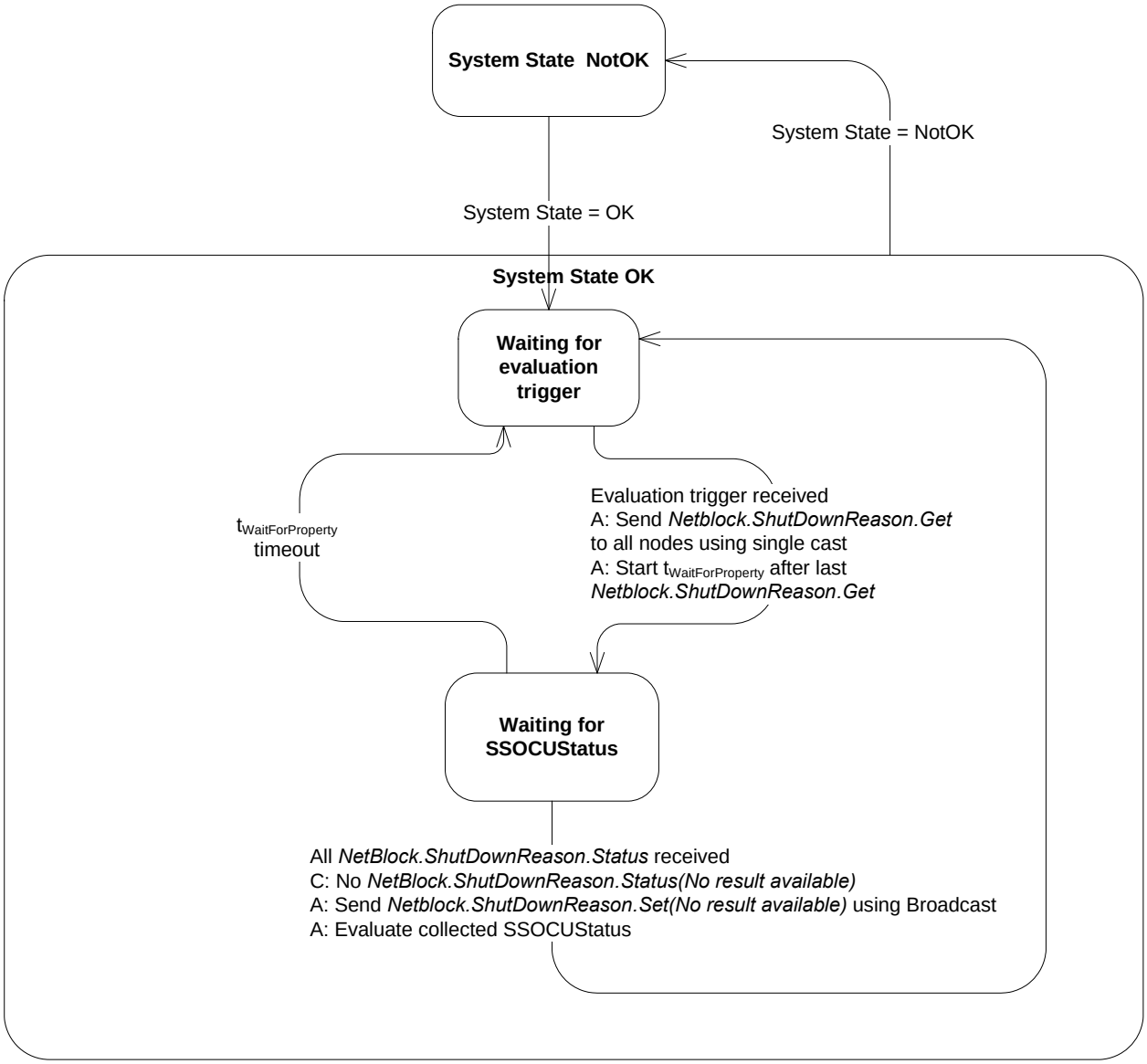


Figure 3-1: SSO CU evaluation state chart

***Notes**:*

- *There is no check that NetBlock.ShutDownReason.Set("No result available") is executed by all nodes correctly. If one node does not set the SSOCUStatus to "No result available", there could be a wrong result during the next evaluation.*
- *The SSOCUStatus evaluation works only with nodes having a DiagID as introduced in [1].*

