**Heinrich–Theodor Hannen**

# Beitrag zur Analyse sicherer Kommunikationsprotokolle im industriellen Einsatz

Die vorliegende Arbeit wurde vom Fachbereich Elektrotechnik / Informatik der Universität Kassel als Dissertation zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften (Dr.-Ing.) angenommen.

| | |
|---|---|
| Erster Gutachter: | Prof. Dr.-Ing. habil. Josef Börcsök |
| Zweiter Gutachter: | Prof. Dr. habil. Hartmut Hillmer |

Tag der mündlichen Prüfung: 21. August 2012

# Danksagung

Die vorliegende Arbeit entstand unter der wissenschaftlichen Leitung von Herrn Professor Dr.-Ing. habil. Josef Börcsök.

An dieser Stelle danke ich dem Lehrstuhlinhaber Herrn Professor Dr.-Ing. habil. Josef Börcsök für die Möglichkeit zur Durchführung dieser Arbeit herzlich.

Ebenso danke ich Professor Dr. habil. Hartmut Hillmer für seine Bereitschaft, als Gutachter zu wirken.

Ferner danke ich den Mitgliedern der Promotionskommission für die Übernahme der damit verbundenen Arbeiten.

Ich danke weiterhin der Firma HIMA Paul Hildebrandt GmbH + Co KG für Ihre Unterstützung und insbesondere für die Bereitstellung der Rechenleistung zur Berechnung der CRC Gewichte.

Mein Dank gilt allen Kollegen für ihre Diskussionsbereitschaft, sowie allen Korrekturlesern.

Besonderer Dank gilt auch meiner Frau Beate, die für diese Arbeit viele Stunden auf mich verzichten musste, mir immer wieder motivierend zu Seite stand und auch zu den fleißigen Korrekturlesern gehörte.



Das Verhalten der Protokolle PROFIsafe, CIP-Safety, FF-SIF ist den Spezifikationen PROFIsafe Profiles[1], The CIP Networks Library – Volume 5[2] und FF-SIF Protocol Specification[3] entnommen.

[1] [PROFIsafeV1] und [PROFIsafeV2]

[2] [CIP5] und [CIP5-2.2]

[3] [FF-SIF]

# Inhaltsverzeichnis

# 1 Einleitung

In den letzten 15 Jahren hat die sicherheitsgerichtete Prozessdatenkommunikation mehr und mehr Einzug in die Automatisierungs- und Prozesstechnik gehalten. Waren es anfangs nur wenige Anwendungen und Protokolle, die vorwiegend auf Feldbussen zum Einsatz kamen, so ist im Jahre 2012 die Anzahl der Protokolle beträchtlich gewachsen. Alleine die IEC 61784-3[4] listet 7 Protokolle und weitere sind bereits in Vorbereitung. Die Protokolle setzen nun vermehrt auf Ethernet-basierte Übertragungstechniken, wodurch eine Vielzahl verschiedener Technologien für die Übertragung zur Verfügung stehen.

## 1.1 Zielsetzung und Motivation

Bei der Arbeit an sicherheitsgerichteten Kommunikationsprotokollen stellten sich immer wieder die Fragen nach der Eignung von einzelnen Mechanismen der Protokolle, um den Gefährdungen entsprechende Maßnahmen zur Beherrschung entgegenzusetzen.

Beim Protokoll PROFIsafe V1 wurde diese Arbeiten, die Definition des Protokolls, in einer Arbeitsgruppe der Profibus-Nutzerorganisation durchgeführt. Für zwei weitere Protokolle, CIP-Safety und PROFIsafe V2, beinhaltete die Aufgabe neben der Definition auch die Implementierung der Protokolle in sicherheitsgerichtete Steuerungen. Für die Protokolle Fail-Safe over EtherCAT und Foundation-Fieldbus FF-SIF, war nur eine sicherheitstechnische Analyse zu erstellen. Schlussendlich waren 2 sicherheitsgerichtete Protokolle, safe**ethernet** und HSP[5] mit der Definition und der anschließenden Realisierung samt TÜV Abnahme umzusetzen.

Diese Arbeiten forderten immer wieder die Analyse und den sicherheitstechnischen Nachweis der Protokolle zum Einsatz in den jeweiligen industriellen Umgebungen. Dabei waren durchweg die Anforderungen der IEC 61508 mit dem Safety Integrity Level 3 zu erfüllen. Die IEC 61508-2[6] ist mit 5 Zeilen Umfang für die Anforderungen an die sicherheitsgerichtete Kommunikation sehr kurz gehalten und die IEC 61784-3 war zu Beginn der Arbeiten noch nicht geschrieben. Aber die IEC 61784-3 stellt auch heute nur einen unvollständigen Kriterienkatalog zusammen. Insbesondere die in IEC 61508-2[7] geforderte quantitative Analyse wird in der IEC 61784-3 nicht thematisiert und trägt immer wieder zu schwachen Protokolldefinitionen bei.

In dieser Arbeit wird vor diesem Hintergrund ein Bewertungskatalog für Kommunikationsprotokolle zusammengestellt und einige der populärsten Protokolle, wie PROFIsafe, FF-SIF und CIP-Safety, auf ihre Sicherheitsfunktion hin analysiert. Dazu werde eine quantitative Analyse aller Sicherheitsmaßnahmen der Protokolle durchgeführt, was in bisherigen Arbeiten und Normen, außer beim Verfälschungsschutz, nicht geschah.

Um einen vertieften Einblick in die Thematik zu erhalten, wurde im Rahmen dieser Arbeit ein eigenes sicherheitsgerichtetes Kommunikationsprotokoll Transport–Safety–Protocol (TSP) spezifiziert, entworfen und realisiert. Es ist ähnlich effizient wie PROFIsafe, einfacher als CIP-Safety und wird vor allem den Ansprüchen an ein Black-Channel Protokoll für IEC 61508 SIL3 gerecht. Hierbei wurde besonderer Augenmerk auf die Wirksamkeit des Verfälschungsschutzes mittels CRCs für kurze Nachrichten gelegt und auch die Wirksamkeit gekoppelter CRCs betrachtet.

---

4 [IEC 61784-3]

5 Rockwell Automation GuardPLC High-Speed-Serial Protocol

6 [IEC 61508-2] Kapitel 7.4.11.1

7 [IEC 61508-2] Kapitel 7.4.11.2

## 1.2 Aufbau der Arbeit

Die Arbeit beginnt mit einem Überblick zu den Normen IEC 61508, EN 50159-1/2, IEC 13849 und IEC 61784-3, die für die sicherheitsgerichtete Kommunikation in industriellen Anwendungen zu beachten sind. Es folgt eine Betrachtung der gebräuchlichen sicherheitsgerichteten Kommunikationsprotokolle und in Kapitel 2.1 werden die zu berücksichtigen Fehlerarten in Kommunikationseinrichtungen erläutert. Zu diesen Fehlerarten wird jeweils der Bezug zu den Sicherheitsnormen IEC 61508-2 und IEC 61784-3 hergestellt und eine Beschreibung der konkreten Fehlerausprägungen rundet die Darstellung ab.

In Kapitel 2.2 werden weitere Gefahrenpotentiale in Kommunikationseinrichtungen aufgelistet. Diese werden in den oben genannten Normen im Allgemeinen nicht mit Bezug zur sicherheitsgerichteten Kommunikation betrachtet. Diesen Bezug stellt diese Arbeit her und zeigt mögliche Auswirkungen auf die Sicherheit und Verfügbarkeit der Kommunikation.

Im Folgenden werden mögliche Maßnahmen zum Erkennen der Fehlerarten und Vermeidung der Gefahrenpotentiale vorgestellt. Dabei geht die Arbeit auf die quantitative Bewertung der jeweiligen Maßnahmen ein.

Im Anschluss werden Reaktionszeiten von Kommunikationsprotokollen betrachtet. Nach der Definition der Zeiten Safety-Function-Response-Time[8], maximale Reaktionszeit, Worst-Case-1- und Worst-Case-2-Reaktionszeit wird ein Modell entwickelt, dass sich auf eine ganze Menge von Kommunikationsprotokollen, wie z.B. PROFIsafe und TSP anwenden lässt.

An Hand des Modells zeigt diese Arbeit, welche Problemsituationen auftreten können und zu beherrschen sind. Daraus leitet sich in der Reaktionszeitanalyse ab und welche minimale Safety-Function-Response-Time erreichbar ist.

Weiter werden in Kapitel 4 die Protokolle PROFIsafe, CIP-Safety und FF-SIF analysiert. Es werden die Maßnahmen der jeweiligen Protokolle den Fehlerarten und Gefahrenpotentialen gegenüber gestellt und anhand der quantitativen Analyse die Eignung zur Erkennung der Fehlerarten aufgezeigt.

Als ein Resultat ergibt sich für das PROFIsafe Protokoll in Kapitel 4.1, dass die Maßnahmen für die Erkennung von verfälschten Nachrichten nicht ganz den Anforderungen der IEC 61508 gemäß SIL3 entsprechen. Aus der Gegenüberstellung des Modells für die Reaktionszeitberechnung und den Rechnungen von PROFIsafe[9] ergibt sich, dass die Safety-Function-Response-Time der PROFIsafe Spezifikation in einigen Fällen zu gering ist. Eine weitere Sicherheitslücke zeigt sich in der Adressierungstechnik von PROFIsafe, die zu einem unerkannten Versagen der Sicherheitsfunktion führen kann. Dieses Resultat wurde dem PROFIsafe-Arbeitskreis der PROFIBUS-Nutzerorganisation mitgeteilt und in der ATP-Edition[10] veröffentlicht.

Mit CIP-Safety[11] wird in Kapitel 4.2 ein weiteres im Einsatz befindliches sicherheitsgerichtetes Kommunikationsprotokoll der sicherheitsgerichteten Analyse unterzogen. In dieser Arbeit werden dabei die Maßnahmen des Protokolls der Edition 1.1[12] betrachtet. Die dabei ermittelten Schwächen führten zu Verbesserung des Protokolls in seiner Edition 2.2[11].

Als letztes offenes sicherheitsgerichtetes Kommunikationsprotokoll wird FF-SIF[13] der sicherheitstechnischen Analyse unterzogen. Diese zeigt, dass eingefügte Nachrichten nicht in jedem Fall mit der notwendigen Restfehlerwahrscheinlichkeit erkannt werden. Dazu wird eine Lösung aufgezeigt, die dem

[8] SFRT gemäß [IEC61784-3]

[9] [PROFIsafeV2]

[10] [Hann12]

[11] [CIP5-2-2]

[12] [CIP5]

[13] [FF-SIF]

Gremium der Foundation-Fieldbus Organisation zur Verfügung gestellt wurde. Eine weitere Schwachstelle des FF-SIF, die Erkennung von Queuing und Drift von Nachrichten wurde bei der Analyse ebenfalls aufgedeckt. Dazu zeigt die Arbeit ein zweistufiges Lösungskonzept auf, das der Foundation-Fieldbus Organisation ebenfalls zur Verfügung gestellt wurde.

In Kapitel 5 wird ein neues sicherheitsgerichtetes Kommunikationsprotokoll TSP für den Einsatz auf unterschiedlichen Bussystemen spezifiziert, das die Anforderungen gemäß IEC 61508/SIL3 und IEC 13849/CAT4/PL-e erfüllt.

Dabei werden Lösungen aufgezeigt, die die Schwächen von anderen Protokollen, Qualität der Erkennung verfälschter Nachrichten, Adressierungssicherheit und automatischer Anlauf umfassend beherrscht und dennoch eine vergleichbare Effizienz in der Realisierung zulässt.

Die in Kapitel 6 folgende sicherheitstechnische Analyse zeigt, das die Maßnahmen von TSP den Anforderungen der IEC 61508 gemäß SIL3 und IEC 13849/CAT4/PL-e genügen. Bei der Analyse wird insbesondere der Vergleich der Berechnungsmethoden zur Qualität der Bitfehlererkennung für den binär-symmetrischen Übertragungskanal mit der Qualität der konkret eingesetzten CRCs aufgestellt. Für diese wurden die konkreten Gewichte im Rahmen der Arbeit mittels Simulation ermittelt. Insbesondere bei der Kombination zweier CRCs zeigen sich im betrachteten Fall, dass die Rechnung von unabhängigen Wahrscheinlichkeiten gemäß der Approximation (2.9) und konkreten Gewichten, vergleichbare Restfehlerraten ergeben.

Die Ergebnisse der Arbeit werden in Kapitel 7 zusammen gefasst.

Im Ausblick in Kapitel 8 wird auf zukünftige Maßnahmen eingegangen, die dazu beitragen können, sicherheitsgerichtete Kommunikationsprotokolle zu entwickeln, die weniger Schwächen aufweisen. Ebenfalls von aktuellem Interesse ist das Thema Security aus Sicht der sicherheitsgerichteten Kommunikation, wird es heute meist auf den Anwender überantwortet, ohne ihm geeignete Leitlinien zu geben. Aber auch grundlegende Analysen von CRC Sicherungsmaßnahmen sind heute noch nicht mit allgemeinen, praktikablen Berechnungen möglich.

# 2 Stand der Technik

Der Stand der Normen mit Bezug zur sicherheitsgerichteten Kommunikation für die Anwendungsbereiche Fertigungsautomatisierung und Prozessautomatisierung umfasst in Europa im Wesentlichen 3 aktuelle Werke.

Dies ist einerseits die IEC 61508 „Functional safety for electrical/electronic/programmable electronic safety related systems“[14], mit dem relevanten Teil 2. Sie befasst sich mit der Sicherheit elektronischer Systeme allgemein und nimmt dabei auch Bezug auf die sicherheitsgerichtete Kommunikation.

Hervorgegangen ist diese Norm unter anderem aus der vor einiger Zeit abgelösten DIN 0801[15], die bereits Bezüge zur sicherheitsgerichteten Kommunikation aufzeigte.

Eine der ersten Normen für diesen Bereich war die EN 50159-1 & -2 „Railway applications – Communication, signaling and processing systems“[16], wobei sich Teil 1 mit geschlossenen und Teil 2 mit offenen Übertragungssystemen beschäftigt. Diese Norm hat, ebenso wie die Entwicklung des PROFIsafe[17] Protokolls, den Teil 3 der IEC 61784 maßgeblich beeinflusst.

Die IEC 61784-3 „Industrial Process Measurement and Control, Part 3: „Profiles for functional safety communications in industrial networks – General rules and profile definitions“[18], ist Teil der Normenserie IEC 61784 und befasst sich allgemein mit der Kommunikation in industriellen Prozessen.

---

[14] [IEC 61508]

[15] [DIN0801]

[16] [EN50159-1], [EN50159-2]

[17] [PROFIsafeV2]

[18] [IEC 61784-3]

Weitere Bereichsnormen, wie z.B. die IEC 61511[19] oder IEC 13849[20], verweisen bezüglich der sicherheitsgerichteten Kommunikation auf IEC 61508 oder neuerdings auch auf IEC 61784-3.

Heute ist eine vergleichsweise große Zahl von sicherheitsgerichteten Kommunikationsprotokollen im industriellen Einsatz. Waren dies vor 10 Jahren noch einige wenige proprietäre Implementierungen, so zeigt sich nun der Trend hin zu offen zugänglichen Protokollen.

Der Wunsch der Anwender nach standardisierten Protokollen scheint sich bei der Vielzahl an unterschiedlichen Protokollen nicht zu erfüllen, obwohl deren Definitionen im Allgemeinen frei verfügbar sind und den Herstellern zur Implementierung beiseite stehen.

Viele Hersteller oder Herstellervereinigungen haben jedoch versucht, auf den bei ihnen eingesetzten Standardtransportprotokollen aufbauend, ein speziell auf ihre Anwendungen und Geräte abgestimmtes Sicherheitsprotokoll zu entwickeln. Somit setzt sich für die Anwender das Dilemma der unterschiedlichen, inkompatiblen Protokolle der Feldbusebene auch bei den Sicherheitsprotokollen fort.

Nachfolgend wird eine Auswahl an sicherheitsgerichteten Kommunikationsprotokollen vorgestellt. Dabei wird auf Foundation Fieldbus FF-SIF, PROFIsafe und CIP-Safety genauer eingegangen, da sie im Rahmen dieser Arbeit einer eingehenden Sicherheitsanalyse unterzogen werden.

FF-SIF steht für **F**oundation **F**ieldbus **S**afety **I**ntegrated **F**unction. Es wurde innerhalb des Foundation Fieldbus Protokolls FF-H1[21] aufgehängt. FF-SIF ist für die Nutzung in der Prozessautomatisierung entwickelt worden und für das Safety Integrity Level 3 gemäß IEC 61508 ausgelegt. Dabei wird das Modell vom Black-Channel genutzt und ein geschlossenes Übertragungssystem vorausgesetzt.

Die Möglichkeiten der Multi-Cast-Datenübertragung, wie sie beim unterlagerten Transportprotokoll FF-H1 möglich sind, sind für die sicherheitsgerichtete Kommunikation mit FF-SIF nicht verfügbar. Ebenfalls nicht verfügbar ist die Möglichkeit Nachrichten direkt zwischen Geräten auszutauschen, wie dies bei FF-H1 zur Beschleunigung der Kommunikation möglich ist.

PROFIsafe ist ein von der Profibus-Nutzer-Organisation unter Federführung von SIEMENS, Deutschland entwickeltes sicherheitsgerichtetes Kommunikationsprotokoll. PROFIsafe ist für die Anwendung in der Fertigungs- und Prozessautomatisierung ausgelegt und verwendet die unterlagerten Transportprotokolle PROFIBUS-DP und PROFINET-IO. Das Protokoll wurde für den Safety Integrity Level 3 gemäß IEC 61508 und PLe/CAT4 gemäß EN 13849[22] entworfen.

Entwickelt wurde PROFIsafe in einer Version 1.30, die ursprünglich nur den Einsatz über PROFIBUS-DP vorsah. Nach der Einführung von PROFINET-IO sollte diese auf Ethernet basierende Technik ebenfalls genutzt werden. Dafür musste PROFIsafe auf die Version 2 (V2-Mode) erweitert werden, um den durch die Ethernet-Netzwerke erhöhten Gefährdungen Rechnung zu tragen. So wurde der Monotoniezähler von 8-Bit auf 24-Bit erhöht und mit in die CRC Rechnung einbezogen. Weiterhin wurde der CRC für 12 Bytes Nutzdatennachrichten von 16-Bit auf 24-Bit erhöht, um die funktionalen Sicherheitsanforderungen zu erfüllen.

Heute werden Geräte meist nur noch mit dem V2-Mode von PROFIsafe eingesetzt, auch wenn sie sich am PROFIBUS-DP befinden.

CIP-Safety[23] wurde von der ODVA[24] unter Federführung von Rockwell Automation, USA entwickelt. CIP-Safety ist für die Anwendung in der Fertigungs- und Prozessautomatisierung ausgelegt und verwendet die unterlagerten Transportprotokolle EtherNet/IP und DeviceNet. Das Protokoll wurde für

[19] [IEC 61511]

[20] [IEC 13849]

[21] [FFSA]

[22] [EN 13849]

[23] [CIP5]

[24] Open DeviceNet Vendors Association //www.odva.org

den Safety Integrity Level 3 gemäß IEC 61508 entworfen. Dabei wird von einem Black-Channel Modell ausgegangen.

CIP-Safety wurde anfangs nur für den Einsatz über DeviceNet entwickelt. Ebenso wie PROFIsafe musste es für den Einsatz in Ethernet-Netzwerken sicherheitstechnisch weiter ertüchtigt werden. Dazu wurden die verwendeten CRCs und die Mechanismen zur Aufdeckung von Nachrichteneinfügungen erweitert.

Es werden ebenfalls zahlreiche andere sicherheitsgerichtete Protokolle industriell eingesetzt. Dazu folgend ein kleiner Überblick.

- AS-i Safety[25] nutzt AS-Interface zum Transport. Die Zertifizierung erfolgte gemäß EN 954-1/CAT4[26]. Anwendungsbereiche sind die Fertigungsautomatisierung und die Prozessautomatisierung, aber auch in der Gebäudeanwendungen findet man AS-i Safety.
- CANopen Safety[27] über den Feldbus CANopen ist ein von der CIA[28], Deutschland entwickeltes Protokoll für verschiedene Einsatzgebiete, u.a. für Transportsysteme und für die Fertigungsautomatisierung. Die Schutzmaßnahmen genügen SIL3 gemäß IEC 61508 und EN 954-1/CAT4[26] und beziehen eine spezielle sicherheitsgerichtete Hardware, den CANopen Safety Chip, mit ein[29].
- CC-Link Safety[30] wurde von der CC-Link Partner Association, unter Federführung von Mitsubishi Electric Automation, Japan, entwickelt. Die Schutzmaßnahmen genügen SIL3 gemäß IEC 61508 und EN 13849/PLe/CAT4[31]. Als Transportprotokoll wird der Bus CC-Link verwendet. Das Haupteinsatzgebiet von CC-Link-Safety ist die Fertigungsautomatisierung.
- INTERBUS-Safe[32] über den Feldbus Interbus, wurde vom INTERBUS-Club unter Federführung von PHOENIX CONTACT, Deutschland entwickelt. Hauptanwendungsbereich ist die Fertigungsautomatisierung. Die Schutzmaßnahmen genügen SIL3 gemäß IEC 61508.
- POWERLINK-Safety[33] über Ethernet-Powerlink und CAN, wurde von Bernecker + Rainer, Österreich und der Universität Winterthur, Schweiz entwickelt. Das Protokoll wird für die Maschinen- und Fertigungsautomatisierung eingesetzt und zeichnet sich durch seine Echtzeitfähigkeit aus. Die Schutzmaßnahmen genügen SIL3 gemäß IEC 61508.
- safe**ethernet** nutzt Standard Ethernet[34] und UDP/IP zum Transport und wurde von HIMA Paul Hildebrand GmbH + Co KG, Deutschland, entwickelt. Die Schutzmaßnahmen genügen SIL3 gemäß IEC 61508 und EN 13849/PLe/CAT4[31]. Das Protokoll ist für den allgemeinen Einsatz in der Fertigungsautomatisierung vorgesehenen. In seiner redundanten Ausführung ist es auch in der Prozessautomatisierung im Einsatz. Es wird jedoch auch in außergewöhnlichen Bereichen, wie der Pipeline-, Verkehrs- und Schiffsautomatisierung eingesetzt und kann dabei nahezu beliebige Netzwerkinfrastrukturen nutzen.

---

25 [ASIs09]

26 [EN 954-1]

27 [CANs09]

28 CAN in Automation GmbH

29 [CANt04, CANc04]

30 [CCLS]

31 [EN 13849]

32 [IBS]

33 [EPLS]

34 [IEEE 802.3]

- Safety over EtherCAT[35] wurde von Beckhoff, Deutschland entwickelt und nutzt das Transportprotokoll EtherCAT. Das vorwiegende Einsatzgebiet ist die Fertigungsautomatisierung. Die Schutzmaßnahmen genügen SIL3 gemäß IEC 61508.
- SafetyBUS p[36] über ein CAN Netzwerk, wurde von SafetyBUS p Club International e.B. unter Federführung der Firma Pilz GmbH & Co KG, Deutschland, entwickelt. Hauptanwendungsbereich ist die Fertigungsautomatisierung. Die Schutzmaßnahmen genügen SIL3 gemäß IEC 61508. Als Weiterentwicklung steht seit einiger Zeit auch SafetyNET p über Ethernet zur Verfügung.

## 2.1 Fehler in Kommunikationseinrichtungen

Neben der eigentlichen Aufgabe von Kommunikationsprotokollen für den sicherheitsgerichteten Einsatz, die Übertragung von Daten, gehört die Erkennung und Beherrschung von Fehlern.

Gemäß IEC 61508 werden dazu an das Kommunikationsprotokoll folgende Anforderungen gestellt.

„When data communication is used in the implementation of a safety function then the failure measure (such as the residual error rate) of the communication process shall be estimated taking into account transmission errors, repetitions, deletion, insertion, re-sequencing, corruption, delay and masquerade.“[37]

Diese und weitere Anforderungen werden im Folgenden im Detail betrachtet.

„Werden Daten für sicherheitsgerichtete Funktionen mittels Kommunikationssystemen übertragen, so müssen Verfälschungen der Daten bei der Übertragung in Betracht gezogen werden.“[38]

Durch Fehler im Übertragungssystem oder durch äußere Einwirkung können übertragene Informationen innerhalb des Übertragungssystems verfälscht werden.

Zu betrachten sind einerseits systematische Fehler der Komponenten (Design- und Implementierungsfehler), wie z.B. unzureichende elektrische Verbindungen, ungeeignete optische oder kapazitive Dämpfung für die eingesetzte Treibertechnologie, minderwertige Steckverbinder, unzureichende Schirmung oder minderwertige Kabel[39].

Andererseits sind auch zufällige Fehler, wie z.B. defekte HW-Komponenten, elektrostatische Entladungen, elektromagnetische Störungen, Probleme durch die Stromversorgung oder Kabelvibrationen, zu betrachten[39].

Verfälschungen können sich durch verschiedene Variationen, sowie einer Mischung davon, darstellen.

1. Einzelne Bits oder eine Folge von Bits innerhalb einer Nachricht wurden invertiert.
2. Die Nachricht wurde hinten und/oder vorne verkürzt.
3. Ein Nachrichtenteil wurden, auf Grund fehlerhafter Fragmentierung, bezüglich der Reihenfolge von Bit- und Byte-Sequenzen innerhalb der Nachricht vertauscht.

„Werden Daten für sicherheitsgerichtete Funktionen mittels Kommunikationssystemen übertragen, so müssen unbeabsichtigte Wiederholungen von Nachrichten in Betracht gezogen werden.“[38]

Unbeabsichtigte Wiederholungen von Nachrichten können durch Fehlfunktionen der nicht sicherheitsgerichteten Komponenten des Übertragungssystems, die eine Nachricht speichern können, verursacht werden.

---

35 [FSoE]

36 [SBp]

37 [IEC 61508-2] Kapitel 7.4.11.1

38 [IEC 61508-2] Kapitel 7.4.11.1, [IEC 61784-3] Kapitel 5.3.2

39 [Boer07]

Komponenten, die Nachrichten speichern können, kommen in sehr vielen Anwendungsfällen vor, da die Hardwareeinheiten (Prozessor, Speicher, Kommunikations-Controller) oder zumindest die SW-Komponenten zur Ansteuerung des Netzwerks im Allgemeinen nicht sicherheitsgerichtet ausgeführt sind.

Bei üblichen Anwendungen der hier betrachteten Kommunikationsprotokolle müssen zusätzlich zu den Netzwerk-Anschaltungen auch Komponenten, wie Switches, Bridges, Router und Gateways berücksichtigt werden, die ebenfalls die Fähigkeit besitzen, Nachrichten zu speichern. Dies trifft insbesondere bei den zunehmend eingesetzten Ethernet-Übertragungssystemen zu. Aber auch schon bei PROFIBUS-DP gehören so genannte Linking-Devices (Übergang von PROFIBUS-DP nach PROFIBUS-PA) zum Bereich der Anwendungsmöglichkeiten.

Bei einigen der für sicherheitsgerichtete Protokolle genutzten Übertragungssysteme werden Wiederholungen jedoch gezielt eingesetzt, um die Verfügbarkeit der Übertragung zu erhöhen.

Als Beispiel kann hier PROFIsafe dienen, das die Übertragungstechniken PROFINET-IO und PROFIBUS-DP nutzt. Bei PROFINET-IO und PROFIBUS-DP werden Nachrichten(-kopien) periodisch versendet. Da der Zyklus der Bearbeitung von PROFIsafe gegenüber dieser Periode im Allgemeinen deutlich größer ist, werden bei PROFIsafe Nachrichten standardmäßig mehrfach übertragen.

Ähnliche Mechanismen findet man auch bei CIP-Safety, das die EtherNet/IP Publisher/Subscriber Mechanismen nutzt, die ebenfalls eine periodische Übertragungstechnik einsetzen.

Ein Sonderfall der Mehrfachübertragung kommt auch bei redundanten Transportschichten zum Einsatz, wie z.B. beim safe**ethernet** Protokoll[40]. Jede Nachricht wird dabei zur Steigerung der Verfügbarkeit doppelt übertragen.

„Werden Daten für sicherheitsgerichtete Funktionen mittels Kommunikationssystemen übertragen, so müssen Reihenfolgevertauschung von Nachrichten in Betracht gezogen werden.“[38]

Den puffernden[41] nicht sicherheitsgerichteten Komponenten eines Übertragungssystems muss unterstellt werden, dass diese die Reihenfolge der von einer sicherheitsgerichteten Komponente versendeten Nachrichten vertauschen könnten. Hierbei ist nicht nur das Fehlerverhalten der puffernden Komponenten zu betrachten, sondern insbesondere auch die reguläre, d.h. die von den Komponenten erwartete Funktion, da diese eine sehr viel höhere Eintrittswahrscheinlichkeit als das fehlerhafte Verhalten aufweist. Als Beispiel seien hier Router mit dynamischer, Kosten basierter Wegwahl genannt, wodurch Nachrichten beim Empfänger in anderer Reihenfolge ankommen können, als sie vom Absender versendet wurden.

„Bei der Datenübertragung für sicherheitsgerichtete Funktionen mittels Kommunikationssystemen müssen Einfügungen von Nachrichten als Fehler betrachtet werden.“[42]

Den puffernden, nicht sicherheitsgerichteten Komponenten eines Übertragungssystems muss unterstellt werden, dass sie eine „korrekte“ Nachricht speichern und zu einem späteren Zeitpunkt an die sicherheitsgerichtete Einheit schicken. Dies ist zum Beispiel auch eine Annahme in der IEC 61784-3-3, die dass fehlerhafte Speicherverhalten von Ethernet-Switches unterstellt.

Da die verwendeten, nicht sicherheitsgerichteten Komponenten heute oft sehr leistungsfähig sind und daher über eine umfangreiche Pufferfähigkeit von Nachrichten verfügen, muss zusätzlich betrachtet werden, dass eine Folge von (aufgezeichneten) Nachrichten durch eine solche fehlerhafte Komponente nach der Aufzeichnung in genau der aufgezeichneten Reihenfolge wieder versendet werden könnten[43].

---

[40] [safeethernet]

[41] Puffernde Komponenten sind nicht nur solche, die den Puffer für die reguläre Übertragung von Nachrichten einsetzen, sondern auch solche, die dies fehlerhaft tun könnten.

[42] [IEC 61508-2] Kapitel 7.4.11.1, [IEC 61784-3] Kapitel 5.3.2

[43] [IEC 61784-3-3]

Selbst „kleine“ 5-Port-Switche haben 128kByte Puffer für Nachrichten. Bei größeren Switches, Routern und Gateways ist die Nachrichtenspeicherkapazität in der Größenordnung von einigen MBytes und mehr.

Das Szenario der unerwünschten Wiederholungen von Nachrichtensequenzen ist besonders interessant, da dadurch auch die Anlaufmechanismen der Protokolle wiederholt werden können. Es ist daher zu unterstellen, dass eine solche puffernde Komponente eine „korrekte“ Anlaufnachrichtensequenz an eine sicherheitsgerichtete Einheit schickt.

„Werden Daten für sicherheitsgerichtete Funktionen mittels Kommunikationssystemen übertragen, so ist der Verlust von Nachrichten in Betracht zu ziehen.“[42]

Viele der nicht sicherheitsgerichteten Übertragungskomponenten beinhalten heute eigene Mechanismen zur Erkennung von Nachrichtenverfälschungen. Erkennen sie eine solche Verfälschung, verwerfen sie i.A. diese Nachricht und es kommt aus Sicht der sicherheitsgerichteten Komponente zu einem Nachrichtenverlust.

„Werden Daten für sicherheitsgerichtete Funktionen mittels Kommunikationssystemen übertragen, so müssen inakzeptable Verzögerungen von Nachrichten in Betracht gezogen werden.“[42]

Die Verzögerung von Nachrichten kann viele Ursachen haben. Allgemein ist, dass puffernde Komponenten die Nachrichten zwischenspeichern und mit einem mehr oder weniger großen Zeitverzug, weitersenden.

Häufig ist die Ursache für die zusätzliche oder unbeabsichtigte Verzögerung eine Überlastung des Kommunikationssystems an dieser Stelle. Dies stellt daher im eigentlichen Sinne keinen Fehler dar, sondern gehört zur „normalen“ Funktion des Übertragungssystems.

Eine weitere Ursache für die Verzögerung von Nachrichten kann eine gestörte Übertragungsstrecke sein, für die die Komponenten des Übertragungssystems versuchen, mittels Wiederholungen von Nachrichten, diese dennoch auszuliefern. Dieses Verfahren wird z.B. bei IEEE 802.11 (WiFi) oder der CSMA/CD[44]-Technik angewendet.

Schließlich kann auch eine fehlerhaft agierende Komponente eine Ursache für Verzögerungen von Nachrichtenübertragungen sein.

Insbesondere ist zu beachten, dass die Verzögerung auf einem Nachrichtenpfad nicht konstant sein muss, sich also mit der Zeit ändern kann. Ebenso ist zu berücksichtigen, dass die Verzögerung auf dem Nachrichtenpfad für Hin- und Rückweg unterschiedlich sein kann.

Diese asymmetrischen und zeitlich veränderlichen Verzögerungen werden nicht nur durch fehlerhafte Komponenten verursacht. Werden Router mit mehreren alternativen Wegen eingesetzt, so kann die dynamische Wegwahl des Routers die Verzögerung verändern. Ebenso müssen Hin- und Rückweg nicht über die gleichen Strecken gehen oder gar das gleiche physikalische Übertragungssystem nutzen, so dass auch hierbei unterschiedliche Verzögerungen auftreten können.

Schließlich sei noch die Auslastung des Übertragungssystems als Ursache für Verzögerungen genannt. Da nicht immer von einem gleichmäßigen Nachrichtenaufkommen auszugehen ist, kommt es durch die Puffermechanismen und die möglicherweise Bandbreiten limitierten Übertragungsstrecken, zu sich dynamisch ändernden Verzögerungen. Dies ist insbesondere dann zu berücksichtigen, wenn das Übertragungssystem erweitert oder für zusätzliche Aufgaben genutzt wird, die zum Zeitpunkt der Installation des Sicherheitssystems noch nicht bekannt waren oder beachtet wurden.

„Werden Daten für sicherheitsgerichtete Funktionen mittels Kommunikationssystemen übertragen, so muss ebenfalls die falsche Adressierung von Nachrichten berücksichtigt werden.“[45]

---

[44] Carrier Sense Multiple Access / Collision Detection

[45] [IEC 61784-3] Kapitel 5.3.2

Bei einer sicherheitsgerichteten Übertragung von Nachrichten ist sicher zu stellen, dass eine Nachricht vom richtigen Absender kommt und beim richtigen Empfänger ankommt.

Dabei ist den nicht sicherheitsgerichteten Komponenten des Übertragungssystems zu unterstellen, dass sie in fehlerhafter Weise eine Nachricht an den falschen Empfänger ausliefern, wie auch der Fall, dass durch falsche Konfiguration der nicht sicherheitsgerichteten Übertragungssysteme, Nachrichten zum falschen Empfänger geleitet werden.

Zusätzlich zum nachfolgend beschriebenen Masquerading deckt der Aspekt Adressierung auch den Fall ab, dass eine von einer sicheren Einheit generierte Nachricht, bei der falschen, sicheren Empfangseinheit ankommt.

„Werden Daten für sicherheitsgerichtete Funktionen mittels Kommunikationssystemen übertragen, so muss das Masquerading von Nachrichten betrachtet werden.“[46]

Unter Masquerading versteht man, dass Nachrichten, insbesondere auch nicht sicherheitsgerichtete, von nicht sicherheitsgerichteten Komponenten unberechtigter Weise an eine sicherheitsgerichtete Komponente versendet werden. Ziel ist es die Authentizität des Absenders einer Nachricht zu gewährleisten. Damit deckt Masquerading auch einen Teil des Aspekts Adressierung ab.

Beim Masquerading kommt noch hinzu, dass nicht sicherheitsgerichteten Komponenten im Übertragungssystem unterstellt werden muss, dass sie zufällig sicherheitsgerichtete Nachrichten generieren.

Bei geschlossenen Übertragungssystemen wird jedoch ausgeschlossen, dass die nicht sicherheitsgerichteten Komponenten absichtlich sicherheitsgerichtete Nachrichten generieren.

## 2.2 Gefahren in Kommunikationseinrichtungen

Neben den Fehlern, die in Kommunikationseinrichtungen auftreten können, müssen sicherheitsgerichtete Kommunikationsprotokolle auch die Gefahren, die sich durch die Umgebung und den Einsatz ergeben, beherrschen.

Diese Aspekte werden in den Sicherheitsnormen IEC 61508 und IEC 61784-3 für die Kommunikation nicht gesondert erwähnt, da diese Normen sich im Rahmen der Kommunikation vornehmlich mit zufälligen Hardware-Fehlern befassen.

Ableiten lässt sich diese Gefährdung jedoch aus der IEC 61508-1[47], wonach „`malevolent or unauthorized action`“, sowie der „`foreseeable misuse`“ in der Gefährdungsbetrachtung eines sicherheitsgerichteten Systems zu berücksichtigen sind.

Dem Thema muss in den heute gängigen Anwendungsbereichen jedoch sehr wohl Aufmerksamkeit geschenkt werden, da die üblichen Einschränkungen der Sicherheitsprotokolle auf „geschlossene Übertragungssysteme“, das heißt solche, die diese Gefährdung ausschließen, nur bedingt mit dem Einsatzgebieten vereinbar sind.

Besonders beim Einsatz Ethernet basierter Übertragungssysteme sind umfangreiche Gefährdungspotentiale zu betrachten. Hierbei sind nahezu immer auch ein oder mehrere PCs samt PC-Betriebssystem im Netzwerk vorhanden. Diese der Steuerungsebene nahen PCs sind weiterhin mit Rechnern der Produktionsplanung und -steuerung verbunden und sind somit bis in die Unternehmens-IT integriert.

Für die Steuerungsebene sind die heute üblichen Möglichkeiten der Fernwartung ebenfalls relevant, sowie temporär anwesendes Wartungspersonal des Herstellers oder Betreibers mit mobilen PCs, die Zugriff auch auf sicherheitsgerichtete Komponenten erhalten.

Zu betrachten sind die Bedienungsmöglichkeiten, die dazu führen könnten, dass die Mechanismen der sicherheitsgerichteten Protokolle von Personen ohne Berechtigung sabotiert werden.

---

[46] [IEC 61508-2] Kapitel 7.4.11.1, [IEC 61784-3] Kapitel 5.3.2

[47] [IEC 61508-1] Kapitel 7.4.2.2

Erwähnung findet das Thema unberechtigter Zugriff in EN 50159-2[48], sowie in IEC 61784-3-3, bei den dort erwähnten Funk-Übertragungssystemen WiFi und Bluetooth. Jedoch wird dabei nur Bezug auf den Schutz des Transports der sicherheitsgerichteten Nachrichten genommen, der Zugriff auf die sicherheitsgerichteten Knoten und deren Konfiguration wird nicht betrachtet.

Durch unberechtigten Zugriff auf eine sichere Komponente kann eine falsche Konfiguration der Kommunikationsbeziehungen erfolgen. Daraus entsteht die Gefahr, dass die sicherheitsgerichtete Funktion nicht erfüllt werden kann.

Durch einen unberechtigten Zugriff kann zum Beispiel die Überwachungszeit der Kommunikationsverbindung zu hoch eingestellt werden. Im Falle einer Verbindungsunterbrechung kann dies zu einer verspäteten Sicherheitsreaktion führen.

Der unberechtigte Zugriff kann andererseits auch die eigentliche Funktion, die Datenübertragung, verhindern. Dies führt zwar in der Regel zur sicherheitsgerichteten Reaktion, jedoch ist die dann nicht vorhandene Verfügbarkeit für den sicheren Betrieb gefährlich, da in diesem Fall mit der Umgehung der sicherheitsgerichteten Einrichtungen durch den Anwender zu rechnen ist.

In offenen Übertragungssystemen ist bezüglich Masquerading davon auszugehen, dass nicht nur bekannte andere Protokolle, sondern auch beliebige, insbesondere zum Zeitpunkt der Installation, beziehungsweise bei späteren Erweiterungen, unbekannte Protokolle im gleichen Übertragungssystem eingesetzt werden.

In offenen Übertragungssystemen ist bezüglich der sicheren Adressierung davon auszugehen, dass das selbe Sicherheitsprotokoll in einem anderen Bereich unwissentlich zum Einsatz kommt, beziehungsweise in Zukunft zum Einsatz kommen wird. In diesem Fall ist es wahrscheinlich, dass die Adressierung der sicheren Komponenten der verschiedenen Bereiche nicht eineindeutig ist.

Wurde in den vorangegangenen Betrachtungen die zufällige Gefährdung durch Standard-Protokolle betrachtet, so ist bei der absichtlichen Unterminierung der unerlaubte Einsatz von Stationen zu berücksichtigen, die ebenfalls eine Implementierung des Sicherheitsprotokolls aufweisen.

Mit diesen Stationen kann versucht werden, die Sicherheitsfunktion im Netzwerk außer Kraft zu setzen oder aber den Betrieb zu unterbinden, so dass es zu einer Gefährdung der Sicherheit kommt.

Zum Beispiel kann durch Einschleusen von Sicherheitsnachrichten mit korrupten CRCs bei einigen Sicherheitsprotokollen die Kommunikation unterbrochen und nachfolgend komplett verhindert werden. Ebenso besteht die Möglichkeit, dass eine solche Station die Rolle einer sicherheitsgerichteten Komponente übernimmt, indem sie nach dem Prinzip „man in the middle“ deren Nachrichten unterdrückt und durch eigene ersetzt.

Sicherheitsgerichtete Kommunikationsverbindungen müssen im Allgemeinen vom Betreiber konfiguriert werden. Zu den Parametern gehören, neben den Daten und der Adressierung, immer auch die Überwachungszeit des Protokolls.

Während Daten und Adressierung in der Regel durch den Betreiber verifiziert werden können, ist der Parameter Überwachungszeit und die daraus resultierende Worst-Case-Reaction-Time nicht innerhalb der installierten Anlage prüfbar.

Der Grund dafür liegt in den zu simulierenden Fehlerszenarien begründet. Um die Worst-Case-Reaction-Time zu verifizieren, müssen in der Regel mehrere Fehler zu präzisen Zeitpunkten simuliert werden. Unterbricht man beispielsweise die Netzwerkverbindung von einer Sicherheitssteuerung, an den eine Eingangskomponente und eine Ausgangskomponente angeschlossen sind, so wird, gleiche Überwachungszeiten für Ein- und Ausgangskomponente vorausgesetzt, immer die Überwachungseinrichtung der Ausgangskomponente die Sicherheitsreaktion auslösen. Für die Worst-Case-Betrachtung muss aber erst die Verbindung zur Eingangskomponente getrennt und kurz vor Ablaufen der Überwa-

[48] [EN 50159] Anhang A.2

chungszeit zur Eingangskomponente, die Verbindung zur Ausgangskomponente getrennt werden. In diesem Fall wird man eine doppelt so lange Reaktionszeit messen.

Ein weiterer Aspekt fehlerhafter Konfiguration ist der Fall, dass die Konfiguration zweier kommunizierender sicherheitsgerichteter Komponenten nicht zueinander passt. Dies ist beispielsweise der Fall, wenn trotz korrekter Länge das Datenlayout unterschiedlich ist. Das führt unweigerlich zu einer falschen Interpretation der sicheren Daten.

Weitere Möglichkeiten der fehlerhaften Konfiguration hängen vom konkreten sicherheitsgerichteten Kommunikationsprotokoll und dessen Engineering-Werkzeug ab.

Gemäß den allgemeinen Anforderungen aus IEC 61508-1[49], IEC 61508-4[50], EN 13849-1[51] sind in der Risikoanalyse Szenarien des absehbaren Missbrauchs zu betrachten. Übertragen auf sicherheitsgerichtete Kommunikationsanwendungen bedeutet dies insbesondere die absichtliche Einstellung zu hoher Überwachungszeiten.

Die zu unterstellende Motivation des Anwenders dazu liegt in der damit verbundenen Erhöhung der Verfügbarkeit und damit auch seiner Produktivität begründet.

Ein weiteres zu betrachtendes Szenario ist der Einsatz von Standardprotokollen, anstatt von sicherheitsgerichteten Protokollen. Dies muss insbesondere dann angenommen werden, wenn die sicherheitsgerichteten Komponenten auch solche Standardprotokolle unterstützen.

Weitere Möglichkeiten des Missbrauchs hängen vom konkreten sicherheitsgerichteten Kommunikationsprotokoll und dessen Engineering-Werkzeug ab.

In IEC 61508-1[49], IEC 61508-4[50], EN 13849-1[52] ebenfalls gefordert, ist Szenarien der absehbaren Fehlbedienung in der Risikoanalyse aufzunehmen. Übertragen auf sicherheitsgerichtete Kommunikationsanwendungen bedeutet dies insbesondere die versehentliche Einstellung falscher Adressen. Dabei ist besonders die Verwechslung von Adressen zu berücksichtigen.