The trigger to start the SSO/CU report evaluation has to be defined by the System Integrator. The following triggers may be possible:

- Defined time interval after transition to System State OK
- The availability of the MOST system is reported by *NetworkMaster.SystemAvail.Status(…)* (refer to the MOST Specification [1])
- After receiving *NetBlock.ShutDown.StartAck(..., Query)*

Table 3-1 describes the states and their transitions used in Figure 3-1.

State	Description
System State NotOK	This is the initial state where the central component is waiting for the System State to get OK. The transition to "Waiting for evaluation trigger" is taken when the System State changes to OK.
System State OK	This state is the parent state of "Waiting for evaluation trigger" and "Waiting for SSOCUStatus". The state (including its child states) changes to "System State NotOK" if the System State changes to NotOK.
Waiting for evaluation trigger	This state waits for the trigger to start sending the *NetBlock.ShutdownReason.Get* commands. The state changes to "Waiting for SSOCUStatus" if a trigger to start the evaluation is received. The central component sends *NetBlock.ShutDownReason.Get* to every node (including the central component itself) using unicast messages (node position or logical node addressed). In addition, the timer $t_{WaitForProperty}$ is started, which is used to escape from state "Waiting for SSOCUStatus" in case of lost *NetBlock.ShutDownReason.Status* commands or reception of *NetBlock.ShutDownReason.Status("No result available")*.
Waiting for SSOCUStatus	This state waits for the incoming SSOCUStatus reports. The state does not change if *NetBlock.ShutDownReason.Status* is received while there are still outstanding *NetBlock.ShutDownReason.Status* reports. Also, the state does not change if *NetBlock.ShutDownReason.Error* is received. The state changes to "Waiting for evaluation trigger" if the timer $t_{WaitForProperty}$ expires. Also, the state changes to "Waiting for evaluation trigger" if all *NetBlock.ShutDownReason.Status* were received and none of them were *NetBlock.ShutDownReason.Status("No result available")*. The central node sends *NetBlock.ShutDownReason.Set("No result available")* to all nodes using a broadcast message ("unblocking broadcast" recommended) and performs the evaluation of the collected SSOCUStatus (refer to section 3.2). ***Note:*** *The central component has to wait for NetBlock.ShutDownReason.Status reports from all participating nodes in order to avoid reception of NetBlock.ShutDownReason.Status reports from the previous run in the following cycle.*

Table 3-1: SSO CU state description

3.2 Evaluation of ShutDownReasons

This section defines the evaluation of the *NetBlock.ShutDownReason.Status* reports received from the nodes. The evaluation of NetBlock.ShutDownReason.Status reports must differentiate between SSO and CU reports. Criteria which have to be considered by the evaluation are described in this section.

Due to its special role in the MOST system, the TimingMaster might report an SSO or CU that does not reflect the actual situation; therefore, additional differentiations are necessary.
Additional use cases are listed in *APPENDIX A: ShutDown Result Analysis*.

Note: *Initial SSOs or CUs occur during Error Shutdown caused by undervoltage (U_{Low}) or over-temperature. Depending on the definitions of the System Integrator, voltage drops that occur when the engine of the vehicle is started (Error Shutdown caused by U_{Low}) may be excluded from the evaluation of ShutDownReason reports.*

3.2.1 Sudden Signal Off Evaluation

In the evaluation of SSO reports, the following definite statements can be made:

- If a TimingSlave reports an SSO, the initial SSO error is in the MOST segment in front of the TimingSlave.
- If the TimingMaster reported an SSO and no TimingSlave reported an SSO or CU, the initial SSO error is the MOST segment in front of the TimingMaster.