Da in Anlagen häufig viele baugleiche Geräte eingesetzt werden, kann bei vom Anwender vertauschten Adressen, ein Protokoll die Adressvertauschung nicht aufdecken, da im Allgemeinen auch die Konfiguration weiter gültig bleibt.

Da industrielle Anlagen im Laufe der Zeit verändert werden, ist mit dem Zusammenschalten oder Erweitern von Netzen, in denen Geräte, mit dann nicht mehr eineindeutiger sicherheitsgerichteter Adressierung, verwendet werden, zu rechnen. Die typgleichen Geräte akzeptieren dann möglicherweise auch die Konfiguration eines entsprechenden anderen Geräts.

Weitere Szenarien der Fehlbedienung leiten sich vom konkreten sicherheitsgerichteten Kommunikationsprotokoll und dessen Engineering-Werkzeug ab.

Da ein sicherheitsgerichtetes Kommunikationsprotokoll einige der aufgezeigten Gefahren nicht selber durch technische Maßnahmen beherrscht oder beherrschen kann, ist es erforderlich, dass die Anforderungen (Auflagen) für den Einsatz des Protokolls dem Betreiber mitgeteilt werden.

Die Entwicklung und Verifikation von sicherheitsgerichteten Protokollen setzt zahlreiche Kenntnisse der Sicherheitstechnik und ein vertieftes Verständnis der Kommunikationstechnologien voraus. Dies stellt insbesondere dann eine Gefahr dar, wenn bei Hersteller zum ersten Mal ein solches Produkt entwickeln.

---

[49] [IEC 61508-1] Kapitel 7.4.1.1

[50] [IEC 61508-4] Kapitel 3.1.14

[51] [EN 13849-1] Kapitel 4.19

[52] [EN 13849-1] Kapitel 4.1

## 2.3 Maßnahmen

Die Maßnahmen der sicherheitsgerichteten Kommunikationsprotokolle zur Erkennung und Beherrschung müssen die Qualität aufweisen, die für das jeweilige zu erreichende Sicherheitslevel ausreichend ist.

*Tabelle 2.1: Safety-Integrity-Level*[53]

| **Safety integrity level (SIL)** | **Average probability of dangerous failure on demand of the safety function for low demand mode of operation ($PFD_{avg}$)** | **Average frequence of a dangerous failure of the safety function [$h^{-1}$] (PFH)** |
|---|---|---|
| 4 | $\geq 10^{-5}$ to $< 10^{-4}$ | $\geq 10^{-9}$ to $< 10^{-8}$ |
| 3 | $\geq 10^{-4}$ to $< 10^{-3}$ | $\geq 10^{-8}$ to $< 10^{-7}$ |
| 2 | $\geq 10^{-3}$ to $< 10^{-2}$ | $\geq 10^{-7}$ to $< 10^{-6}$ |
| 1 | $\geq 10^{-2}$ to $< 10^{-1}$ | $\geq 10^{-6}$ to $< 10^{-5}$ |

Wie für die Kommunikation in IEC 61508-2[54] gefordert, muss die „Residual Error Rate“ kleiner als die zulässige Grenze für den jeweiligen Safety-Integrity-Level sein.

In IEC 61784-3 wird das maximal empfohlene Restrisiko des Kommunikationssystems auf 1% der Grenze des jeweiligen Safety-Integrity-Level's festgelegt.

Damit folgt für Kommunikationssysteme eine einzuhaltenden Restfehlerrate je nach SIL.

*Tabelle 2.2: Restfehlerrate für Kommunikationssysteme*[55]

| **Applicable for safety functions up to SIL** | **Maximum permissible residual error rate for the functional safety communication system** | **Probability of a dangerous failure per hour for the function safety communication system** |
|---|---|---|
| 4 | $< 10^{-10}$ $h^{-1}$ | $< 10^{-10}$ $h^{-1}$ |
| 3 | $< 10^{-9}$ $h^{-1}$ | $< 10^{-9}$ $h^{-1}$ |
| 2 | $< 10^{-8}$ $h^{-1}$ | $< 10^{-8}$ $h^{-1}$ |
| 1 | $< 10^{-7}$ $h^{-1}$ | $< 10^{-7}$ $h^{-1}$ |
| NOTE Values in this table are based on the assumption that the functional safety communication system contributes no more than 1 % of the total failures of the safety function. | | |

Dabei wird der „low demand mode“ nicht betrachtet, da die sicherheitsgerichteten Kommunikationsprotokolle im Allgemeinen unabhängig vom Einsatzgebiet entwickelt wurden und sich so den erhöhten Anforderungen des „high demand mode“, auch „continuous mode“ genannt, unterziehen.

[53] aus [IEC 61508-1] Kapitel 7.6.2.10, Tabelle 2 und 3

[54] [IEC 61508-2] Kapitel 7.4.11

[55] aus IEC 61784-3-5.6.2, Tabelle 3

Zur Erkennung von Nachrichtenverfälschungen kann die Technik des Schutzes der Nachricht durch Cyclic Redundancy Codes[56] angewendet werden. Weiterhin kann die Technik des mehrfachen, meist doppelten, Übertragens mit Vergleich in der empfangenden Komponente eingesetzt werden.

Die sicherheitsgerichteten Protokolle setzen in der überwiegenden Zahl der Fälle die CRC Technik ein. Einige Protokolle setzen auch beide Techniken ein. Wie zum Beispiel FF-SIF[57] und CIP-Safety[58].

„Für den Einsatz des Nachrichtenschutzes mittels CRC ist darzulegen, wie hoch das verbleibende Risiko einer unerkannten Nachrichtenverfälschung trotz CRC-Überprüfung ist.“[59]

Die Restfehlerrate einer Kommunikationsverbindung berechnet sich aus

$$\Lambda = R(p) \bullet 3600 \bullet \nu \qquad (2.1)$$

Dabei steht

*R(p)* für die Restfehlerwahrscheinlichkeit des Mechanismus zur Erkennung von Nachrichtenverfälschungen bei der Bitfehlerrate *p* des Übertragungssystems

*v* für die Anzahl der Nachrichten einer Kommunikationsverbindung je Sekunde

*3600* zur Skalierung des Wertes auf die Restfehlerwahrscheinlichkeit je Stunde.

Diese Berechnung betrachtet eine einzelne Kommunikationsverbindung. Zur Bewertung einer Sicherheitsfunktion ist es jedoch erforderlich, dass alle für die Sicherheitsfunktion genutzten Verbindungen herangezogen werden.

Somit ergibt sich[60]

$$\Lambda = R(p) \bullet 3600 \bullet \nu \bullet c \qquad (2.2)$$

Dabei steht

*c* für die Anzahl der Kommunikationsverbindungen einer Sicherheitsfunktion.

Zur Betrachtung von *R(p)* einer Nachricht wird die Restfehlerwahrscheinlichkeit für eine unerkannt verfälschte Nachricht, ausgehend vom Modell für den binären symmetrischen Kanal, ermittelt[61].

Ein binärer Kommunikationskanal wird symmetrisch genannt, wenn die Wahrscheinlichkeiten $P_{f0}=P_{f1}=p$ identisch sind. Dabei ist $P_{f1}$ die Wahrscheinlichkeit für die Verfälschung eines 1-Bits und $P_{f0}$ die Wahrscheinlichkeit für die Verfälschung eines 0-Bits.

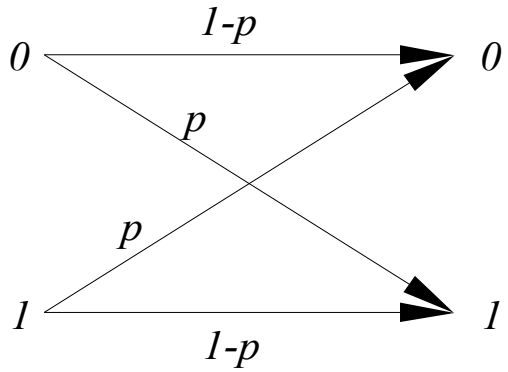


*Abbildung 2.1: Übergangswahrscheinlichkeiten des binär symmetrischen Kanals*

---

[56] abgekürzt CRC

[57] [FF-SIF]

[58] [CIP5]

[59] [Boer07]

[60] siehe z.B. [IEC 61508-2]

[61] [Boer07]

Daraus folgt, dass für eine übertragene Nachricht der Länge *N*-Bits die Wahrscheinlichkeit für *i*-Bitfehler wie folgt berechnet werden kann:

$$P_N(i)=\binom{N}{i}\bullet p^i\bullet(1-p)^{N-i} \qquad (2.3)$$

Setzt man für den Sicherungsmechanismus eine Hamming-Distanz *d* voraus, so berechnet sich die Restfehlerwahrscheinlichkeit einer unerkannt verfälschten Nachricht mit[61]:

$$R(p,N,k)\le\sum_{n=d}^{N+k}\binom{N+k}{n}\bullet p^n\bullet(1-p)^{(N+k-n)} \qquad (2.4)$$

Werden CRCs als Sicherungsmechanismus eingesetzt, so wird obige Gleichung durch die Gewichte $A_n^{N+k}$ bestimmt. $A_n^{N+k}$ ist die Menge von Codeworten der Länge $N+k$, die bei *n*-Bit-Verfälschungen durch den CRC Sicherungsmechanismus nicht erkannt werden.

$$R(p,N,k)=\sum_{n=1}^{N+k}A_n^{N+k}\bullet p^n\bullet(1-p)^{(N+k-n)} \qquad (2.5)$$

mit

*k* für den Grad des CRC-Polynoms – 1

Gemäß IEC 61784-3 und IEC 61508-2 ist die Restfehlerwahrscheinlichkeit für eine unerkannt verfälschte Nachricht für propere CRC-Polynome wie folgt zu bestimmen:

$$R_{CRC}(p,N,k)\approx\frac{1}{2^k}\bullet\sum_{n=d}^{N+k}\binom{N+k}{n}\bullet p^n\bullet(1-p)^{(N+k-n)} \qquad (2.6)$$

Die Motivation zu dieser Berechnung liegt darin begründet, dass sich die Gewichte nur mit großem Aufwand oder gar nicht berechnen lassen und so eine handhabbare Methode gesucht wurde. Die mathematische Begründung liegt darin, dass propere CRCs bei großen Bitfehlerraten ($p=0,5$) einen Restfehler von $2^{-k}$ haben, wie dies mathematisch nachweisbar bei großen Nachrichtenlängen gegeben ist[62].

Ein allgemeingültiger mathematischer Nachweis, insbesondere für kleine Bitfehlerraten und kleine Nachrichtenlängen, ist derzeit nicht bekannt.

„Propere CRC Polynome werden dadurch charakterisiert, dass ihre Restfehlerwahrscheinlichkeit mit der Bitfehlerrate p bei fester Nachrichtenlänge monoton ansteigt. Der Gradient dieses Anstiegs ist die minimale Hamming-Distanz dieser Nachrichtenlänge.“[63]

„Weiterhin decken CRCs auch Fehler auf, die oberhalb ihrer Hamming-Distanz liegen. So ist nachgewiesen, dass CRCs Burst-Fehler bis zur Länge *k* immer erkennen. Ebenfalls nachgewiesen ist, dass, wenn das CRC-Polynom den Term *(x+1)* enthält, die Erkennung aller ungeraden Bitfehler (Parity-Funktion) gewährleistet ist.“[63]

Für die Berechnung der Qualifizierung des CRC-Sicherungsverfahrens ist weiterhin zu berücksichtigen, dass die Hamming-Distanz eines CRC-Polynoms von der Länge $N+k$ der Nachricht abhängt. Im Allgemeinen sinkt die Hamming-Distanz, je länger die Nachricht wird. Im Gegenzug kann für eine Anwendung, wenn nur kurze Nachrichten verwendet werden, eine günstige (=höhere) Hamming-Distanz eingesetzt werden[64].

---

62 [IEC 61784-3]

63 [Pete81]

64 [Koop02], [Cast93]

Liegt der Betrachtung zum Schutz der Nachrichten die CRC-Technik zu Grunde und wendet man zusätzlich die *m*-fache Übertragung mit Vergleich an, so lässt sich die Restfehlerrate wie folgt berechnen

$$\Lambda_{\mathrm{m}} = R(p^m) \bullet 3600 \bullet \nu \bullet c \qquad (2.7)$$

sofern davon ausgegangen werden kann, dass die *m* Nachrichten auf bezüglich der Fehlerannahmen unabhängige Weise übertragen werden. Für das Modell des binär symmetrischen Kanals ist dies gegeben. Meist wird bei Mehrfachübertragungen die doppelte Übertragung angewendet, siehe CIP-Safety oder Foundation-Fieldbus FF-SIF.

Zur Erzielung einer reduzierten Restfehlerwahrscheinlichkeit für Nachrichten kommen auch Maßnahmen zum Einsatz, die die Daten mit 2 verschiedenen CRC-Polynomen sichern.

Mathematisch ist die daraus resultierende Qualität der Sicherung bislang jedoch nicht allgemeingültig darstellbar.

Offensichtlich würde bei der Verwendung zweier identischer CRC-Polynome über die selben Nutzdaten, die unerkannten Verfälschungen die gleichen sein und somit zu keiner Reduktion der Restfehlerwahrscheinlichkeit führen. Dabei wurde allerdings nicht berücksichtigt, dass dies nur für Verfälschungen im Nutzdatenteil der Nachricht gilt und nicht für die Verfälschungen im übertragenen CRC selbst. Somit können bei im Vergleich zur CRC-Länge kleinen Nutzdatenlängen sehr wohl Reduktionen von *R(p)* erreicht werden.

Eine andere Situation ergibt sich, wenn die beiden CRC-Polynome derart gewählt werden, dass sie zueinander teilerfremd sind. Das heißt, die Terme der Primfaktorzerlegung der Polynome ergibt für die beiden CRCs keine gleichen Faktoren. Die Motivation in der Auswahl solcher CRCs liegt darin, dass ein CRC dann „versagt“, wenn er das Codewort (die Nachricht) mit Rest 0 teilt. Damit dies für beide CRCs zutrifft, ist es erforderlich, dass das Codewort ein gemeinsames Vielfaches darstellt. Je größer dieses Vielfache ist, desto unwahrscheinlicher ist das Versagen beider CRCs bei einer Verfälschung.

Diese Eigenschaft kann, wie gesagt, zur Zeit nicht mathematisch belegt werden! Für kleine Nachrichtenlängen lässt sich dies algorithmisch mit vertretbarem Aufwand ermitteln. Jedoch stellt sich diese Fragestellung meist erst bei größeren Nachrichtenlängen und mindestens 32-Bit-CRCs, deren algorithmische Berechnung mit exponentiellem Aufwand einhergeht.

Unbestritten ist jedoch, dass die Verwendung eines derartigen Sicherungsmechanismus eine deutlich erhöhte Aufdeckung von Verfälschungen aufweist, als ein einzelner der verwendeten CRCs. Wird die Berechnung zusammen mit einer sehr konservativen Annahme der Bitfehlerrate $p=10^{-2}$ durchgeführt und unterstellt man die Unabhängigkeit der Sicherungseigenschaften beider CRCs, so können zwei Berechnungsmethoden als gute Schätzung angewendet werden.

Bei der ersten Methode wird angenommen, dass die Hamming-Distanz *d* der kombinierten CRCs gleich der Summe der einzelnen Hamming-Distanzen *d1+d2* ist und die Länge *k* der CRCs ebenfalls addiert wird. Somit wird die Restfehlerwahrscheinlichkeit einer unerkannt verfälschten Nachricht mit

$$R_{CRC1+CRC2}(p, N, k) \approx \frac{1}{2^k} \bullet \sum_{n=d}^{N+k} \binom{N+k}{n} \bullet p^n \bullet (1-p)^{(N+k-n)} \qquad (2.8)$$

approximiert, wobei:

*k* Summe der Länge der beiden CRCs in Bits

*d* *d1+d2,* Summe der Hamming-Distanzen der beiden CRCs

angenommen wird.

Die zweite Methode verwendet den üblichen Ansatz unabhängiger Wahrscheinlichkeiten und berechnet die resultierende Restfehlerwahrscheinlichkeit aus dem Produkt der Einzelwahrscheinlichkeiten. Mit der approximativen Gleichung $R_{CRCi}()$ erhält man:

$$R_{CRC1+CRC2}(p, N) \approx R_{CRC1}(p, N, k_1) \bullet R_{CRC2}(p, N, k_2) \qquad (2.9)$$

wobei:

$k_i$ Länge der jeweiligen CRCs in Bits

Wird eine dieser Berechnungsmodelle angewendet, so sollte die resultierende Restfehlerwahrscheinlichkeit, auch bei der Worst-Case Bitfehlerrate $p=10^{-2}$, mindestens einer Größenordnung unter dem geforderten Limit des Safety-Integrity-Level bleiben.

Besser ist es jedoch, die konkreten Gewichte zu berechnen, da die Approximationen von den realen Eigenschaften der CRCs und auch von den kombinierten, abweichen kann.

Für relativ kleine Nachrichtenlängen zeigt diese Arbeit bei der Analyse des TSP2 Protokolls, in welcher Relation die approximativen Rechnungen zu den realen Restfehlerwahrscheinlichkeiten stehen.

Eine weitere, eher selten angewendete Maßnahme zur Erkennung von Übertragungsfehlern, ist das Zurücksenden der empfangenen Daten und der Vergleich beim ursprünglichen Absender. Ist dieser Vergleich erfolgreich, so sind die Daten unverfälscht übertragen worden und die nächsten Daten können versendet werden. Andernfalls werden die Daten erneut gesendet. Dies wiederholt sich solange, bis sie positiv quittiert wurden oder die Überwachungszeit abgelaufen ist.

Die Restfehlerwahrscheinlichkeit dieser Technik bestimmt sich durch die Wahrscheinlichkeit, dass die verfälschten Daten vom ursprünglichen Absender bei der Quittierung wieder in die „richtigen“ Daten verfälscht werden.

Nimmt man beispielsweise eine Übertragung mittels Dual-Port-RAM an und unterstellt, dass gewisse Adressbereiche die Daten fälschlicherweise invertieren, so muss davon ausgegangen werden, dass der Fehler in der Übertragung nicht erkannt wird, ohne dass weitere Maßnahmen zur Fehleraufdeckung herangezogen werden.

Aus diesem Grund müssen die Daten in den Feedback-Nachrichten anders dekodiert werden, damit derartige Fehler nicht unentdeckt bleiben. Im einfachsten Fall ist eine Invertierung ausreichend, jedoch sind je nach Übertragungssystem auch komplexere Maßnahmen erforderlich.

Es ist daher notwendig die verwendeten Übertragungseinrichtungen sehr gründlich zu analysieren und die daraus resultierende Dekodierungstechnik zu ermitteln.

Wird eine geeignete Technik verwendet, erhält man durch Feedback-Nachrichten mindestens eine Quadrierung der Restfehlerwahrscheinlichkeit der einzelnen Nachricht.

In der Mehrzahl der eingesetzten sicherheitsgerichteten Übertragungseinrichtungen kommen serielle Übertragungstechniken zum Einsatz. Daher wird die anzunehmende Bitfehlerrate zu einer der bestimmenden Größen für die Restfehlerwahrscheinlichkeit.

Gemäß IEC 61508-2 können zwei Betrachtungsweisen herangezogen werden.

"The techniques and measures necessary to ensure the required failure measure (such as the residual error rate) of the communication process (see 7.4.11.1) shall be implemented according to the requirements of this standard and IEC 61508-3. This allows two possible approaches:

- the entire communication channel shall be designed, implemented and validated according to the IEC 61508 series and (IEC 61784-3 or IEC 62280). This a so-called 'white channel' (see Figure 7 a); or

- parts of the communication channel are not designed or validated according to the IEC 61508 series. This is a so-called 'black channel' (see Figure 7 b). In this case, the measures necessary to ensure the failure performance of the communication process shall be implemented in the E/E/PE safety-related subsystems or elements that interface with the communication channel in accordance with the IEC 61784-3 or IEC 62280 series as appropriate"[65]

Für den ersten Fall, des „white channel", sind die Restfehlerwahrscheinlichkeiten der einzelnen Komponenten der Datenübertragung gemäß den üblichen HW-/SW-Techniken der IEC 61508 zu bestimmen und anzuwenden. Daher kann dann auch die konkrete anzunehmende Bitfehlerrate ermittelt und zur weiteren Berechnung der Restfehlerrate herangezogen werden.

Für den Fall, dass für den Anwendungsbereich des sicheren Protokolls das „black channel" Modell gewählt wird und keine weiteren Annahmen über die Bitfehlerrate getroffen werden können, ist eine Bitfehlerrate von $10^{-2}$ anzunehmen[66].

*Anmerkung: Werden keine oder nicht nur seriellen Übertragungstechniken genutzt, so ist zu prüfen, ob das Modell vom binär symmetrischen Kanal zur Anwendung kommen kann oder ob andere Modelle erstellt werden müssen. So ist z.B. zu untersuchen, wie sich ein Übersprechen paralleler Datenleitungen auf das Fehlermodell auswirkt, da hier nicht mehr von einer Unabhängigkeit der Verfälschung einzelner Bits ausgegangen werden kann.*

## Beispielbetrachtung für Safety over EtherCAT CRC

Safety over EtherCAT[67] benutzt einen 16 Bit CRC 0x1_39B7. Zum Vergleich der oben angeführten Berechnungsmethode für die Restfehlerwahrscheinlichkeit einer Nachricht, wurden nachfolgend die möglichen unerkannten Fehler für das Polynom berechnet. Dabei wurde beispielhaft von einem Telegramm mit 2-Bytes Nutzdaten ausgegangen. Weiterhin wurde davon ausgegangen, dass bis auf die Nutzdaten und den CRC, jede Verfälschung durch die Erwartungshaltung aufgedeckt wird. Als Bitfehlerrate wird wie üblich $10^{-2}$ eingesetzt, da Safety over EtherCAT keine Einschränkungen für den vorgesehenen Anwendungsbereich macht.

*Tabelle 2.3: Restfehlerwahrscheinlichkeit Safety over EtherCAT[68]*

| ***Anzahl unerkannter fehlerhafter Nachrichten*** | ***Anzahl verfälschter Bits der Nachricht*** | ***Restfehlerwahrscheinlichkeit*** |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 2 | 0 |
| 0 | 3 | 0 |
| 0 | 4 | 0 |
| 0 | 5 | 0 |
| 25 | 6 | $1{,}92511 \cdot 10^{-11}$ |
| 14 | 7 | $1{,}08895 \cdot 10^{-13}$ |
| 149 | 8 | $1{,}17066 \cdot 10^{-14}$ |
| 525 | 9 | $4{,}16647 \cdot 10^{-16}$ |
| 915 | 10 | $7{,}33492 \cdot 10^{-18}$ |
| 2006 | 11 | $1{,}62431 \cdot 10^{-19}$ |

[65] [IEC 61508-2] Kapitel 7.4.11.2

[66] [IEC 61784-3]

[67] [FSoEC]

[68] Ergebnisse des im Rahmen der Arbeit entwickelten Simulationsverfahrens

| *Anzahl unerkannter fehlerhafter Nachrichten* | *Anzahl verfälschter Bits der Nachricht* | *Restfehlerwahrscheinlichkeit* |
|---|---|---|
| 3498 | 12 | $2{,}86104 \cdot 10^{-21}$ |
| 5110 | 13 | $4{,}22172 \cdot 10^{-23}$ |
| 7364 | 14 | $6{,}14536 \cdot 10^{-25}$ |
| 8630 | 15 | $7{,}27460 \cdot 10^{-27}$ |
| 8981 | 16 | $7{,}64694 \cdot 10^{-29}$ |
| 8784 | 17 | $7{,}55475 \cdot 10^{-31}$ |
| 7176 | 18 | $6{,}23412 \cdot 10^{-33}$ |
| 5310 | 19 | $4{,}65964 \cdot 10^{-35}$ |
| 3486 | 20 | $3{,}08994 \cdot 10^{-37}$ |
| 1898 | 21 | $1{,}69935 \cdot 10^{-39}$ |
| 1019 | 22 | $9{,}21565 \cdot 10^{-42}$ |
| 420 | 23 | $3{,}83677 \cdot 10^{-44}$ |
| 141 | 24 | $1{,}30107 \cdot 10^{-46}$ |
| 67 | 25 | $6{,}24484 \cdot 10^{-49}$ |
| 13 | 26 | $1{,}22392 \cdot 10^{-51}$ |
| 4 | 27 | $3{,}80396 \cdot 10^{-54}$ |
| 0 | 28 | 0 |
| 0 | 29 | 0 |
| 0 | 30 | 0 |
| 0 | 31 | 0 |
| 0 | 32 | 0 |
| | **Summe** | $\mathbf{1{,}9372 \cdot 10^{-11}}$ |

Die Restfehlerwahrscheinlichkeit je Nachricht mit 2 Bytes Nutzdaten und 2 Byte CRC ergibt somit $1{,}9372 \cdot 10^{-11}$. Zieht man die Gleichung (2.6) heran, so ergibt die Berechnung für $p=10^{-2}$, $N=16$, $k=16$ und $d=6$, $R() = 1{,}0641 \cdot 10^{-11}$.

Für den hier betrachteten Safety over EtherCAT CRC zeigt sich, dass die Gleichung (2.6) einen etwas zu optimistischen Wert ergibt, auch wenn sich dieser noch in der selben Größenordnung, befindet.

Eingesetzt in die Gleichung (2.2) ergibt sich mit $c=2$ und $v=10$ für die Restfehlerwahrscheinlichkeit der Sicherheitsfunktion $\Lambda = 7{,}66152 \cdot 10^{-7}$.

**Wiederholungen, Einfügungen, Reihenfolge und Verluste beherrschen**

Als Maßnahme zur Beherrschung von Nachrichtenwiederholungen, Einfügungen, Verlust[69] und Reihenfolgevertauschungen hat sich der Einsatz eines monotonen Nachrichtenzählers, oft Monotonie-Nummer oder Frame-Nummer genannt, weit verbreitet. Alternativ kommen auch Zeitstempel, wie zum Beispiel bei CIP-Safety, zum Einsatz.

Beiden Techniken liegt zu Grunde, dass anhand der zuvor als gültig ermittelten Nachrichtenzählers/Zeitstempels, beim Empfang der nächsten Nachricht erkennen zu können, ob sie neuer ist oder nicht. Ist sie nicht neuer, so verwerfen die Sicherheitsprotokolle diese Nachricht; andernfalls werden weitere Prüfungen durchgeführt.

Beim Zeitstempel-Verfahren wird in der Regel ein Verlust einer Nachricht nicht festgestellt, sofern zeitgemäß eine ausreichend neue Nachricht empfangen wird. Aus diesem Grund ist bei diesem Verfahren, falls keine Zusatzmaßnahmen ergriffen werden, für eine Nachricht n+1 nicht erkennbar, wenn diese vor der zuvor gesendeten, aber noch nicht empfangenen Nachricht n empfangen wird. Wird danach die Nachricht n empfangen, so wird sie verworfen.

Diese Eigenschaft ist nur dann akzeptabel, wenn die übertragenen Daten einen Update-Charakter aufweisen. Das heißt, neuere Daten sind immer geeigneter als alte Daten und es ist nicht erforderlich,

[69] Verlust wird hier nur innerhalb einer Sequenz und nicht vom zeitlichen Aspekt betrachtet.

dass alle Daten übertragen werden. Damit kann dieses Verfahren zum Beispiel nicht eingesetzt werden, wenn inkrementelle oder Kommando-Datenübertragung verwendet werden soll.

Mit dem Nachrichtenzähler-Verfahren kann der Verlust und die Vertauschung der Nachrichtenreihenfolge erkannt werden, da im Allgemeinen eine stets um 1 steigende Monotonie-Nummer erwartet wird.

Als Maßnahme zur Erkennung von Verlust, Wiederholung, Reihenfolgevertauschung und Einfügung kann auch der CRC herangezogen werden. Dies liegt daran, dass einige Sicherheitsprotokolle den Nachrichtenzähler / Zeitstempel nicht mit der Nachricht übertragen. Statt dessen benutzen sie den nicht übertragenen Wert des Nachrichtenzählers/Zeitstempels bei der CRC Berechnung der Nachricht und setzten auf der Empfängerseite eine entsprechende Erwartungshaltung voraus, sodass dieser mit dieser Erwartungshaltung ebenfalls den CRC der Nachricht berechnet und somit deren Monotonie feststellen kann. Dies hat den Vorteil, dass die Nachrichtenlänge reduziert werden kann[70].

Ein großer Nachteil dieses Verfahrens ist, dass nicht zwischen echter Nachrichtenverfälschung und Monotonie-Verletzungen unterschieden werden kann. Dies ist insbesondere dann relevant, wenn das unterlagerte Übertragungssystem mit seiner „Normalfunktion“ zu Einfügungen und Reihenfolgevertauschungen führt, wie dies bei Ethernet-Ringen, bzw. (Rapid-) Spanning-Tree Anwendungen im Falle der Rekonfiguration vorkommen kann.

Ein weiterer nachteiliger Aspekt ist, dass diese Protokolle damit sehr anfällig gegenüber absichtlichen oder unabsichtlichem Einschleusen von Nachrichten sind, da bei dieser Anwendung bei einigen Protokollen gleichzeitig eine Abschaltung der sicherheitsgerichteten Verbindung erfolgt[71].

Um die Qualifizierung der jeweiligen Maßnahmen durchzuführen, ist festzulegen, welche Wahrscheinlichkeit für das Auftreten von Einfügungen, Reihenfolgevertauschungen, Verlusten und Wiederholungen vorliegt. Auf dieser Basis ist dann zu berechnen, mit welcher Wahrscheinlichkeit die verwendete Maßnahme versagt. Dies sollte ebenfalls unter 1% der für das SIL erforderlichen Grenze liegen.

Für die zu betrachtenden Wahrscheinlichkeiten sind kaum konkrete Werte bekannt. Lediglich für Ethernet-Switch-e wurde im Rahmen der Entwicklung von PROFIsafe exemplarisch eine Fehlerrate von 100 FIT d.h. $10^{-7}$ $h^{-1}$ ermittelt[71]. Nachfolgend wird stets angenommen, dass 10 Ethernet-Switches eingesetzt werden, d.h. es wird mit einer Gesamtrate von 1000 FIT gerechnet, da in der Praxis meist 2 oder mehr Ethernet-Switch-Bausteine zu Einsatz kommen.

Die Versagenswahrscheinlichkeit ist damit zu begründen, dass die eingesetzten Monotoniezähler / Zeitstempel eine begrenzte Darstellungsgröße haben. Üblich sind hier 16 Bit bis 64 Bit. Somit versagen sie für eine einzelne Nachricht mit einer Wahrscheinlichkeit von

a) $2^{-16} = 1{,}52 \cdot 10^{-5} > 10^{-9} = 1\%$ SIL3

b) $2^{-32} = 2{,}32 \cdot 10^{-10} < 10^{-9} = 1\%$ SIL3

c) $2^{-64} = 5{,}42 \cdot 10^{-20} < 10^{-9} = 1\%$ SIL3.

Bezogen auf die unterstellte Fehlerrate ergeben sich Restfehlerwahrscheinlichkeiten von

a) $10^{-6}\ h^{-1} \cdot 2^{-16} = 1{,}52 \cdot 10^{-11}\ h^{-1}$

b) $10^{-6}\ h^{-1} \cdot 2^{-32} = 1{,}52 \cdot 10^{-14}\ h^{-1}$

c) $10^{-6}\ h^{-1} \cdot 2^{-64} = 5{,}42 \cdot 10^{-23}\ h^{-1}$

die jeweils kleiner als 1% von SIL3 sind. Dabei ist jedoch sorgfältig zu beachten, dass die angenommen Größe $10^{-6}$ $h^{-1}$ als Wahrscheinlichkeit für den Effekt mit einer Nachricht gelten muss. Ist die Fehlerrate je übertragener Nachricht zu kalkulieren, so wird man bei kleiner Darstellungsgröße von 16 Bit

---

70 [PROFIsafeV2], [FF-SIF], [TSP]

71 [PROFIsafeV2]

oberhalb von 1% von SIL3 gelangen, da ein Übertragungssystem mit 65 Nachrichten pro Stunde kaum praktikable Anwendungen finden wird.

Weiterhin ist zu beachten, dass in einem Übertragungssystem mehrere und oft sehr viele solcher Komponenten eingesetzt werden, für die jeweils 100 FIT angenommen werden müssen. Auch hierbei wird man bei kleiner Darstellungsgröße sehr schnell keine für SIL3 taugliche Beherrschung erreichen.

Eine weitere Maßnahme zur Erkennung von Nachrichtenverlusten ist die Quittierung von Nachrichten[72].

Das Einfügen von Nachrichtensequenzen, insbesondere der Anlaufsequenzen, wird von den Protokollen sehr unterschiedlich behandelt. Nachfolgend wird der Fall, dass Sequenzen deren Ursprung aus Nachrichten nach der Verbindungsaufnahme sind und in eine Verbindung eingefügt werden, nicht weiter betrachtet, da dies zu keinen anderen Ergebnissen führt, als die Betrachtung für einzelne eingefügte Nachrichten.

Es bleibt die Betrachtung, dass die Sequenz der Verbindungsaufnahme gespeichert wurde und unerwünscht zu einem anderen Zeitpunkt an einen Teilnehmer geschickt wird.

Einer der verwendeten Schutzmechanismen ist, dass die Nachrichten der Verbindungsaufnahme andere Inhalte haben, als die Nachrichten während der normalen Kommunikation. Somit sind diese als Verbindungsaufnahme-Nachrichten erkennbar. Weiterhin wird dabei nicht nur eine solche erkennbare Nachricht verwendet, sondern mehrere, wie z.B. bei PROFIsafe 3 in Folge[73].

Die spezielle Kennzeichnung und die Ausdehnung auf mehr als eine Nachricht verringert das Restrisiko, dass es zu einer ungewollten (gefährlichen) Aufnahme der Kommunikation mit alten Daten kommt. Setzt man die Wahrscheinlichkeit, dass eine Nachricht gespeichert und wiederholt wird, gleich $p_w$ und die Wahrscheinlichkeit, dass eine wiederholte (eingefügte) Nachricht nicht erkannt wird, gleich $p_e$, so ergibt sich die Restfehlerwahrscheinlichkeit

$$p_1 = p_w \bullet p_e. \qquad (2.10)$$

Die Restfehlerwahrscheinlichkeit für eine Folge gilt dann

$$p_f = (p_1)^n, \qquad (2.11)$$

wobei $n$ die Anzahl der Nachrichten der Folge ist.

Eine weitere zusätzliche Schutzmaßnahme bei Protokollen ist, dass sie die Verbindungsaufnahme nur dann gestatten, wenn die Verbindung geschlossen ist[74]. Damit ist ausgeschlossen, dass während einer etablierten Kommunikationsverbindung eine Anlaufsequenz erfolgreich eingefügt wird. Dies stellt vor dem Hintergrund, dass Kommunikationsverbindungen im Allgemeinen dauerhaft in Betrieb sind, eine nennenswerte Reduktion des Restrisikos dar. Kritisch ist jedoch anzumerken, dass gerade nach Fehlern, die zum Schließen der Kommunikationsverbindung führen, eine Wiederholung durch die „normale“ Funktion der Komponenten entstehen kann.

Um diesem und der obigen Problematik effizienter entgegenwirken zu können, verwenden Protokolle ein wechselseitiges Handshake-Verfahren, bei dem sie im Rahmen des Verbindungsaufbaus zufällige Schlüssel austauschen, deren Quittung erst zur Aufnahme des normalen Betriebs führt[75]. Die Restfehlerwahrscheinlichkeit berechnet sich somit aus dem Produkt von $p_f$ und der Wahrscheinlichkeit $p_k$, dass die gespeicherte Folge den zufälligen Schlüssel des Teilnehmers enthält.

---

[72] [IEC 61784-3]

[73] [IEC 61784-3-3]

[74] [safeEthernet] und [TSP]

[75] [safeEthernet] und [TSP]

Allen bekannten Sicherheitsprotokollen ist die Eigenschaft gleich, dass sie eine zeitliche Erwartungshaltung für den Nachrichtenempfang vom Kommunikationspartner aufweisen. Dabei kann es sich sowohl um die Quittierung einer Anfrage, aber auch nur um den Abstand der Anfragen handelt.

Wird diese Erwartungshaltung nicht erfüllt, so wird die sicherheitsgerichtete Reaktion eingeleitet.

Verwendet ein Protokoll nur den zeitlichen Abstand zweier Nachrichten des Kommunikationspartners, so erkennt es mit dieser Maßnahme alleine nicht, wenn sich die Nachrichtenverzögerung geringfügig, aber kontinuierlich vergrößert (Problem der schleichenden Delays).

Aus diesem Grund haben die meisten[76] Sicherheitsprotokolle einen irgendwie gearteten Feedback-Mechanismus, das heißt, eine Signalisierung zwischen den Kommunikationspartnern findet immer in beide Richtungen statt. Selbst bei Multi-Cast basierten Protokollen, wie CIP-Safety[77], gibt es eine Rückkopplung bezüglich des Time-Stamps vom jedem Consumer zum Provider.

Eine häufig anzutreffende Technik ist der Einsatz eines einfachen Handshake-Verfahrens. Dabei sendet eine zentrale Einheit, meist Host oder Master genannt, eine Anfrage an eine abhängige Einheit, meist Slave oder Device genannt, und erhält von dieser eine Antwort. Von der zentralen Einheit ist zu ermitteln, ob die Übertragung von Nachrichten zu und von den abhängigen Einheiten zeitgerecht und ohne unzulässigen Drift der Übertragungszeit erfolgt. Die abhängigen Einheiten überwachen die Abfrageabstände der zentralen Einheit, können aber bezüglich des Drift selber keine Aussage treffen.

Ist eine geeignete Feedback-Technik im Protokoll realisiert, bleibt diesbezüglich noch die Betrachtung der Reaktionszeiten.

Als Maßnahmen für die Aufdeckung falscher Adressierung werden zwei Verfahren angewendet.

Ein angewendetes Verfahren definiert für eine sicherheitsgerichtete Kommunikationsverbindung eine dieser Verbindung zugeordnete eineindeutige ID (Nummer). Anhand dieser Nummer kann der Empfänger erkennen, ob dies eine Sendung für die bei ihm vorhandene Kommunikationsverbindung ist. Damit die Richtung, in der die Nachricht versendet wird, erkannt werden kann, gibt es in der Nachricht ein weiteres Kennzeichen[78].

Ein weiteres Verfahren definiert für die Kommunikationspartner in einem sicherheitsgerichteten Netzwerk Adressen (Nummern), die innerhalb des Netzwerks eineindeutig sein müssen.

Diese beiden Nummern werden mit den sicherheitsgerichteten Nachrichten übertragen und ermöglichen dem Empfänger zu erkennen, ob die Nachricht vom richtigen Absender an ihn gerichtet ist.

Neben der Übertragung der Adressinformationen innerhalb der Nachricht gibt es auch ein durch Patent geschütztes Verfahren. Dabei werden die Adressinformationen nicht übertragen, gehen jedoch in die Berechnung des CRCs ein, so dass die Empfänger dies auf Grund ihrer Erwartungshaltung überprüfen können[79].

Die Eignung der jeweiligen Adressierungstechnik hängt von den Datengrößen der verwendeten Nummern ab. Im Falle des Ersatzes der Nummern durch die CRC-Berechnung kann für die Qualifizierung der Adressierungstechnik höchstens die Größe des CRCs herangezogen werden, auch dann, wenn die eingesetzten nicht übertragenen Nummern größer sind.

Der grundlegende Ansatz, den sicherheitsgerichtete Protokolle zur Behandlung des Masquerading's heranführen ist, dass angenommen wird, dass andere Protokolle nicht den selben Nachrichtenrahmen und insbesondere nicht den selben CRC(s), wie das sichere Protokoll, verwenden.

Die Qualifizierung der Annahme muss daher betrachten, mit welcher Wahrscheinlichkeit eine Nachricht erzeugt werden kann, die zufällig einer gültigen sicherheitsgerichteten Nachricht entspricht.

---

[76] Es ist kein Protokoll bekannt, dass ohne Fehlerausschluss das Problem ohne Feedback-Technik löst.

[77] [CIP5]

[78] siehe z.B. [FF-SIF] und [TSP]

[79] [PROFIsafeV2], [CIP5], [FF-SIF]

Verwendet das sichere Protokoll einen entsprechenden Schutz gegen Verfälschung, so berechnet sich die Wahrscheinlichkeit aus der zufälligen Generierung einer Nachricht der Länge $N+k$ Bits und der Versagenswahrscheinlichkeit des Verfälschungsschutzes.

Weitere Aspekte kommen zur Anwendung, wenn die obige Annahme nicht herangezogen werden kann, wie dies beispielsweise bei offenen Übertragungseinrichtungen und damit verbundener absichtlicher Unterminierung einhergeht.

Als Maßnahme gegen eine divergierende Konfiguration auf den beiden Kommunikationspartnern hat sich die Berechnung eines CRCs über diese und die Einbeziehung des CRCs in das Kommunikationsprotokoll etabliert.

Die Einbettung selbst erfolgt durch Übertragung des CRCs beim Verbindungsaufbau[80] oder durch die Einbeziehung in die CRC-Berechnung der regulären Nachrichten des Protokolls[81].

Eine weitere Variante findet sich bei safe**ethernet**[82], die die Konfigurationssignatur im Header der Nachricht transportiert.

Diese Maßnahme kann zusätzlich auch das Aufdecken einer fehlerhaften Adressierung unterstützen, unterstellt man, dass die fehlgeleitete Nachricht bei einem Gerät ankommt, dass eine andere Konfiguration besitzt, als der korrekte Adressat. Dies gilt natürlich je nach verwendeter Variante nur für die Verbindungsaufnahme oder wie bei PROFIsafeV2 und safe**ethernet** bei jeder Nachricht.

Bei allen Varianten kommt der zusätzliche Adressschutz nicht zur Anwendung, wenn die Adressen von Geräten vertauscht werden, die die gleiche Konfiguration besitzen, da sie zum Beispiel baugleich sind.

Um eine fehlerhafte Konfiguration aufzudecken, bevor sie zu einer Gefährdung führt, ist es erforderlich, dass der Anwender die Funktion seiner Sicherheitskreise in der Anlage prüft. Damit kann aufgedeckt werden, das die richtigen Eingangsdaten von der richtigen Applikation zu den richtigen Ausgangsdaten führt.

Ebenfalls muss der Anwender prüfen, ob die Reaktionszeit über die sicherheitsgerichtete Kommunikation wie erwartet und für den Prozess geeignet ist.

Ferner ist auf theoretischer Basis die Worst-Case-Reaction-Time für jede Sicherheitsfunktion, vom physikalischen Eingang, der Eingangskomponente, der sicherheitsgerichteten Kommunikationsverbindung zur Sicherheitssteuerung, deren Verarbeitung, der sicherheitsgerichteten Kommunikationsverbindung zur Ausgangskomponente, bis zu deren physikalischem Ausgang, zu berechnen und mit den Anforderungen des Prozesses abzugleichen. Die Reaktionszeitmessung ist, wie in Kapitel 2.1angeführt, für diese Betrachtung nicht tauglich.

Kommen in einer Anwendung Komponenten verschiedener Hersteller zum Einsatz, so ist vom Anwender eine intensive Zusammenarbeit mit den Herstellern anzustreben, da es bislang keine allgemeingültigen Berechnungsvorschriften für die Worst-Case-Reaction-Time gibt.

Schlussendlich sollte immer durch eine unabhängige Person und je nach Gefährdungspotential, durch eine unabhängige Organisation, die Prüfung der Konfiguration der sicherheitsgerichteten Kommunikation erfolgen. Nur so lässt sich dem absehbaren Missbrauch begegnen, da hier die Technik keine Lösung anbieten kann. Es bleibt in der Verantwortung des Anwenders die Kommunikation sicher einzusetzen.

Alle gängigen sicherheitsgerichteten Kommunikationsprotokolle unterstellen, dass vom Anwender eine eineindeutige Adressierung der Kommunikationspartner erfolgt. Die Einstellung der richtigen

---

[80] siehe z.B. [TSP], [FSoEC], [CIP5]

[81] siehe [PROFIsafeV2]

[82] [safeethernet]

Adresse liegt im Allgemeinen außerhalb der Betrachtung der durch die Protokolle selber realisierten Maßnahmen.

Der Anwender bekommt hierbei wiederum eine besondere Verantwortung. Dieser sollte er durch geeignete Verifikation der Konfiguration und Einstellung der Geräte gerecht werden. Das dies im Rahmen der Inbetriebnahme meist selbstverständlich ist, versteht sich von selbst.

Jedoch ist es sehr wichtig, dass der Anwender Verfahren entwickelt, dokumentiert und schult, die auch im Fall einer Wartungsaktivität die eineindeutige Adressierung gewährleistet. Dabei sollte immer ein Konzept vorliegen, wie auch beim Ausfall mehrerer, insbesondere baugleicher Geräte, ein sicherer Gerätetausch mit Adresseinstellung möglich ist.

Maßnahmen gegen unberechtigten Zugriff auf sichere Kommunikationseinrichtungen können durch physischen Zugangsschutz und dem Einsatz von Authentifizierung erzielt werden.

Eine wesentliche Maßnahme zum Schutz vor Gefährdungen des Kommunikationssystems ist, dass der Zugang zu den sicherheitsgerichteten Komponenten nur autorisierten Personen möglich ist.