3.2.2 Critical Unlock Evaluation

In the evaluation of CU reports, the following statements can be made:

- Following the direction of the MOST signal, the MOST segment with the initial CU is between the TimingMaster node and the first TimingSlave node that reported the CU.
- Depending on the system, there could be further CU reports because of sequential faults in other MOST segments.

4 Evaluation of Coding Errors

MOST devices have to store the occurrence of Coding Errors (refer to [1]). The MOST property "Diagnosis.CodingErrors", which is specified in the MOST FBlock Library (refer to [2]), can be used to read and write the *CounterValue* in each node. Through evaluation, the defective MOST segments can be localized. In this chapter, the evaluation of Coding Errors by a central component is described.

After an internal request for performing a evaluation of Coding Errors, the central component uses *Diagnosis.CodingErrors.Set(...)* to reset the Coding Error Counter in all nodes and to start the counting.

In the next step, the central component waits for the time t_{count}. The timer t_{count} is defined by the System Integrator. If there is a restart of the MOST network during this phase, the evaluation of Coding Errors has to be aborted and the defective MOST segment has to be localized with the evaluation of Sudden Signal Off and Critical Unlock errors as described in chapter 3.

After expiration of the timer t_{count}, the central component uses *Diagnosis.CodingErrors.Get* to read the value of the Coding Error counter from each node and saves the values for the following evaluation.

Evaluation

During evaluation, the Coding Error *CounterValues* of each node are compared with a threshold which is defined by the System Integrator. The following scenarios are possible:

a) **Coding Error CounterValues of all nodes are below or equal to threshold**
If there was no restart during the test, all MOST segments are okay.

b) **Coding Error CounterValue of only one node is above threshold**
If there was no restart during the test, the MOST segment in front of the node that reported the CounterValue above the threshold is defective.

c) **Coding Error CounterValues of more than one node is above threshold (see Figure 4-1)**
The defect is in the MOST segment in front of the node with the lowest node position that reported a Coding Error *CounterValue* above the threshold. During this evaluation, the central component has to know the position of each node in the MOST network. The TimingMaster is in this consideration the last node in the MOST network.
One defective MOST segment can lead to Coding Error *CounterValues* above the threshold in the following MOST segments, too, if the Coding Error CounterValue of the defective MOST segment is extremely above the threshold or there are Unlocks in the defective MOST segment. This implies that only one defect can be localized during one test run. If there is another defective MOST segment, this can only be located with another test run after repairing the first defect.