Für Kommunikationssysteme sollte dies ebenfalls erfolgen, wenngleich dies auf Grund der Ausdehnung und den örtlichen Gegebenheiten nicht immer möglich oder wirtschaftlich tragbar ist.

Da der physische Schutz von Netzwerkinstallation nicht immer möglich ist, sollten die sicherheitsgerichteten Systeme Maßnahmen zur Authentifizierung von Benutzern aufweisen, so dass nur berechtigte Personen deren Konfiguration und damit auch die der sicherheitsgerichteten Kommunikation, verändern können.

Im besonders gefährdeten Bereich der Ethernet-Übertragungssysteme gibt es zahlreiche weitere Maßnahmen, die das Gefährdungsrisiko reduzieren. Beim Einsatz von WiFi oder Bluetooth können starke kryptographische Verfahren, gekoppelt mit dazu passender Authentifizierung[83] zum Einsatz kommen[84]. Auch für den Anschluss von Ethernet-Knoten an Switches können heute Techniken zur Authentifizierung[85] von Gerät und Anwender eingesetzt werden.

Schlussendlich ist der Einsatz von Firewalls heute eine Selbstverständlichkeit zur Abschottung der Netze nicht nur nach außen, sondern auch innerhalb der Anlage.

Der Einsatz von kryptographischen Methoden im Sicherheitsprotokoll oder dessen Übertragungssystems, ist eine Maßnahme, die der Gefährdung durch absichtliche Unterminierung der Sicherheitsmechanismen entgegen wirkt und gleichzeitig die Gefährdung bezüglich Masquerading reduziert.

Durch die Verwendung, im Vergleich zur herkömmlichen Adressierung und CRCs, großer Schlüssel, mit 128-Bit, 256-Bit oder mehr, sinkt das Risiko einer zufälligen korrekten Nachricht deutlich unter das für SIL3 erforderliche Level.

Baut man in der Schlüsselgenerierung Techniken mit ein, die mit dem Zeitpunkt der Generierung einen weiteren Zufallsanteil, zusätzlich zu den Adressinformationen enthält, so wirkt diese Technik auch gegen die bei zukünftigen Erweiterungen oder in offenen Netzen zu unterstellende Vergabe nicht eindeutiger Adressen, bzw. Verbindungs-IDs.

Wenn die Schlüssel den „Angreifern“ nicht bekannt sind, so ist auch eine absichtliche Unterminierung erheblich schwieriger. Selbst bei dem „Angreifer“ bekannten initialen Schlüsseln kann durch die Nutzung von Schlüsselsequenzen ein Angriff auf die sicherheitsgerichtete Kommunikation erheblich erschwert werden.

Durch Einsatz von Kryptographie wird das Risiko des unberechtigten Zugriffs ebenfalls reduziert, wenn die Technik auch für den Bedien- und Wartungszugang zum System verwendet wird.

---

83 RADIUS Protokoll

84 [PROFIsafeV2]

85 Switches mit Port-basierter Authentifizierung

Frühe Erwähnung finden kryptographische Methoden in Zusammenhang mit sicheren Protokollen in der EN 50129-2, bzw. deren Vorgängerversion und auch in den nachfolgenden IEC 61784-3 und IEC 61784-3-3.

Um den Anwender mit den Anforderungen für den Einsatz und den Betrieb eines sicherheitsgerichteten Kommunikationssystems vertraut zu machen, sollte der Hersteller dem Anwender entsprechende Sicherheitsanweisungen zur Verfügung stellen. Diese sollten die allgemeinen Anforderungen für den Einsatz von Kommunikationssystemen, wie auch die speziellen Aspekte des verwendeten Sicherheitsprotokolls darlegen, die der Betreiber zu berücksichtigen hat.

Dies wird in der IEC 61508-2 ganz allgemein für den Einsatz von sicherheitsgerichteten Systemen, wie auch in IEC 61784-3, für die sicherheitsgerichtete Kommunikation gefordert.

Die Entwicklung und Verifikation von sicherheitsgerichteten Protokollen ist im Allgemeinen im Rahmen einer IEC 61508 konformen Produktentwicklung vorzunehmen. Der dabei zum Einsatz gebrachte Entwicklungsprozess muss die Anforderungen der IEC 61508 erfüllen und ist durch Audits gemäß Functional Safety Management zu verifizieren.

Die mit der Spezifikation und der Verifikation von sicherheitsgerichteten Protokollen betrauten Personen müssen profunde Kenntnisse in der Sicherheitstechnik, der Kommunikationstechnologie und der Software-Entwicklung in Realzeitsystemen aufweisen. Diesen Anspruch unterstreicht auch die IEC 61508-1, die entsprechend qualifizierte Personen für den Entwicklungsprozess fordert.

# 3 Reaktionszeit der Kommunikationsprotokolle

Die Reaktionszeit ist eine wesentliche Leistungsgröße sicherheitsgerichteter Kommunikationsprotokolle. In allen relevanten Anwendungen wird die Dauer der Reaktion auf ein Eingangsereignis, bis zur Reaktion am Ausgang, als entscheidende Größe betrachtet.

Sind in der Kette von Eingang bis zum Ausgang sicherheitsgerichtete Kommunikationsstrecken enthalten, so stellen deren typische Reaktionszeiten, wie auch deren garantierte maximale Reaktionszeiten, sich als Teil der gesamten Dauer der Reaktion dar.

Als Modell für eine solche Kette werden typischerweise 5 Komponenten betrachtet. Begonnen wird diese Kette mit dem Eingangsmodul, z.B. ein sicherheitsgerichteter, dezentraler, digitaler Eingang, an dem ein Sensor, Sicherheitsschalter oder ähnliches angeschlossen ist. Daran schließt sich die sicherheitsgerichtete Kommunikation unter Anwendung eines spezifischen Kommunikationsprotokolls an.

Die nächste Komponente stellt die Applikationslogik verarbeitende Komponente dar. Dabei handelt es ist im Allgemeinen um eine Sicherheitssteuerung mit sicherheitsgerichteter Logikverarbeitung.

Das Ergebnis der Logikverarbeitung wird wiederum durch eine sicherheitsgerichtete Kommunikation, meist unter Verwendung des selben sicherheitsgerichteten Kommunikationsprotokolls, wie zwischen Eingangsmodul und Sicherheitssteuerung, zum Ausgabemodul übertragen.

Den Abschluss der Kette bildet das Ausgabemodul, das das Ergebnis der Logik in die nachgeschalteten Prozessglieder, z.B: Relais, Pumpen oder Ventile speist.

Die Reaktionszeit der Kette setzt sich aus den Zeiten der einzelnen Komponenten zusammen. Dabei werden maximale und Worst-Case Reaktionszeiten betrachtet. Bei diesen Zeitbetrachtungen werden vorerst keine spezifischen Implementierungsmodelle vorausgesetzt.
Die Zeiten der einzelnen Komponenten sind wie folgt definiert.

- $t_{in}$ das Alter eines Eingangssignals, wenn es von der Eingangskomponente (siehe DI in Abbildung 3.1) für die sicherheitsgerichtete Kommunikation verwendet werden kann.
- $t_{com\text{-}in}$ die Zeit, die die Übertragung eines neuen von der Eingangskomponente erkannten Signalwerts in einer Nachricht zur Sicherheitssteuerung dauert.

$t_{logic}$ die Zeit, die die Sicherheitssteuerung mit ihrer Logik benötigt, bis sie auf den mit der Nachricht eingetroffenen Signalwert ein Ergebnis berechnet hat.

$t_{com\text{-}out}$ die Zeit, die die Übertragung eines neuen berechneten Ergebnisses der Sicherheitssteuerung in einer Nachricht zur Ausgangskomponente dauert.

$t_{out}$ die Zeit, die die Ausgangskomponente (siehe DO in Abbildung 3.1) benötigt, bis sie das, mit der Nachricht eingetroffene, Ergebnis an ihrem Ausgang einstellt.

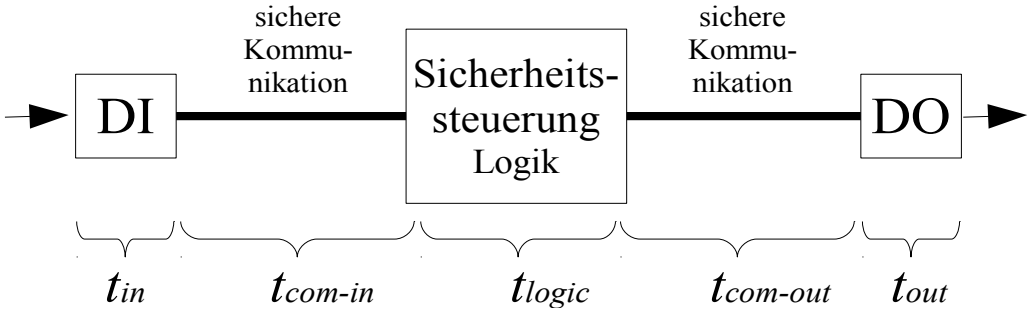


*Abbildung 3.1: Verarbeitungskette*

Die maximale Reaktionszeit $t_{max}$ ist als die Zeit definiert, die die Komponenten mit ihrer Funktion, **ohne** das Vorhandensein eines Fehlers, maximal benötigen, um auf ein Ereignis am Eingang, mit einer Reaktion am Ausgang zu reagieren. Für Kommunikationsprotokolle ist hier meist die Zeit relevant, die das maximale Alter der Daten eines korrekt empfangenen und akzeptierten Nachrichtenpaketes beschreibt.
Hierbei ist vor dem Hintergrund der Betrachtung sicherheitsgerichteter Kommunikationssysteme der Fall zu berücksichtigen, bei dem ohne Verletzung von sicherheitsgerichteten Überwachungen und ohne Störungen im Übertragungssystem, die Kommunikation durchgeführt werden kann.
Die maximale Reaktionszeit der Kette ist:

$$t_{max} = t_{max\text{-}in} + t_{max\text{-}com\text{-}in} + t_{max\text{-}logic} + t_{max\text{-}com\text{-}out} + t_{max\text{-}out} \qquad (3.1)$$

Dies entspricht der Sichtweise vieler Spezifikationen, PROFIsafe[86] sei hier stellvertretend genannt.

Für die einzelnen Zeitanteile werden jeweils die maximalen Zeiten der einzelnen Komponenten gewählt. Die maximale Zeit der Logikverarbeitung $t_{max\text{-}logic}$ ist typischerweise die doppelte Zykluszeit der Sicherheitssteuerung.

### Worst-Case-1 Reaktionszeit

Die Worst-Case-1 Reaktionszeit $t_{wc1}$ ist als die Zeit definiert, die die Komponenten mit ihrer Funktion, auch beim Vorhandensein **eines** Fehlers in einer Komponente, maximal benötigen, um zu reagieren.
Dies entspricht der Definition der „safety function response time“ (*SFRT*), wie sie auch in den Spezifikationen von PROFIsafe[86] und CIP-Safety[87] verwendet wird.
Die Berechnung der Worst-Case-1 Reaktionszeit basiert auf der maximalen Reaktionszeitberechnung. Für den Ort, an dem sich der Fehler auswirkt, wird der maximale Aufschlag für die Worst-Case Reaktionszeit der betreffenden Komponente zur maximalen Reaktionszeit der gesamten Kette hinzu addiert.

Die Worst-Case1 Reaktionszeit der Kette ist:

$$t_{wc1} = \{\sum t_{max-x}\} + max\{t_{wc-x} - t_{max-x}\} \qquad (3.2)$$

mit *x* aus *{ in, com-in, logic, com-out, out}*

[86] [PROFIsafeV2]

[87] [CIP5]

Im Unterschied zu den $t_{max}$ Werten, betrachten die $t_{wc1}$ Werte auch die Reaktionszeit beim Vorhandensein eines Fehlers.
Damit die Betrachtung der Worst-Case-1 Reaktionszeit $t_{wc1}$, z.B. wie in PROFIsafe[86] und CIP-Safety[87], vorausgesetzt werden kann, darf in der zu betrachtenden Zeit $t_{wc1}$ nur mit einem Fehler in der Verarbeitungskette gerechnet werden müssen.
Damit diese Sichtweise Bestand hat, ist es notwendig zu fordern, dass der eine Fehler sich innerhalb der Worst-Case-1 Reaktionszeit der Sicherheitsfunktion nur auf eine Komponente auswirkt. Dies wird leider in den Spezifikationen von PROFIsafe[86] und CIP-Safety[87] nicht erwähnt und stelle somit ein Problem dar.
Den unterlagerten, nicht sicherheitsgerichteten Komponenten und Netzen dürfen aus sicherheitstechnischer Sicht keine Eigenschaften, wie z.B. die symmetrische Geschwindigkeit des Nachrichtentransports oder maximale Verzögerungszeiten unterstellt werden.
Diese Eigenschaften können sich, auch wenn sie zum Zeitpunkt der Inbetriebnahme gegeben waren, durch Sicherheitsmechanismen unentdeckt, im Laufe des Betriebs verändern. Dieser problematische Ansatz wird bei PROFIsafe zur Bestimmung der *SFRT* verwendet[86]. Hier werden die maximalen Reaktionszeiten von nicht sicherheitsgerichteten Kommunikationsanschaltungen, von Geräte internen, nicht sicherheitsgerichteten Übertragungstechniken[88] und den unterlagerten Standard-Transportsystemen PROFINET und PROFIBUS-DP, mit in die Berechnung der *SFRT* einbezogen.

### Worst-Case-2 Reaktionszeit

Kann es durch eine Fehlerursache zu Common-cause Effekten in mehreren oder gar allen Komponenten der Sicherheitsfunktion kommen, darf für die sicherheitstechnische Betrachtung die Definition der Worst-Case-1 Reaktionszeit nicht herangezogen werden.
Die Common-cause Effekte können sich je nach Umgebung und Aufbau, z.B. im Bereich von Spannungsversorgung, Erdpotential, Temperatur außerhalb des Betriebsbereichs, einstellen. Sehr wahrscheinlich ist auch, dass Netzwerkstörungen die sich auf die Kommunikation zwischen Eingangskomponente und Sicherheitssteuerung auswirken, auch auf die Kommunikation zwischen Sicherheitssteuerung und Ausgabekomponente Einfluss haben. Erschwerend kommt hierbei hinzu, dass Netzwerkstörungen oft transienter Natur sind (elektromagnetische Störungen, schadhafte Kommunikationskabel, Vibrationen mit dafür ungeeigneten Steckverbindern) und somit mehrmals innerhalb der Worst-Case Reaktionszeit auftreten.
Die Worst-Case-2 Reaktionszeit $t_{wc2}$ ist als die Zeit definiert, die eine Sicherheitsfunktion, auch beim Vorhandensein von einem oder mehr Fehler, in einer oder mehreren Komponenten der Kette, maximal benötigt, um auf einen Signalwechsel am Eingang oder auf einen Fehler in der Kette zu reagieren.
Die Worst-Case-2 Reaktionszeit der Kette ist:

$$t_{wc2} = t_{wc\text{-}in} + t_{wc\text{-}com\text{-}in} + t_{wc\text{-}logic} + t_{wc\text{-}com\text{-}out} + t_{wc\text{-}out} \qquad (3.3)$$

### Safety Function Response Time

Die IEC 61784-3 definiert als maximale sichere Reaktionszeit die „safety function response time“ (*SFRT*). Diese beschreibt die maximale Zeit, die eine Sicherheitsfunktion für eine Reaktion benötigt.
Eine Betrachtung, dass nur ein Fehler vorliegen darf, oder der Fehler sich nur in einer Komponente manifestiert, wird in der IEC 61784-3 nicht weiter ausgeführt. Die Definition der IEC 61784-3 muss daher mit der Definition der Worst-Case-2 Reaktionszeit gleichgesetzt werden. Ausnahme stellen Übertragungssysteme und Umgebungen dar, in denen zwei oder mehr Fehler mit hinreichender Wahrscheinlichkeit ausgeschlossen werden kann.

[88] z.B. ein Backplane-Bus innerhalb der Geräte

## Modellprotokoll

Möchte man die sicherheitstechnisch relevante, garantierte Reaktionszeit ermitteln, so müssen folgende Szenarien analysiert werden:

1. Szenarien der Reaktionszeit, wie lange es dauert, bis ein Signalwechsel zum ungünstigsten Zeitpunkt eine Reaktion am Ausgang der Ausgabekomponente bewirkt (fehlerfreier Fall).
2. Das zuvor beschriebene Szenario, zusätzlich erweitert für das Eintreten von Fehlern nach dem oben beschriebenen Signalwechsel, bis zur Wirkung am Ausgang.

Betrachtet werden müssen für das erste Szenario (fehlerfreier Fall) auch jene Ereignisse, die von den Sicherheitsfunktionen des Kommunikationsprotokolls nicht erkannt werden. Da diese Klasse von Ereignissen nicht erkannt wird, darf nicht davon ausgegangen werden, dass sie nicht zusammen mit den Fehlern des zweiten Szenarios auftreten.

Dies kann zum Beispiel die Verzögerung von Nachrichten sein, die jedoch noch rechtzeitig vor dem Ablaufen der Überwachungszeit bei einem Knoten eintreffen.

Zur weiteren Betrachtungen der Reaktionszeiten wird hier ein Modell eines sicherheitsgerichteten Kommunikationsprotokolls definiert, soweit es zur zeitlichen Betrachtung erforderlich ist. Die Regeln des Protokolls sind wie folgt definiert (Abbildung 3.2):

1. Das Protokoll definiert die Rollen eines Masters und eines Slaves.
2. Der Master sendet an einen Slave eine Nachricht, die einen Monotoniezähler enthält. Der Slave beantwortet die Master Nachricht mit dem empfangenen Monotoniezähler des Masters. Dies gilt für Slaves in der Rolle der Eingangskomponente und in der Rolle der Ausgabekomponente gleichermaßen.
3. Der Abstand zweier aufeinander folgender, korrekt empfangener Nachrichten muss kleiner gleich $t_{com}$ sein. Der Einfachheit halber wird die Überwachungszeit $t_{com\text{-}in}$ zwischen Eingangskomponente und Sicherheitssteuerung und $t_{com\text{-}out}$ zwischen Sicherheitssteuerung und Ausgabekomponente gleich $t_{com}$ gesetzt.
4. Erkennt einer der Slaves eine Verletzung der Regeln, stellt er die Kommunikation zur Gegenseite ein. Bei der Ausgangskomponente erfolgt dann zusätzlich auch die Sicherheitsreaktion an ihren Ausgängen.
5. Erkennt der Master eine Verletzung der Regeln der Eingangskomponente, so sendet er, verarbeitet durch die Logik, die dazu passenden Werte an die Ausgabekomponente.
6. Erkennt ein Master eine Verletzung der Regeln durch die Ausgabekomponente, so stellt er die Kommunikation zu dieser ein.

Bei diesem Modell kann der Master mit mehreren Slaves zusammenarbeiten. Die Regeln sind je Slave anzuwenden. So ist im einfachsten Fall ein Slave eine Eingangskomponente und ein Slave eine Ausgabekomponente und der Master die Logik-Verarbeitung in einer Sicherheitssteuerung.

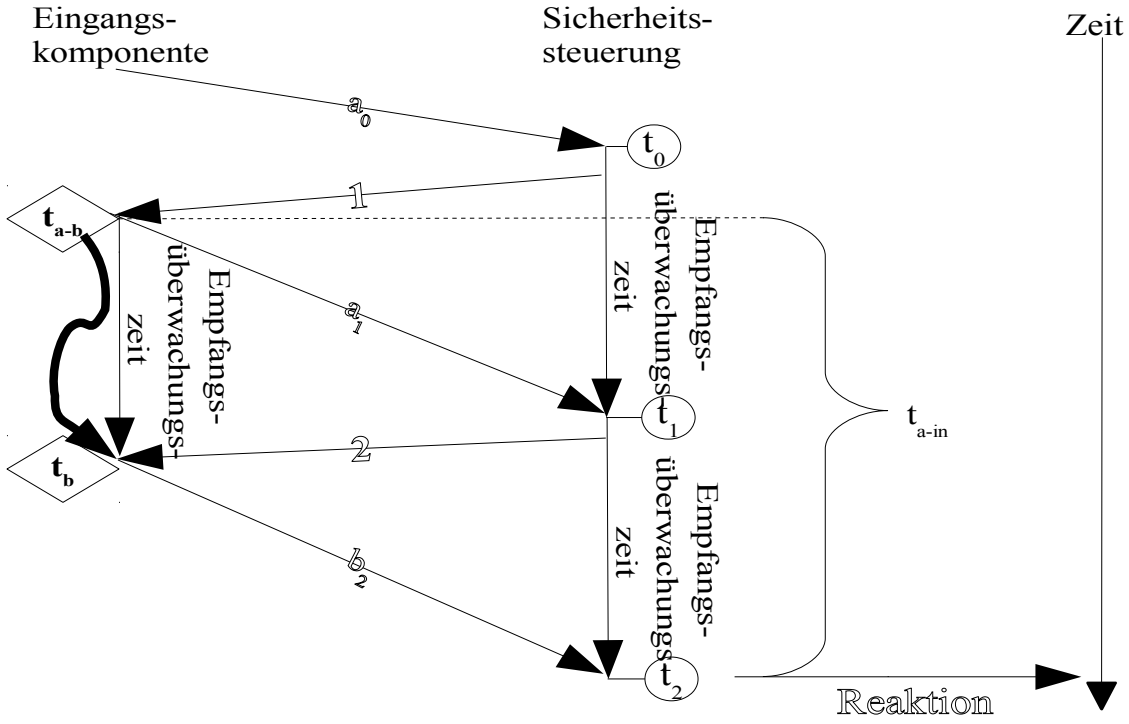


*Abbildung 3.2: Reaktionszeit der Sicherheitssteuerung auf Signalwechsel am Eingang*

Regeln für die Beherrschung von Nachrichtenverlusten werden hier nicht betrachtet, da sie einerseits oft von unterlagerten Schichten realisiert werden und zum Anderen die Reaktionszeiten, insbesondere die Worst-Case Zeiten, nicht beeinflussen.

Das Modell selbst entspricht in seinem Kern diversen, in der Praxis eingesetzten Protokollen, PROFIsafe sei hier nur stellvertretend genannt. PROFIsafe reagiert zwar auf diverse Regelverletzungen nicht mit dem Einstellen der Kommunikation, sondern es werden „Fail-Safe-Values“ übertragen, bzw. angenommen und ein neuer Anlauf der Verbindung wird erwartet, doch entspricht das Verhalten semantisch dem oben beschriebenen Modellprotokoll.