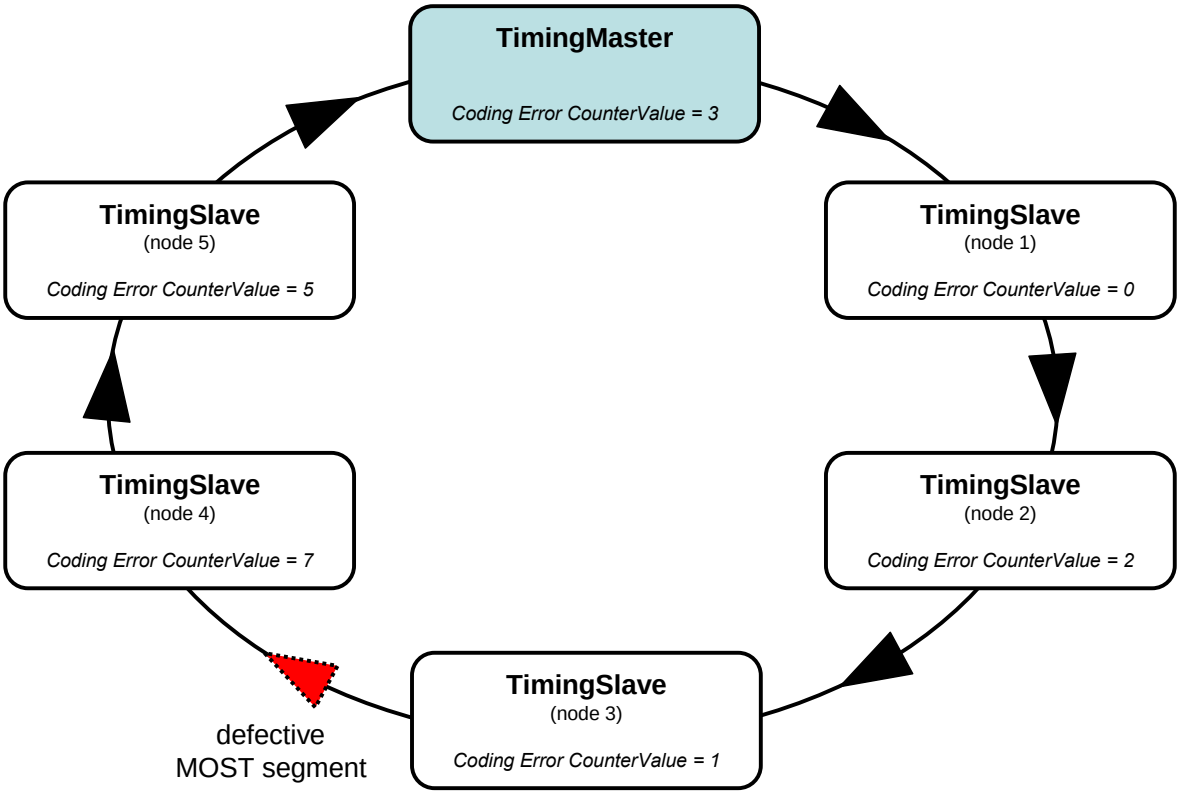


Figure 4-1: Exemplary Coding Error CounterValues

5 APPENDIX A: ShutDown Result Analysis Use Cases

General information about the evaluation of *NetBlock.ShutDownReason.Status* reports is described in section 3.2.

Note: *In the following descriptions, MOST segments are identified in relation to MOST nodes. Those relations are based on the direction of the MOST signal as it travels around the MOST ring.*

If, for example, an error occurs "in front of node B", the error is in the segment that represents the MOST signal input for node B.
If, in another example, an error occurs "between the MOST nodes A and D", it means that the error is located in the MOST segments that transport the MOST signal from the MOST signal output of node A to the MOST signal input of node D.

5.1 Evaluation of SSO Reports

5.1.1 No SSO

NetBlock.ShutDownReason.Status Report	SSO Evaluation Result
No node answers with an SSO report.	No SSO detected.

Table 5-1: Evaluation use case "no SSO error"

5.1.2 Exactly One TimingSlave Reports an SSO

NetBlock.ShutDownReason.Status Report	SSO Evaluation Result
TimingSlave nodes Exactly one TimingSlave node reports an SSO. The other TimingSlave nodes report a CU or "no fault saved". **TimingMaster node** The TimingMaster node reports an SSO, CU, or "no fault saved".	There is an SSO in the MOST segment in front of the TimingSlave that reported the SSO. It cannot be determined if there was a CU in the MOST segments between the TimingSlave that reported the SSO and the TimingMaster. It cannot be determined if there was an SSO in the MOST segment in front of the TimingMaster.

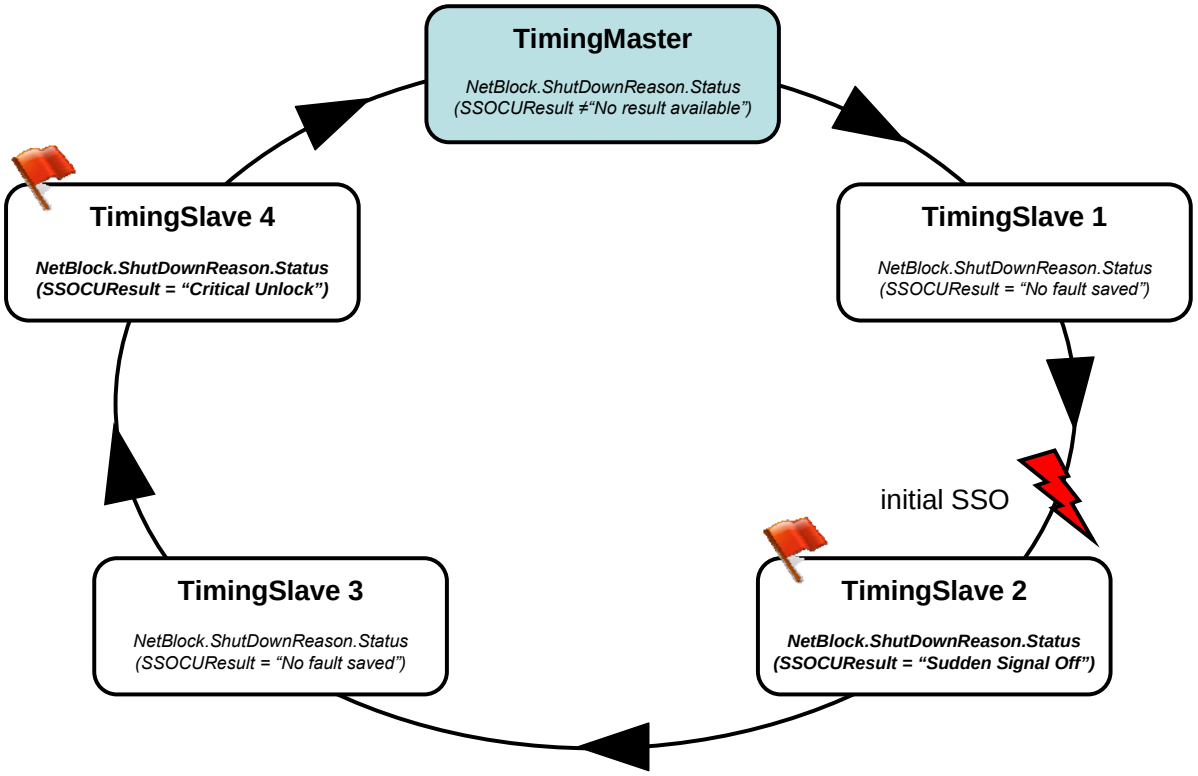


Figure 5-1: Example "TimingSlave node reports SSO"

SSO Evaluation result

- SSO error in MOST segment in front of TimingSlave 2
- No SSO error in MOST segment in front of TimingSlave 1, TimingSlave 3 and TimingSlave 4
- Not determinable if there was an SSO error in the MOST segment in front of the TimingMaster

Note: *additionally, there could be a CU report from a TimingSlave node that is placed between the TimingSlave node which reported the SSO and the TimingMaster node, if*

- *there was an SSO and CU error at the same time*
- *NetBlock.ShutDownReason.Set("No result available") was not executed correctly*

Table 5-2: Evaluation use case "exactly one TimingSlave reports an SSO"

5.1.3 Only the TimingMaster Reports an SSO

NetBlock.ShutDownReason.Status Report	SSO Evaluation Result
TimingSlave nodes All nodes report “no fault saved”. **TimingMaster node** The TimingMaster reports an SSO.	There is an SSO in the MOST segment in front of the TimingMaster.

Table 5-3: Evaluation use case “only TimingMaster reports an SSO”

5.1.4 More than One TimingSlave Node Reports an SSO

NetBlock.ShutDownReason.Status Report	SSO Evaluation Result
TimingSlave nodes More than one TimingSlave node reports an SSO. The other TimingSlave nodes report a CU or "no fault saved". **TimingMaster node** The TimingMaster node reports an SSO, CU or "no fault saved".	There are SSO errors in all MOST segments in front of the TimingSlaves that sent the SSO reports. It cannot be determined if there was a CU error in the MOST segments in front of the TimingSlaves that reported "no fault saved" and that are placed between the TimingSlave with the lowest node position that reported the SSO and the TimingMaster. It cannot be determined if there was an SSO in the MOST segment in front of the TimingMaster.