Der Nachrichtenfluss des Modellprotokolls und die daraus resultierende maximale Reaktionszeit ist im fehlerfreien Fall, wie folgt in Abbildung 3.3 definiert:

1. Zum Zeitpunkt $t_0$ empfängt die Sicherheitssteuerung die Antwort auf eine von ihr zuvor verschickte Anfrage (nicht dargestellt). Die Sicherheitssteuerung zieht zu diesem Zeitpunkt ihren Überwachungs-Timer erneut mit $t_{com}$ auf.
2. Nach $t_0$ sendet die Sicherheitssteuerung eine Anfrage (Monotoniezähler = 1) an die Eingangskomponente.
3. Zum Zeitpunkt $t_{a-b}$ empfängt die Eingangskomponente die Anfrage und beantwortet sie mit dem Signalwert a und dem Monotoniezähler = 1. Genau zu diesem Zeitpunkt wechselt gleichzeitig das Eingangssignal von *a* nach *b*.
4. Zum Zeitpunkt $t_1$ empfängt die Sicherheitssteuerung die Nachricht mit Monotoniezähler = 1 und dem nunmehr veralteten Signalwert *a*.

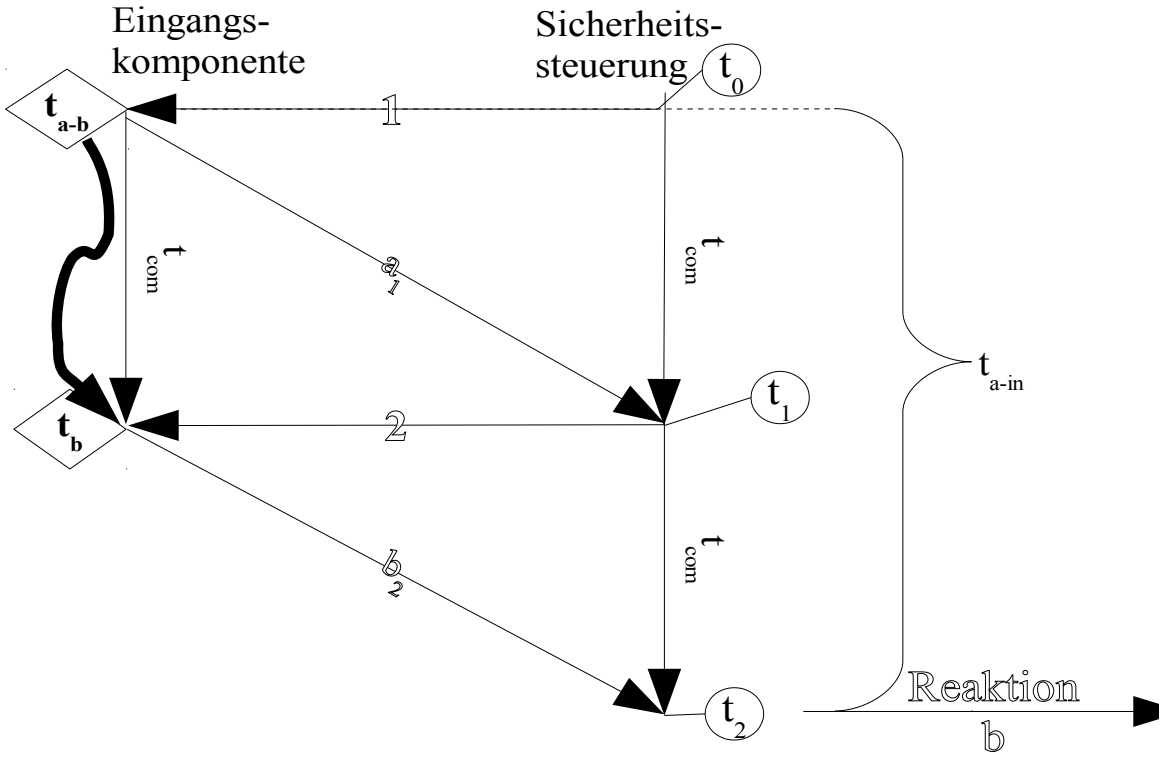


*Abbildung 3.3: Maximale Reaktionszeit der Sicherheitssteuerung auf Signalwechsel am Eingang*

5. Nach $t_1$ sendet die Sicherheitssteuerung eine Anfrage (Monotoniezähler = 2) an die Eingangskomponente.
6. Zum Zeitpunkt $t_b$ empfängt die Eingangskomponente die Anfrage und beantwortet sie mit dem Signalwert b und dem Monotoniezähler = 2. Damit wird nun der Wert übertragen, der seit dem Zeitpunkt $t_{a\text{-}b}$ anliegt.
7. Zum Zeitpunkt $t_2$ empfängt die Sicherheitssteuerung die Nachricht mit Monotoniezähler = 2 und dem Signalwert *b*.
8. Nach dem Zeitpunkt $t_2$ kann die Sicherheitssteuerung den Signalwert b an ihrem Ausgang einstellen.

Die Zeit $t_{a\text{-}in}$ beschreibt die Verzögerung, mit der Signalwert *b* am Ausgang der Sicherheitssteuerung eingestellt wird.

Setzt man für

$$t_1 - t_0 = t_2 - t_1 = t_{com} \qquad (3.4)$$

ein, wobei $t_{com}$ die maximal zulässige Zeit für den Abstand zweier empfangener Nachrichten ist, so ergibt sich eine maximale Reaktionszeit ohne Fehler und ohne Berücksichtigung der internen Zeiten der Komponenten mit:

$$t_{a\text{-}in} = 2 \bullet t_{com} \qquad (3.5)$$

Damit diese ungünstige Reaktionszeit angenommen wird, muss man voraussetzen, dass die Übertragungsdauer der Nachrichten von Sicherheitssteuerung zur Eingangskomponente nahezu Null-Zeit und der Rückweg von Eingangskomponente zu Sicherheitssteuerung annähernd $t_{com}$ ist.

Das in der Abbildung 3.4 dargestellte Übertragungsverhalten wird durch die Regeln des Protokolls akzeptiert. Daher muss es bei der Modellbildung mit dem Back-Channel-Ansatz angenommen werden. Dies ist darin begründet, das sich auf Grund asymmetrischer Wegwahl im Netzwerk, für Hin- und

Rückweg, durchaus unterschiedliche Transportzeiten ergeben können. Zum Anderen können Transportsysteme ihre Eigenschaften hin zum unterstellten Verhalten ändern (Fehlerannahme), ohne dass dies sicherheitstechnisch aufgedeckt würde.[89]

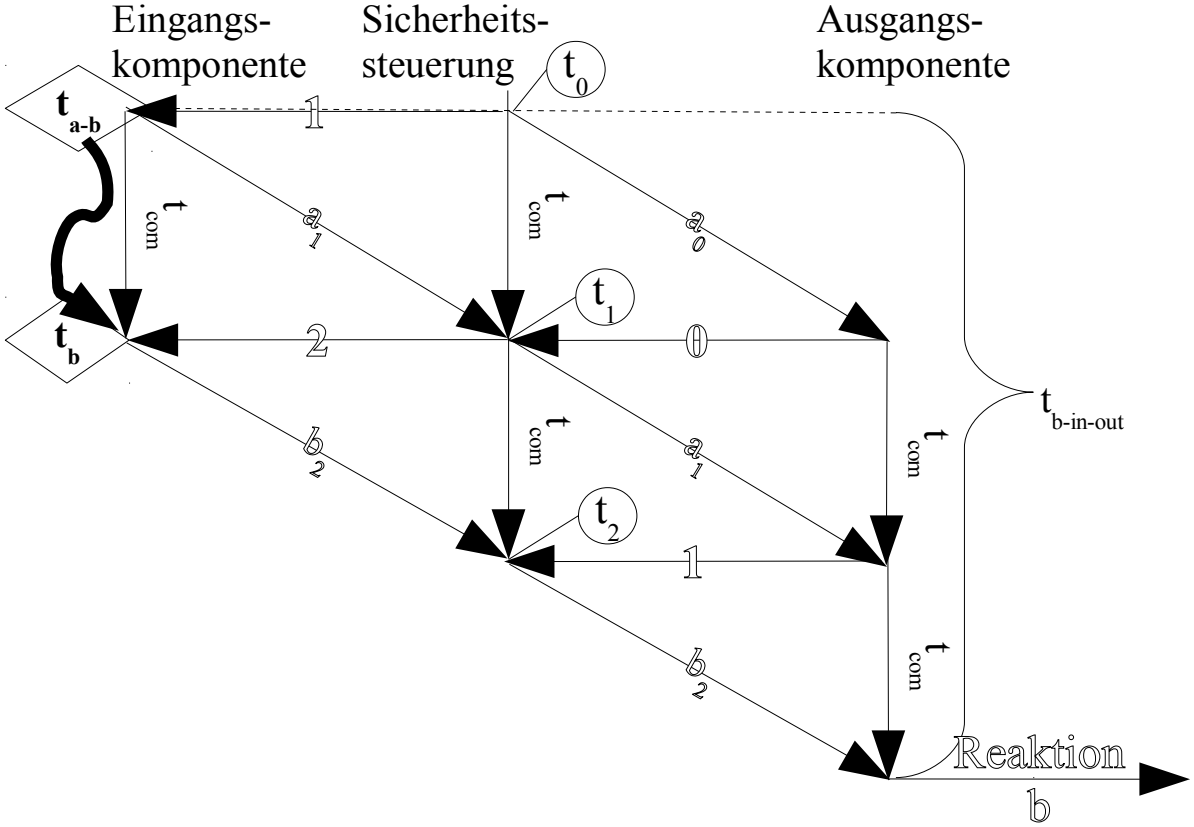


*Abbildung 3.4: Maximale Reaktionszeit auf Eingangssignalwechsel am Ausgang*

Erweitert man das obige Szenario um die Ausgabekomponente (siehe Abbildung 3.5), so erhält man die maximale Reaktionszeit von

$$t_{b\text{-}in\text{-}out} = 3 \bullet t_{com} \qquad (3.6)$$

Implementierungs-abhängige Eigenschaften können die maximale Reaktionszeit jedoch beeinflussen. Dies wird nach der Diskussion der Implementierungsmodelle im Rahmen der Worst-Case Reaktionszeiten nachgeholt.

Die Worst-Case-1/-2 Reaktionszeit des Modellprotokolls lässt sich ohne Berücksichtigung der Implementierungen für die Komponenten und die Transportsysteme nicht sinnvoll darstellen. Dies wird daher in einem späteren Abschnitt zusammen mit den Implementierungsvarianten nachgeholt.

---

[89] *Anmerkung: Die Symmetrie der Übertragungsdauer wird im Allgemeinen von den sicherheitsgerichteten Protokollen nicht überwacht und fällt in den Bereich der nicht sicherheitsgerichteten Elemente innerhalb des Transportkanals. Die Übertragungsdauer ist dabei nicht nur vom eigentlichen Netzwerk, sondern auch von den internen Strukturen der Komponenten abhängig. Z.B. beeinflusst eine Kopfstation, die den Netzwerkanschluss für eine sichere Eingangskomponente zur Verfügung stellt und zu dieser über einen Geräte-internen Bus kommuniziert, die Übertragungsdauer der sicherheitsgerichteten Nachrichten.*
*Bei symmetrischer Übertragungsdauer berechnet sich die Reaktionszeit zu 1,5 • $t_{com}$. Falls die Übertragungsdauer genau umgekehrt wie beim ungünstigsten Fall ist, berechnet sich die Reaktionszeit zu 1 • $t_{com}$., d.h. die Übertragungsdauer der Nachrichten von der Sicherheitssteuerung zur Eingangskomponente benötigt nahezu $t_{com}$ und der Rückweg von der Eingangskomponente zur Sicherheitssteuerung annähernd Null-Zeit.*

## Implementierung und Kopplung von Kommunikation und Funktion

Die im vorherigen Abschnitt dargestellten Reaktionszeitgleichungen geben nicht alle Bestandteile für die Bewertung von Kommunikationsprotokollen vollständig wieder. Die fehlenden Zeiten sind typischerweise in den Angaben der herstellerspezifischen Zeiten mit enthalten.

Es fehlen die Zeitanteile für

1. das Alter der für die Kommunikation verfügbaren Signalwerte ($t_{in}$)
2. die Kopplung von Protokoll und Funktion. Funktion meint hier Eingabe-, Ausgabe- und Logikfunktion
3. die Implementierung des Kommunikationsprotokolls in der jeweiligen Komponente
4. die Verzögerung der Ausgabe von Signalwerten am physikalischen Ausgang ($t_{out}$)

Der Kopplung des Protokolls und der Funktion, z.B. der Logikverarbeitung, kommt eine wichtige Bedeutung zu. So definiert diese, welche Verzüge für die Verarbeitung eines Eingangsereignisses auftreten können und wie lange es dauert, bis darauf eine Reaktion am Ausgang erfolgt.

Die Kopplung präsentiert sich in der Regel als synchrone oder asynchrone Kopplung.

## Asynchrone Variante

Bei der asynchronen Variante wird die Protokollverarbeitung asynchron zur Logikverarbeitung der Sicherheitssteuerung durchgeführt. Diese Kopplungsvariante wird in der Sicherheitssteuerung i.A. nicht eingesetzt, da die dort übliche zyklische Logikverarbeitung durch die asynchrone Kopplung und das allgegenwärtige Abtasttheorem von Shannon, die Reaktionszeit im Vergleich zu einer synchronen Kopplung bei gleicher Hardware-Leistung verschlechtert.
Dabei wird hier ausdrücklich das sicherheitsgerichtete Protokoll betrachtet. Bei den unterlagerten Transportprotokollen ist eine asynchrone Kopplung zwischen sicherheitsgerichtetem Protokoll und Transportkanal durchaus üblich, wie die Beispiele PROFIsafe über PROFINET und PROFIBUS-DP und CIP-Safety über EtherNet/IP zeigen[90].

## Synchrone Kopplung – Variante 1

Bei der synchronen Kopplung der Variante 1 werden über das Protokoll die Eingangswerte in der Anfangsphase des Zyklusses der Sicherheitssteuerung vom Eingangsmodul mittels Kommunikationsprotokoll „abgefragt“, die Logikverarbeitung durchgeführt und schließlich die Ausgangswerte der Logik an das Ausgangsmodul über das Protokoll übertragen.
Setzt man maximale Zykluszeit der Sicherheitssteuerung gleich $t_{com}$, so ergibt sich daraus für die maximale Reaktionszeit

$$t_{b\text{-}in\text{-}out} = 2 \bullet t_{com} = 2 \bullet Sicherheitssteuerung\text{-}Zykluszeit_{max} = t_{wc1} \qquad (3.7)$$

Dies entspricht dem gewohnten Verhalten einer lokal arbeitenden Sicherheitssteuerung[91], die nach zwei Zyklen spätestens auf ein Ereignis am lokalen Eingang reagiert.[92]

---

[90] [PROFIsafeV2], [PROFINET], [PROFIBUS], [CIP5]

[91] Unter einer lokal arbeitenden Sicherheitssteuerung versteht man eine Sicherheitssteuerung, die direkt angeschlossene Eingänge und Ausgänge verwendet.

[92] Anmerkung: Empfängt die Sicherheitssteuerung in ihrer Input-Phase keine Nachricht von der Eingangskomponente, so wertet sie dies als Fehler und kann über die Logik eine entsprechende Reaktionen mittels Nachricht an die Ausgangskomponente erzeugen. Dazu ist es jedoch notwendig, dass die Dauer der Input-Phase von der Sicherheitssteuerung zeitlich überwacht wird. Damit erreicht die Sicherheitssteuerung beim Ausbleiben der Antwort der Eingangskomponente, das sie keine Überschreitung der maximal zulässigen Sicherheitssteuerung Zykluszeit verursacht.
Ebenso muss die Sicherheitssteuerung die Dauer ihrer Output-Phase überwachen. Wenn die zugestandene Zeit überschritten wird, muss die Sicherheitssteuerung im nächsten Sicherheitssteuerung Zyklus eine entsprechende Reaktion auslösen.

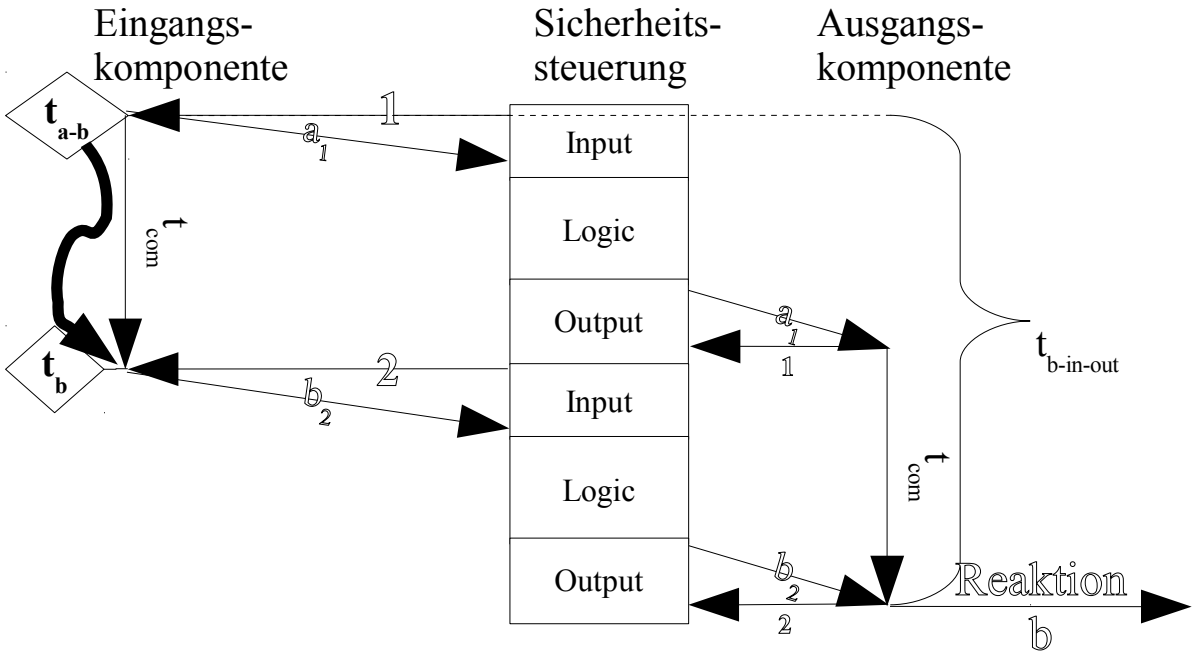


*Abbildung 3.5: Reaktionszeiten der synchronen Kopplung – Variante 1*

**Synchrone Kopplung – Variante 2**

Bei der synchronen Kopplung der Variante 2a wird die Protokollverarbeitung, wie bei Variante 1, in der Anfangs- (Input-) und Endphase (Output-Phase) des Zyklus der Sicherheitssteuerung durchgeführt, jedoch wird dabei nicht auf die Protokolldaten der Ein- und Ausgangsmodule gewartet. Diese Daten werden asynchron zu den Zyklen der Sicherheitssteuerung empfangen.
In Abbildung 3.6 ist beispielhaft die Ankunft der Nachricht a1 von der Eingangskomponente dargestellt, nachdem die Input-Phase der Sicherheitssteuerung schon beendet wurde. Daher wird die Nachricht a1 erst in der nächsten Input-Phase der Sicherheitssteuerung verarbeitet und der nachfolgenden Logikverarbeitung zugeführt. Die Antwortnachricht 1 der Ausgabekomponente kommt in der Abbildung ebenfalls erst nach dem Ende der Output-Phase der Sicherheitssteuerung an und wird, wie schon die Nachricht a1, erst in der nächsten passenden Phase verarbeitet.
Der Vorteil dieser Variante gegenüber Variante 1 ist, dass der lokale Sicherheitssteuerungszyklus schneller abgearbeitet werden kann, da die Wartezeiten auf die Antworten der Ein- und Ausgangskomponenten entfallen. Ob dies für die Worst-Case-2 Reaktionszeit von Vor- oder Nachteil ist, hängt von den Verhältnissen der Überwachungszeiten der Ein- und Ausgangskomponenten und deren Zykluszeit und der Verarbeitungszeit des Zyklusses der Sicherheitssteuerung ab.
Ein weiterer Vorteil ist, dass Komponenten mit unterschiedlichen Überwachungszeiten kombiniert werden können und eine allgemeine Reaktionszeit daraus ableitbar ist.
Weitere verwandte Untervarianten dieser Kopplungsart können ebenfalls zur Anwendung kommen. Diese unterscheiden sich dadurch, dass

- 2b: die Protokollverarbeitung für Ein- und Ausgangskomponenten nur in der Input-Phase der Sicherheitssteuerung stattfindet.
- 2c: die Protokollverarbeitung für Ein- und Ausgangskomponenten nur in der Output-Phase der Sicherheitssteuerung stattfindet.
- 2d: die Protokollverarbeitung für Ein- und Ausgangskomponenten nur in der Input-Phase empfange Nachrichten verarbeitet und Nachrichten nur in der Output-Phase versendet.

Allen Untervarianten gemeinsam ist die feste Reihenfolge bezüglich der Logikverarbeitung.