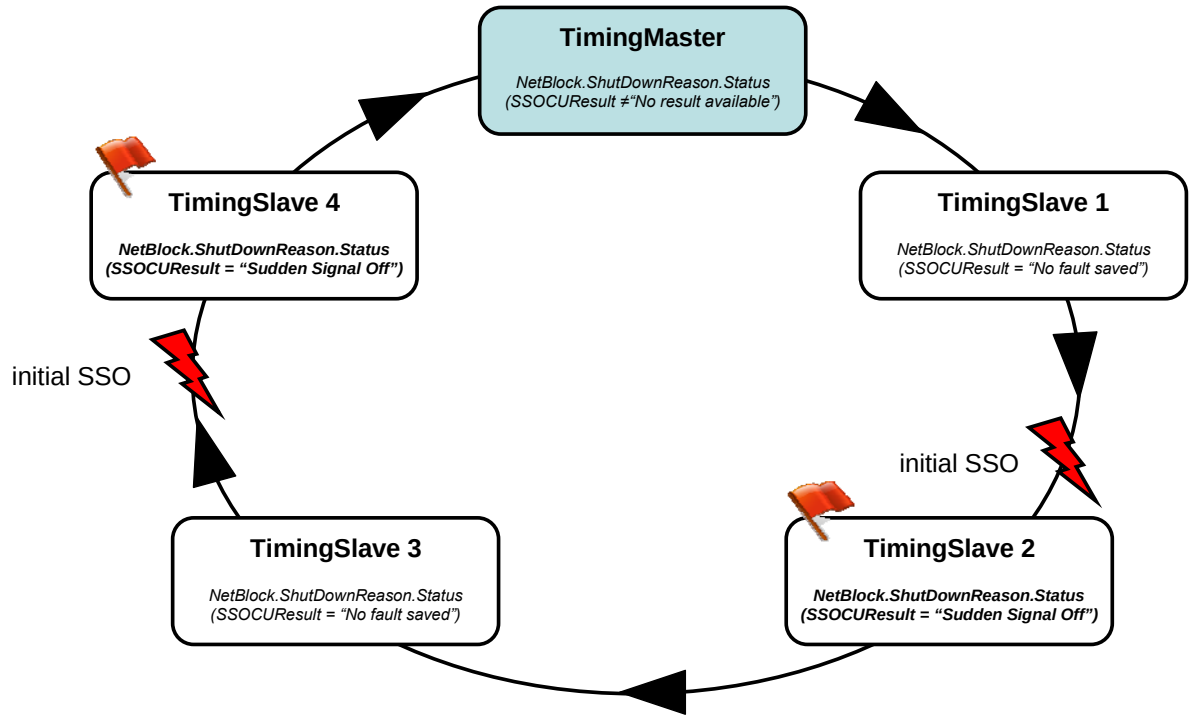


Figure 5-2: Example "more than one TimingSlave node reports SSO"

SSO Evaluation result
- SSO error in MOST segments in front of TimingSlave 2 and TimingSlave 4
- No SSO error in MOST segment in front of TimingSlave 1 and TimingSlave 3
- Not determinable if there is an SSO error in the MOST segment in front of the TimingMaster

***Note:** there could be more than one SSO report by TimingSlave nodes, if*
- *there had been more than one SSO error at the same time*
- *NetBlock.ShutDownReason.Set("No result available") was not executed correctly*

Table 5-4: Evaluation use case "more than one TimingSlave node reports an SSO"

5.2 Evaluation of CU Reports

5.2.1 No CU

NetBlock.ShutDownReason.Status Report	CU Evaluation Result
No node answers with a CU report.	No CU detected.

Table 5-5: Evaluation use case “no CU error”

5.2.2 Exactly One TimingSlave Reports a CU

NetBlock.ShutDownReason.Status Report	CU Evaluation Result
TimingSlave nodes Exactly one TimingSlave node reports a CU. The other TimingSlave nodes report “no fault saved”.	The CU error is in one of the MOST segments between the TimingMaster and the TimingSlave that reported the CU, in direction of the MOST signal.
TimingMaster node The TimingMaster node reports an SSO, CU, or “no fault saved”.	It cannot be determined if there was a CU error in the MOST segments between the TimingSlave that reported the CU and the TimingMaster.

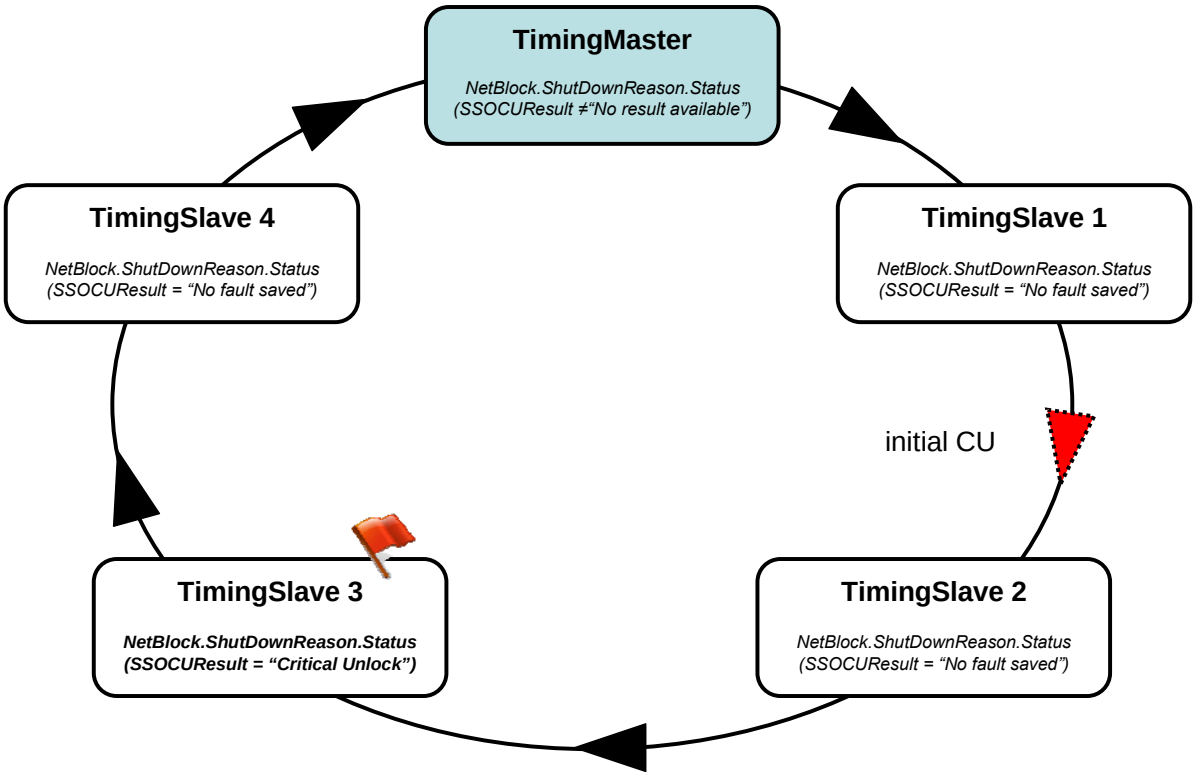


Figure 5-3: Example “exactly one TimingSlave node reports a CU”

CU Evaluation result

- CU error in MOST segment in front of TimingSlave 1 and/or TimingSlave 2 and/or TimingSlave 3
- Not determinable if there are CU errors in the MOST segments in front of TimingSlave 4 and the TimingMaster

Table 5-6: Evaluation use case “exactly one TimingSlave reports a CU”

5.2.3 Only TimingMaster Reports a CU

NetBlock.ShutDownReason.Status Report	CU Evaluation Result
TimingSlave nodes All nodes report “no fault saved”. **TimingMaster node** The TimingMaster reports a CU.	The CU error could be in any MOST segment.