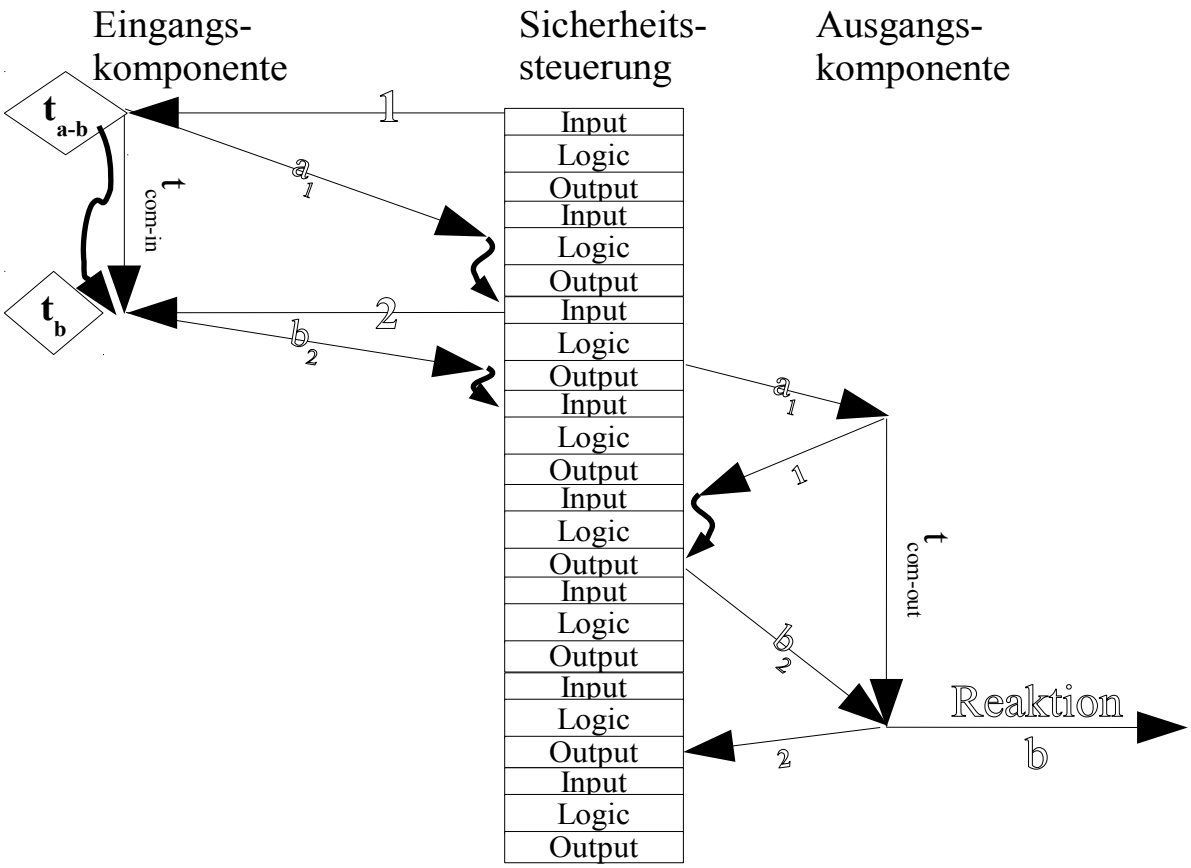


*Abbildung 3.6: Synchrone Kopplung – Variante 2a*

Nachfolgend wird die einfachste Variante 2a für weitere Untersuchungen verwendet.

Die synchronen Kopplungsvarianten 2 (2a,2b,2c,2d) stellen eine sehr häufig anzutreffende Implementierung von Sicherheitsprotokollen dar. Sie ist für PROFIsafe bei Siemens – S7 oder safe**ethernet** bei HIMA – HIMatrix[93] anzutreffen.

Die Varianten, die am Beispiel der Funktion Logikverarbeitung der Sicherheitssteuerung aufgezeigt wurden, können sinngemäß auf die Ein- und Ausgangskomponenten übertragen werden. Da nicht immer eine gleichartige Implementierung zum Einsatz kommt, sind dabei im konkreten Anwendungsfall das Zusammenspiel zahlreicher Varianten zu betrachten.

In der Kombination mit der spezifischen Implementierung der zyklischen Protokollverarbeitung ist ersichtlich, dass bei beiden synchronen Kopplungsarten die Ungenauigkeit des Erkennens des Ablaufens der Überwachungszeit, nicht zusätzlich zum Tragen kommen muss, da es sich durch die maximale Reaktionszeit der Funktion[94] kompensieren lässt.

Wird hingegen in diesem Szenario die asynchrone Variante gewählt, so muss die Präzision der Ablauferkennung der Reaktionszeit hinzugerechnet werden.

Die synchronen Kopplungsvarianten 1 und 2 unterscheiden sich aus zeitlicher Sicht dadurch, dass bei Variante 1 auf die Protokollreaktion der Gegenstelle gewartet wird, dafür aber ein Eingangsdatum erhalten kann, dass nicht älter als der Beginn des laufenden Zyklusses der Sicherheitssteuerung ist. Dabei sind die Auswirkungen des Kopplungsmodells und der Implementierung der Gegenstelle nicht berücksichtigt, die sich auch auf das Alter des Datums auswirken.

---

93 [safeethernet]

94 Eingangs-, Logik-, Ausgangsverarbeitung

Bei der Implementierung kommt einerseits die Auswahl von Protokollvarianten, wie z.B. Application-triggered oder Link-triggered bei CIP-Safety[95] und andererseits das Zeitverhalten der spezifischen Protokollverarbeitung zum Tragen.

Typisch für die Protokollverarbeitung ist das zyklische Ausführen der Protokoll-Funktion, insbesondere im Zyklus der Sicherheitssteuerung[96]. Diese zyklische Ausführung ist in der Regel zeitüberwacht und gibt die Präzision vor, mit der das Empfangen einer Nachricht bzw. das Ablaufen einer Protokoll-spezifischen Überwachungszeit, erkannt werden kann.

Ein alternativer, eher selten genutzter Ansatz, ist die Zeitüberwachung durch eine relative präzise Einheit, z.B. einem HW-Timer. Dieser kann das Ablaufen der überwachten Zeit früher erkennen, als es bei obiger, zyklischer Ausführungsmethode im Worst-Case Fall anzunehmen ist.

Der Grund, warum dieser Ansatz selten genutzt wird, liegt darin, dass die Logikfunktion, die für die Reaktion auf das Ablaufen erforderlich ist, i.A. zyklisch ausgeführt wird. Daher wird die Reaktionszeit auf die Zeitüberwachung aus Sicht der gesamten Sicherheitsfunktion nicht verbessert (siehe Kopplungsvarianten).

Ein weiterer Aspekt der Protokollimplementierung ist die Asynchronität der Protokollabläufe verschiedener Verbindungen zueinander. Sobald die Implementation in der Sicherheitssteuerung eine Zyklus-übergreifende Empfangszeitüberwachung, z.B. mit Variante 2, realisiert, befinden sich die Protokoll-Instanzen für die Eingangs- und die Ausgangsverbindung in unterschiedlichen, zueinander nicht gekoppelten Verarbeitungszuständen.

Das führt zum Beispiel dazu, dass eine Nachricht von der Sicherheitssteuerung an die Ausgangskomponente geschickt wird, obwohl in diesem Zyklus der Sicherheitssteuerung keine Nachricht von der Eingangskomponente empfangen wurde. Die Auswirkung dieser Art der Implementation wird in den nachfolgenden Abschnitten der Reaktionszeitberechnung betrachtet.

Nachfolgend wird wiederum das Modellprotokoll herangezogen. Dabei wird eine Kopplungsvariante 2a und damit gleichbedeutend eine zyklische Protokollverarbeitung angenommen. Für die Kopplungsvariante und Implementierung der Eingangs- / Ausgangskomponenten wird vereinfacht eine zur Sicherheitssteuerung identische Implementierung angenommen. Die Überwachungszeiten werden zwecks einfacherer Darstellung ebenfalls als gleich angenommen

$$t_{com} = t_{com\text{-}in} = t_{com\text{-}out}. \qquad (3.8)$$

Die Resultate bleiben jedoch auch bei unterschiedlichen Überwachungszeiten gültig, wenn man jeweils $t_{com\text{-}in}$ und $t_{com\text{-}out}$ einsetzt[97].

Für die maximale Reaktionszeit mit gekoppelter Eingangskomponente und Sicherheitssteuerung wird die Zeit dargestellt, die nach dem Eintreten eines Wertewechsels am Eingang der Eingangskomponente von *a* nach *b* vergehen kann, bis eine Reaktion am lokalen Ausgang der Sicherheitssteuerung darauf erfolgt.

Dabei wird hier weiterhin vom fehlerfreien, jedoch ungünstigstem Fall ausgegangen.

Der ungünstigste Fall liegt dann vor, wenn die Nachrichten vom Eingangsmodul immer unmittelbar vor Ablaufen der Empfangszeitüberwachung empfangen (verarbeitet) werden.

Für den ungünstigsten Fall gilt weiterhin, dass sich für die Nachricht $a_1$ der Signalwechsel von *a* nach *b* „gerade“ nicht mehr auswirkt. Damit wird der Wert *b* um die Zeit von maximal $t_{com}$ verzögert, bevor er in Nachricht $b_2$ erstmalig versendet wird (siehe Abbildung 3.7).

---

[95] [CIP5]

[96] Variante 2

[97] *Anmerkung*: Die Überwachungszeit $t_{com\text{-}in}$, die die Sicherheitssteuerung für die Eingangskomponente verwendet und die Überwachungszeit, die die Eingangskomponente für die Sicherheitssteuerung verwendet sind immer gleich. Für die Ausgangskomponente und deren Überwachungszeiten $t_{com\text{-}out}$ gilt dies ebenfalls.

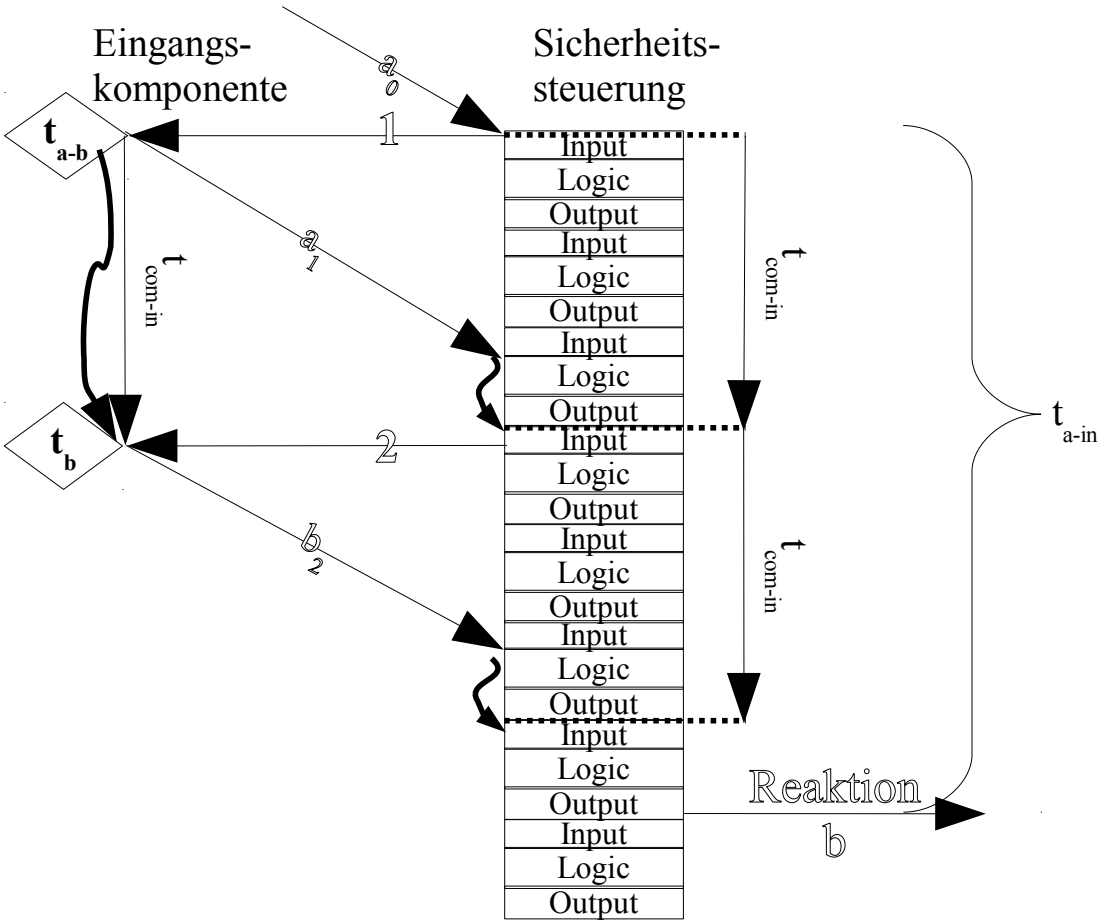


*Abbildung 3.7: Maximale lokale Reaktionszeit bei synchroner Kopplung – Variante 2a*

Das wesentliche Ergebnis, das hier zum Ausdruck kommt, ist, dass bei einer Empfangsüberwachungszeit $t_{com}=t_{com\text{-}in}$ und Sicherheitssteuerung-Zykluszeit $t_{logic}$ die Reaktion auf den Signalwechsel von *a* auf *b* ohne Existenz eines Fehlers im schlechtesten Fall

$$t_{a\text{-}in} = 2 \bullet t_{com} + t_{logic} \qquad (3.9)$$

ist.

Die mögliche Implementierungs- bedingte Ungenauigkeit bei der Empfangszeitüberwachung von einer Sicherheitssteuerung-Zykluszeit kann bei einer idealen Eingangskomponente nicht im ordnungsgemäßen Betrieb auftreten, da die Empfangszeitüberwachung bei der Eingangskomponente für den Abbruch der Verbindung sorgen würde. In diesem Fall würde aus Sicht der Eingangskomponente die Sicherheitssteuerung ihre Sendung „2“ mit einem Abstand zur Sendung „1“, der größer als die Empfangsüberwachungszeit ist, versenden.

Ist die Eingangskomponente nicht ideal, d.h. unterstellt man bei ihr z.B. das gleiche Implementierungsmodell 2a, so kann eine Ungenauigkeit der Empfangszeitüberwachung in der Kommunikationsbeziehung in der Größenordnung der minimalen Zykluszeiten

$$t_u = min \{ t_{cyc\text{-}in}, t_{logic} \} \qquad (3.10)$$

der beiden, Sicherheitssteuerung und Eingangskomponente, unentdeckt bleiben. Dabei ist $t_{cyc\text{-}in}$ die Zykluszeit der Eingangskomponente. Dies ist aus Sicht der Implementierungsvariante 2 kein Fehler,

sondern wird als regulärer Protokollbetrieb betrachtet, da dieses zeitliche Verhalten in der Reaktionszeit der Sicherheitsfunktion berücksichtigt wird.

Um die mögliche Ungenauigkeit erweitert, ergibt sich somit als maximale Reaktionszeit

$$t_{a\text{-}in} = 2 \bullet t_{com} + t_{logic} + min \{ t_{cyc\text{-}in}, t_{logic} \} \quad (3.11)$$
$$= 2 \bullet t_{com} + t_{logic} + t_u$$

Nachdem Kommunikationsstrecke und Sicherheitssteuerung-Logik – Protokollkopplung berücksichtigt wurden, könnte man meinen, dass noch der Anteil fehlt, der durch die Eingangskomponente und deren Mechanismen zur Werteermittlung, verursacht wird. Verwendet diese eine asynchrone Kopplungsart, so wäre dies[98]

$$2 \bullet t_{cyc\text{-}in} \quad (3.12)$$

Ohne eine spezifische Kopplungsart müsste man diesen Term hinzu addieren. Wird jedoch der effiziente Weg gewählt, dass die Protokoll-Eingangsverarbeitung der Eingangskomponente, gefolgt von deren EA-Verarbeitung und anschließender Protokoll-Ausgangsverarbeitung innerhalb ihres Zyklusses erfolgt (Variante 2d), so kann der Verzug von $2 \bullet t_{cyc\text{-}in}$ entfallen. Dies liegt daran, dass im schlechtesten Fall der Signalwechsel a-b verpasst wird, wenn er zum Zeitpunkt des Absendens von Nachricht 1 erfolgt und diese keine Laufzeit auf dem Netz hat. Somit ist der maximale Verzug im Term $2 \bullet t_{com}$ bereits enthalten.

Hier nicht weiter betrachtet bleiben die Verzögerungen der Werteermittlung durch vorgeschaltete Filter oder andere das Eingangssignal verzögernde Komponenten.

Nachfolgend wird der meist anzutreffende Fall der maximalen Reaktionszeit analysiert, bei dem eine dezentrale Eingangskomponente das Eingangsdatum ermittelt, an die Sicherheitssteuerung überträgt, das Eingangsdatum in der Sicherheitssteuerung verarbeitet wird und diese das Ergebnis danach an die Ausgangskomponente überträgt und die Ausgangskomponente das Ergebnis an ihren Ausgängen einstellt.

In der Abbildung 3.8 ist die Betrachtung der Eingangsdatenverarbeitung gegenüber Abbildung 3.7 unverändert geblieben. Die in der Abbildung dargestellten Überwachungszeiten $t_{com\text{-}in}$ und $t_{com\text{-}out}$ werden zwecks vereinfachter Schreibweise nachfolgend beide mit $t_{com}$ bezeichnet.

[98] Anmerkung: Der schlechteste Fall, einer asynchronen Kopplungsart des EA-Zyklus der Eingangskomponente zur Protokollverarbeitung und das Verfügbarmachen eines erkannten Eingangsdatums für die Protokollverarbeitung zum Ende des EA-Zyklusses ergibt $t_{cyc\text{-}in}$. Wird der Signalwechsel am Anfang des EA-Zyklusses verpasst, so dauert es wiederum $t_{cyc\text{-}in}$ bis das Datum erkannt wurde.

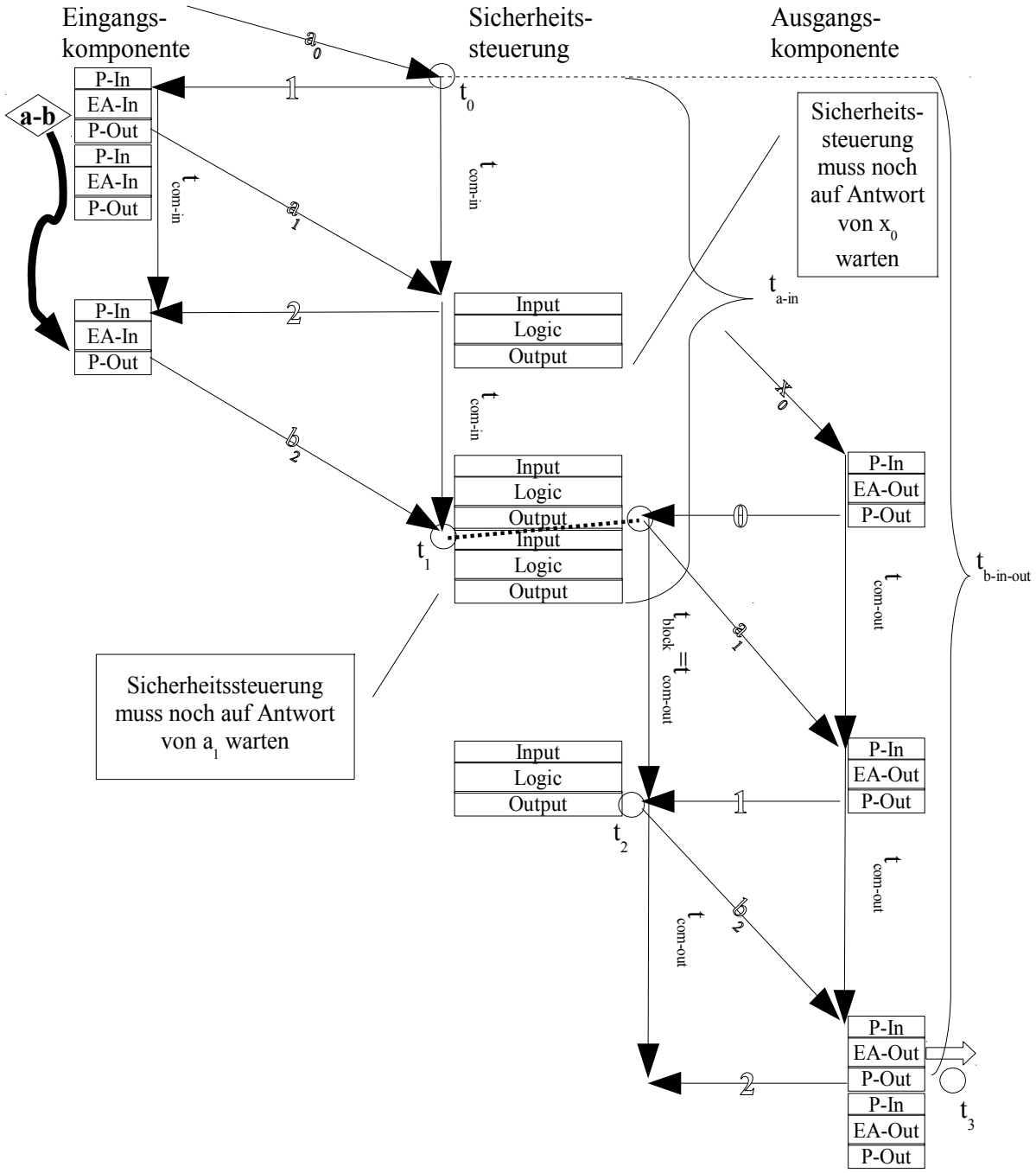


*Abbildung 3.8: Maximale Reaktionszeit Input/Sicherheitssteuerung/Output mit Sicherheitssteuerung Kopplung 2a*

Für den Anteil der Reaktionszeit der Übertragung von der Sicherheitssteuerung zur Ausgangskomponente ergibt sich bei Annahme der Kopplungsvariante 2d der Ausgangskomponente, d.h. der oben vorgestellten Reihenfolge von Protokoll-Input-Verarbeitung, EA-Verarbeitung und Protokoll-Output-Verarbeitung, im einfachsten fehlerfreien Fall $t_{com}$. Die Zeit für den Zyklus der Ausgabekomponente muss man im fehlerfreien Fall nicht hinzurechnen, da ansonsten die Zeit für die Antwort an die Sicherheitssteuerung nicht ausreichen würde. Wie bei der Eingangskomponente kommt hier Implementations- bedingt eine Ungenauigkeit in der Größenordnung der minimalen Zykluszeiten $t_u = min\ \{ t_{cyc\text{-}out}, t_{logic} \}$ der Sicherheitssteuerung und Ausgangskomponente für die Fehlerbetrachtung später hinzu. Dabei ist für die Eingangskomponente ebenfalls die Kopplungsart 2d genutzt worden.

Für die weitere Analyse der Ausgangsdatenverarbeitung muss jedoch eine spezielle Situation zum Zeitpunkt $t_1$ des Empfangs von Datum $b_2$ berücksichtigt werden.

Allgemein kann die Sicherheitssteuerung ein neues Ausgangsdatum solange nicht an die Ausgangskomponente senden, solange eine vorherige Nachricht, in diesem Fall $a_1$, noch nicht bestätigt wurde.

**Das Senden ist für diese Zeit blockiert!**

Der ungünstigste Fall stellt sich dann ein, wenn im Zyklus unmittelbar vor dem Zyklus, in dem $b_2$ empfangen wird, noch das vorherige Datum, in diesem Fall $a_1$, versendet wird und nun im schlechtesten Fall noch die zulässige Überwachungszeit abzuwarten ist, bis die Bestätigung 1 auf $a_1$ empfangen und dann „endlich“ $b_2$ versendet werden kann.

Die maximale Zeit für die Blockade berechnet sich aus:

$$t_{block} = t_{com} - t_{logic\text{-}min} \qquad (3.13)$$

Diese Blockade ergibt sich neben den Regeln des Protokolls aus dem Sachverhalt, dass die Protokollabwicklung von Eingangskomponente zu Sicherheitssteuerung und von Sicherheitssteuerung zu Ausgangskomponente nicht synchronisiert sind. Da die minimale Ausführungsdauer der Logik $t_{logic\text{-}min}$ im Allgemeinen nicht betrachtet wird, beziehungsweise der Zeitpunkt der Protokollverarbeitung innerhalb eines Zyklusses sicherheitstechnisch nicht garantiert werden kann, muss man sie häufig zu 0 setzen. Dies wird nachfolgend angenommen und somit ist $t_{block} = t_{com}$.

*Einschub:*

*Das Problem zwei Aktivitäten zeitlich zueinander in Beziehung zu setzen, wenn sie in zwei aufeinander folgenden Zyklen ablaufen, stellt sich wie folgt dar.*

*a) Sicherheitstechnisch gibt es in der Regel keine Mechanismen, die die Ausführung einer Aktivität in einem definierten Abstand vom Zyklusbeginn garantieren. Die Aktivität startet meist so schnell als möglich, nachdem die vorhergehenden Aktivitäten im Zyklus ausgeführt wurden.*

*b) Die sicherheitstechnische Garantie bezieht sich darauf, dass der gesamte Zyklus eine maximale Dauer nicht überschreitet und dass die jeweilige Aktivität innerhalb eines jeden Zyklusses immer in der gleichen Reihenfolge ausgeführt wird.*

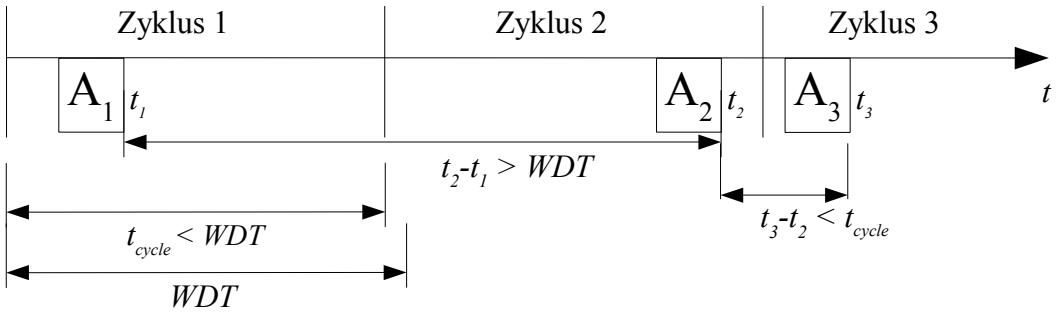


*Abbildung 3.9: Zeitliche Abstände gleicher Aktionen in 2 Zyklen*

*Die Abbildung 3.9 verdeutlicht, dass der Abstand zwischen einer Aktivität A in zwei aufeinander folgenden Zyklen 1 und 2 im Worst-Case bis zu zwei mal Zyklusüberwachungszeit WDT sein kann. Aus dem selben Grund kann der Abstand der Aktivitäten, hier $A_2$ zu $A_3$ jedoch auch kleiner als die minimale Zykluszeit $t_{cycle}$ sein.*

*Die Worst-Case Reaktionszeit der Gesamtheit der Zyklusaufgaben von 2 Zyklen < 2 • WDT ist damit nach wie vor gegeben, solange die Reihenfolge der Aktivitäten innerhalb des Zyklusses nicht verändert werden und dem EVA-Prinzip[99] entsprechen.*

*Aus diesen Gründen muss die minimale Zykluszeit bezüglich der Blockade in der Regel mit 0 abgeschätzt werden.*

Der Nachteil der Blockade bleibt auch dann bestehen, wenn die synchrone Kopplung 2 mit Varianten b), c) oder d) realisiert wird.

Die mögliche Ungenauigkeit der Empfangszeitüberwachungen im ungestörten Betrieb ergibt sich für die 3 Komponenten zu

$$t_u = min\ \{ t_{cyc\text{-}in},\ t_{cyc\text{-}out}, t_{logic}\} \tag{3.14}$$

In Summe ergibt sich die maximale Reaktionszeit ohne Fehler zu

$$t_{b\text{-}in\text{-}out} = 2 \bullet t_{com} + t_{block} + t_{com} + t_u \tag{3.15}$$

$$= 4 \bullet t_{com} + t_u$$

Man beachte, dass hierbei der Anteil $t_{logic}$, der noch in der Reaktionszeitberechnung nur von der Eingangskomponente zur Sicherheitssteuerung enthalten war, entfallen ist, da diese Zeit in $t_{block}$ bereits enthalten ist.

Im Gegensatz bei der Implementierungsunabhängigen Betrachtung der maximalen Reaktionszeit mit

$$t_{max} = t_{max\text{-}in} + t_{max\text{-}com\text{-}in} + t_{max\text{-}logic} + t_{max\text{-}com\text{-}out} + t_{max\text{-}out} \tag{3.16}$$

würde man für

$$t_{max\text{-}in} = 2 \bullet t_{cyc\text{-}in}$$

$$t_{max\text{-}com\text{-}in} = 2 \bullet t_{com\text{-}in}$$

$$t_{max\text{-}logic} = 2 \bullet t_{logic}$$

$$t_{max\text{-}com\text{-}out} = 2 \bullet t_{com\text{-}out}$$

$$t_{max\text{-}out} = 2 \bullet t_{cyc\text{-}out} \tag{3.17}$$

einsetzen und dann die maximale Reaktionszeit

$$t_{max} = 2 \bullet t_{cyc\text{-}in} + 4 \bullet t_{com} + 2 \bullet t_{logic} + 2 \bullet t_{cyc\text{-}out} \tag{3.18}$$

berechnen.

Dieser Wert wird mit der Implementierungsabhängigen Variante um

$$t_{delta} = 2 \bullet t_{cyc\text{-}in} + 2 \bullet t_{logic} + 2 \bullet t_{cyc\text{-}out} - t_u \tag{3.19}$$

unterschritten. Der im Allgemeinen große Zeitanteil von $2 \bullet t_{logic}$ kann somit, vorbehaltlich nachfolgender Fehlerannahmen, sicherheitstechnisch eingespart werden. Somit können weniger leistungsfähige Hardware-Ressourcen eingesetzt werden.

Die Berechnung der maximalen Reaktionszeit bei PROFIsafe[100] verfolgt einen etwas anderen Ansatz, der später im Kapitel zur Analyse von PROFIsafe diskutiert wird.

Zur Berechnung der Worst-Case Reaktionszeiten ist es erforderlich auch die möglichen Fehlerszenarien der sicherheitsgerichteten Kommunikation zu betrachten. Dies erfolgt in den folgenden Abschnitten.

---

[99] **E**ingabe, **V**erarbeitung, **A**usgabe

[100] [PROFIsafeV2]

Als Fehler wird weiter angenommen, dass die Sicherheitssteuerung mit ihrer Protokollverarbeitung (siehe Abbildung 3.8) mit dem Empfang von Nachricht $a_1$ und dem folgenden Senden von Nachricht 2 so spät ist, dass auch bei einer Übertragungsdauer von 0 ms von Sicherheitssteuerung zur Eingangskomponente, die Empfangszeitüberwachung bei der Eingangskomponente $t_{com\text{-}in}$ (nachfolgend $t_{com}$) für den erwarteten Empfang von Nachricht 2 überschritten wird. Als Reaktion darauf sendet die Eingangskomponente keine Antwort $b_2$ auf Nachricht 2.

Auftreten kann das verspätete Senden von Nachricht 2 durch die Implementierung in der Sicherheitssteuerung, die in der Input-Phase der Sicherheitssteuerung erst prüft, ob Nachricht $a_1$ empfangen wurde und diese Nachricht dann akzeptiert, bevor die Überwachungszeit $t_{com}$ verifiziert wird. Das heißt, es wird im schlechtesten Fall eine Nachricht bis zu einem Sicherheitssteuerung-Zyklus $t_{logic}$ nach dem Ablaufen von $t_{com}$ akzeptiert. Der Implementierung liegt das Prinzip der Zeit diskreten Verarbeitung zu Grunde, bei der nicht unterschieden werden kann, was in welcher chronologischer Reihenfolge zwischen den Zeitpunkten geschehen ist (siehe Anmerkungen zu $t_u$ oben).

Weiter wird nun die nächste Überwachungszeit $t_{com}$ auf der Sicherheitssteuerung für die Eingangskomponente im Rahmen der Verarbeitung von $a_1$ gestartet. Eigentlich müsste die Überwachungszeit beim Empfang der Nachricht $a_1$ gestartet werden, jedoch stellt die verspätete Verarbeitung derselben eben den hier betrachteten Fehlerfall dar. Das Ablaufen dieser Überwachung von $t_{com}$ wird schlechtesten Falls wiederum um einen Sicherheitssteuerung-Zyklus $t_{logic}$ verspätet erkannt.

Wiederum schlechtesten Falls[101] wird das Abschaltsignal innerhalb der Sicherheitssteuerung nach der Zeit $t_{logic}$ aktiviert.

Für diese Reaktionszeit und den weiterhin angenommen Signalwechsel $a - b$ ergibt sich für den schlechtesten Fall:

$$t_{error\text{-}in\text{-}a} = t_{com} + t_{logic} + (t_{com} + t_{logic}) + t_{logic} \qquad (3.20)$$
$$= 2 \bullet t_{com} + 3 \bullet t_{logic}$$

- Der erste Term $\boldsymbol{t_{com}}$ ist der zulässige Verzug für die Antwort auf Nachricht 1[102].
- Der zweite Term + $\boldsymbol{t_{logic}}$ ist der Verzug für die Empfangsverarbeitung von $a_1$ und das Versenden von Nachricht 2 ( → der eigentliche Fehler). $t_u$ ist hier nicht mehr zu berücksichtigen, da genau dessen Überschreitung durch $t_{logic}$ zum Fehler führt. Hierbei wird angenommen, dass $t_{logic} > t_{cyc\text{-}in}$ ist.
- Der dritte Term + **(** $\boldsymbol{t_{com}}$ + $\boldsymbol{t_{logic}}$**)** beschreibt das Ablaufen der Überwachungszeit und den Verzug der Erkennung des Ablaufens.
- Der vierte Term + $\boldsymbol{t_{logic}}$ stellt schlussendlich noch die Reaktion auf die abgelaufene Überwachungszeit dar, d.h. Ausführung der Logikfunktion und der lokalen Ausgabe des daraus resultierenden Abschaltsignals.

Erweitert man erläuterte Fehlerszenario um die Ausgabe des Abschaltsignals durch eine Ausgangskomponente, so ist auf die Reaktionszeit noch deren maximale Reaktionszeit zu addieren.

$$t_{error\text{-}a} = t_{error\text{-}in\text{-}a} + ( t_{block} + t_{com} ) - t_{logic} \qquad (3.21)$$
$$= ( 2 \bullet t_{com} + 3 \bullet t_{logic} ) + (t_{com} + t_{com}) - t_{logic}$$
$$= 4 \bullet t_{com} + 2 \bullet t_{logic}$$

Dabei wurde für den Ausgabeweg kein Fehler unterstellt. Der Verzug der lokalen Reaktion $t_{logic}$ wurde gegen gerechnet, da dieser in $t_{block}$ der Kommunikation zur Ausgangskomponente enthalten ist.[103]

---

[101] Reaktion auf das Erkennen erfolgt ganz am Ende des zulässigen Zyklusses

[102] Verzug der Eingangswert-Ermittlung $2 \bullet t_{cyc\text{-}in}$ wieder zu 0 ms gesetzt

Für das nächste zu betrachtende Szenario ergeben die folgende Fehlerannahmen identischen Reaktionszeiten (Basis ist Abbildung 3.8):

1. Nachricht 2 oder Nachricht $b_2$ gehen verloren, bzw. die Kommunikationsstrecke wird nach dem Empfang von $a_1$ und vor dem regulären Empfang von $b_2$ unterbrochen.
2. Die Eingangskomponente fällt in der Zeitspanne nach dem Versenden von $a_1$ und vor dem regulären Versenden von $b_2$ aus.

Aus diesen Fehlerannahmen resultiert eine Reaktionszeit $t_{error\text{-}in\text{-}b}$, ähnlich wie oben bei $t_{error\text{-}in\text{-}a}$. Unterstellt man, dass der zweite Term von Gleichung (3.20) ($+ t_{logic}$) = 0 ms ist und damit die Nachricht $a_1$ unmittelbar vor dem Ablaufen der Überwachungszeit $t_{com}$ empfangen wurde und dass die folgende Überwachungszeit für $b_2$ gestartet wird, so ergibt sich:

$$t_{error\text{-}in\text{-}b} = t_{com} + (t_{com} + t_{logic}) + t_{logic} + t_u \qquad (3.22)$$
$$= 2 \bullet t_{com} + 2 \bullet t_{logic} + t_u$$

Die Terme entsprechen denen beim ersten Fehlerszenario des Kapitels, nur dass das erste $t_{logic}$ entfallen ist und die maximale Ungenauigkeit $t_u$ der Empfangszeitüberwachung wieder hinzugefügt wurde.

Erweitert um die Ausgabe des Abschaltsignals durch eine Ausgangskomponente, ergibt sich die Reaktionszeit somit zu

$$t_{error\text{-}b} = t_{error\text{-}in\text{-}b} + (t_{block} + t_{com}) - t_{logic} \qquad (3.23)$$
$$= (2 \bullet t_{com} + 2 \bullet t_{logic}) + (t_{com} + t_{com}) - t_{logic} + t_u$$
$$= 4 \bullet t_{com} + t_{logic} + t_u$$

Dabei wurde für den Ausgabeweg kein Fehler unterstellt und der Verzug der lokalen Reaktion $t_{logic}$ wurde wieder für $t_{block}$ gegen gerechnet.

Im folgenden Fehlerszenario wird angenommen, dass nach dem Empfangen der von der Ausgangskomponente gesendeten Nachricht 1 durch die Sicherheitssteuerung die Übertragungsstrecke zwischen Sicherheitssteuerung und Ausgangskomponente unterbrochen wird (siehe Abbildung 3.8).

Der Zeitverzug auf den nicht zur Ausgabe gebrachten neuen Wert $b_2$, wird durch die sichere Reaktion der Ausgangskomponente ersetzt. Die sichere Reaktion der Ausgangskomponente wird ausgelöst, nachdem sie für die Zeit $t_{com\text{-}out}$ (nachfolgend $t_{com}$) nach dem Empfang von $a_1$ keine neue Nachricht von der Sicherheitssteuerung bekommt.

Als Reaktionszeit ergibt sich bei diesem Fehlerszenario

$$t_{error\text{-}c} = 2 \bullet t_{com} + t_{block} + (t_{com} + t_{cyc\text{-}out}) + t_{cyc\text{-}out} + t_u \qquad (3.24)$$
$$= 4 \bullet t_{com} + 2 \bullet t_{cyc\text{-}out} + t_u$$

- Der Term $2 \bullet t_{com}$ steht für die maximale Reaktionszeit von Eingangssignal zu Sicherheitssteuerung.
- Der Term $t_{block}$ wie oben[104] für das verzögerte Senden des Signals.
- Der Term $(t_{com} + t_{cyc\text{-}out})$ für die Überwachungszeit und das verzögerte Erkennen des Ablaufens.

[103] Bemerkung: Strategien, um bei der zyklischen Variante den Aufschlag für das Erkennen der abgelaufenen Überwachungszeiten zu verringern, indem die Überwachungszeit um die erwartete Zykluszeit verkürzt wird, sind aus zweierlei Gründen nicht sinnvoll. Erstens wird dadurch die dem Protokollpartner von der Protokollspezifikation zugestandene Zeit nicht eingeräumt und zweitens muss dann nicht nur die erwartete Zykluszeit, sondern auch die Zeit für die Überwachung des Zyklusses herangezogen werden. Insbesondere letztere ist in der Regel aus Sicht der Applikation unangemessen groß. Statt dessen kann die Überwachungszeit besser kürzer definiert werden; hierbei sind insbesondere die möglichen Granularitäten besser, als bei den Zykluszeiten.

[104] siehe Gleichung (3.23)

- Der Term $t_{cyc\text{-}out}$ für die Reaktion, bis der Ausgang geschrieben wird.

Im nächsten Fehlerszenario wird angenommen, dass vor dem spätest möglichen Empfangen der Nachricht $b_2$ durch die Ausgangskomponente, diese ausfällt. Der Ausfall der Ausgangskomponente führt innerhalb der Sicherheitszeit derselben zur Abschaltung der Ausgänge. Für die Sicherheitszeit wird nachfolgend $2 \bullet t_{cyc\text{-}out}$ angenommen.

$$t_{error\text{-}d} = 2 \bullet t_{com} + t_{block} + (t_{com} + t_{cyc\text{-}out}) + 2 \bullet t_{cyc\text{-}out} + t_u \qquad (3.25)$$
$$= 4 \bullet t_{com} + 3 \bullet t_{cyc\text{-}out} + t_u$$

- Der Term $2 \bullet t_{com}$ steht für die maximale Reaktionszeit von Eingangssignal zu Sicherheitssteuerung.
- Der Term $t_{block}$ für das verzögerte Senden des Signals.
- Der Term $(t_{com} + t_{cyc\text{-}out})$ für die Überwachungszeit und das verzögerte Erkennen des Ablaufens.
- Der Term $2 \bullet t_{cyc\text{-}out}$ für die Reaktion bis das Ausfallen der Ausgangskomponente auf die Ausgänge wirkt.
- Der Term $+ t_u$ wiederum die maximale Ungenauigkeit der Empfangszeitüberwachung aller drei Komponenten.

Als Fehlerszenario wird weiter angenommen, dass die Sicherheitssteuerung unmittelbar vor dem Empfang von Nachricht $b_2$ die Nachricht $a_1$ an die Ausgangskomponente geschickt hat und dann ausfällt (siehe $t_1$ in Abbildung 3.8). Weiter wird angenommen, dass die Nachricht $a_1$ bei der Ausgangskomponente zum letztmöglichen gültigen Zeitpunkt ankommt. Der Ausfall der Sicherheitssteuerung führt dazu, dass sie keine weitere Verarbeitungen durchführt.

Dann ergibt sich für die Reaktionszeit

$$t_{error\text{-}e} = 2 \bullet t_{com} + t_{com} + (t_{com} + t_{cyc\text{-}out}) + t_{cyc\text{-}out} + t_u \qquad (3.26)$$
$$= 4 \bullet t_{com} + 2 \bullet t_{cyc\text{-}out} + t_u$$

- Der Term $2 \bullet t_{com}$ steht für die maximale Reaktionszeit von Eingangssignal zu Sicherheitssteuerung.
- Der Term $t_{com}$ für von der Ausgangskomponente tolerierte Überwachungszeit nach ihrem Empfang von Nachricht $x_0$.
- Der Term $(t_{com} + t_{cyc\text{-}out})$ für die Überwachungszeit und das verzögerte Erkennen des Ablaufens der nicht mehr gesendeten Nachricht $b_2$, nachdem $a_1$ bei der Ausgangskomponente verarbeitet wurde.
- Der Term $t_{cyc\text{-}out}$ für die Reaktion bis der Ausgang geschrieben wird.
- Der Term $+ t_u$ wiederum die maximale Ungenauigkeit der Empfangszeitüberwachung aller drei Komponenten.

Bei der Mehrfehlerbetrachtung wird betrachtet, dass die Verbindung zwischen Eingangskomponente und Sicherheitssteuerung unterbrochen wird und zusätzlich vor der spätest möglichen Reaktion des ausbleibenden Empfangs der Nachricht $b_2$ durch die Ausgangskomponente, diese ausfällt (siehe $t_3$ in Abbildung 3.8). Der Ausfall der Ausgangskomponente führt innerhalb der Watch-Dog-Zeit zur Abschaltung der Ausgänge. Für die Watch-Dog-Zeit wird folgend $t_{cyc\text{-}out}$ angenommen.

$$t_{error\text{-}f} = (t_{error\text{-}in\text{-}b}) + (t_{block} + (t_{com} + t_{cyc\text{-}out} + t_{cyc\text{-}out}) + t_{cyc\text{-}out}) - t_{logic} \qquad (3.27)$$
$$= 4 \bullet t_{com} + t_{logic} + 3 \bullet t_{cyc\text{-}out} + t_u$$

Für ein weiteres Fehlerszenario wird angenommen, dass $t_{com}$ der Eingangskomponente durch verspätetes Senden der Sicherheitssteuerung überschritten wird und wiederum die Ausgangskomponente wie oben ausfällt. Dazu erhält man

$$t_{error\text{-}g} = (t_{error\text{-}in\text{-}a}) + (t_{block} + (t_{com} + t_{cyc\text{-}out}) + 2 \bullet t_{cyc\text{-}out}) - t_{logic} \quad (3.28)$$
$$= 4 \bullet t_{com} + 2 \bullet t_{logic} + 3 \bullet t_{cyc\text{-}out} + t_u$$

Betrachtet man die verschiedenen Fehlerszenarien, so erhält man für die Worst-Case-Reaction-Time mit einem Fehler das Maximum aus $t_{b\text{-}in\text{-}out}$, $t_{error\text{-}a}$, $t_{error\text{-}b}$, $t_{error\text{-}c}$, $t_{error\text{-}d}$ , $t_{error\text{-}e}$ (siehe die Gleichungen (3.6), (3.21), (3.23), .. , (3.26)).

Setzt man den üblichen Fall ein, dass $t_{cyc\text{-}in} < t_{cyc\text{-}out}$ und $3 \bullet t_{cyc\text{-}out} < 2 \bullet t_{logic}$ ist, so stellt

$$t_{error\text{-}a} = 4 \bullet t_{com} + 2 \bullet t_{logic} \quad (3.29)$$

das Maximum dar.

Für den Vergleich zu $t_{wc1}$ mit

$$t_{wc1} = \{\sum t_{max-x}\} + max\{t_{wc-x} - t_{max-x}\} \quad (3.30)$$

von Gleichung (3.2), setzt man für $t_{max\text{-}x}$ jeweils $2 \bullet t_x$ mit $t_x$ aus $\{t_{cyc\text{-}in}, t_{cyc\text{-}out}, t_{com}, t_{logic}\}$ ein.

Setzt man hier für $t_{max}$ den gleichen Wert wie für $t_{wc}$ ein, so erhält man $t_{wc2}$ und ein positives Delta zu Gunsten der Implementationsabhängigen Berechnung.

$$t_{delta} = t_{wc1} - t_{error\text{-}a}$$
$$= (2 \bullet t_{cyc\text{-}in} + 2 \bullet t_{com} + 2 \bullet t_{logic} + 2 \bullet t_{com} + 2 \bullet t_{cyc\text{-}out}) - t_{error\text{-}a}$$
$$= 2 \bullet t_{cyc\text{-}in} + 2 \bullet t_{cyc\text{-}out} \quad (3.31)$$

Hierbei wurde für die Implementation ein sehr effizientes Verfahren angesetzt. Ist dies für die konkrete Anwendung nicht der Fall, so