Table 5-7: Evaluation use case “only TimingMaster reports a CU”

5.2.4 More Than One TimingSlave Node Reports a CU

NetBlock.ShutDownReason.Status Report	CU Evaluation Result
TimingSlave nodes More than one TimingSlave node reports a CU. The other TimingSlave nodes report “no fault saved”. **TimingMaster node** The TimingMaster node reports an SSO, CU, or “no fault saved”.	A CU error is in one of the MOST segments between the TimingMaster and the first TimingSlave that reported the CU. It cannot be determined if there was a CU error in the MOST segments between the TimingSlave with the lowest node position that reported the CU and the TimingMaster.

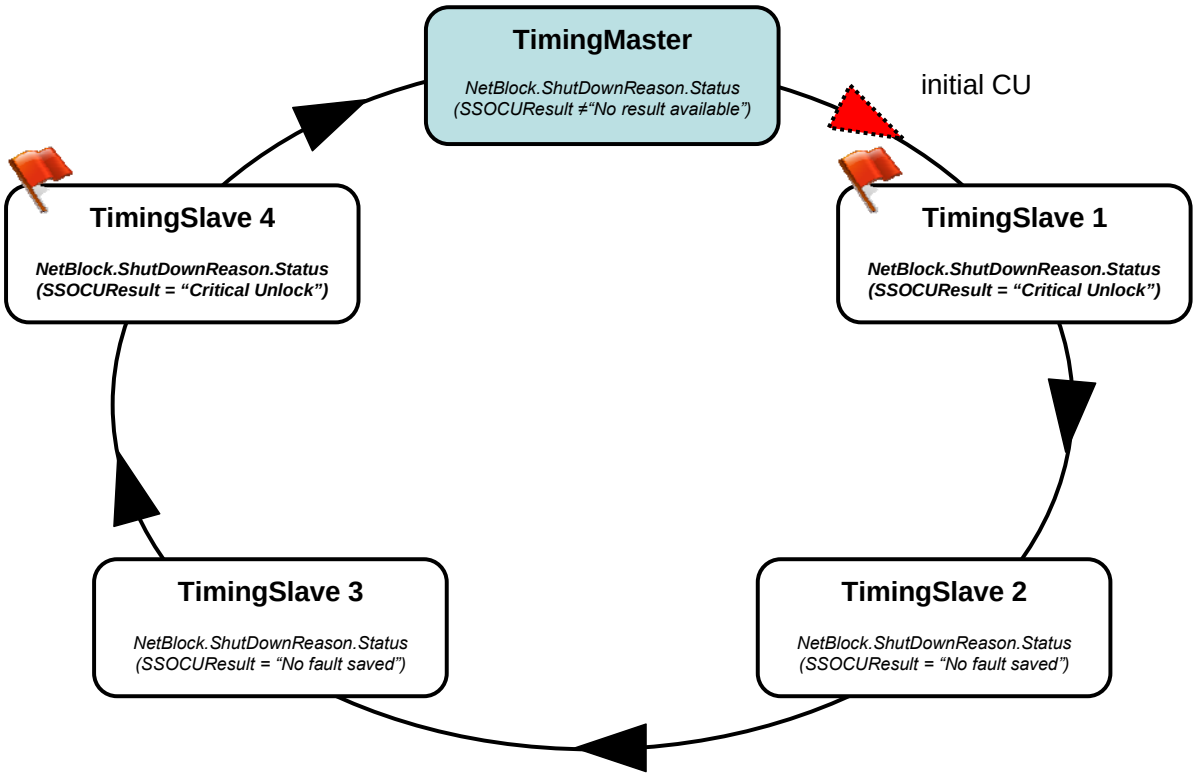


Figure 5-4: Example “more than one TimingSlave node reports a CU”

CU Evaluation result

- CU error in MOST segment in front of TimingSlave 1
- Not determinable if there are CU errors in the MOST segments in front of Timing Slave 2, TimingSlave 3, TimingSlave 4, and the TimingMaster

Table 5-8: Evaluation use case “more than one TimingSlave node reports a CU”

6 Appendix B: List of Figures

7 Appendix C: List of Tables

Notes: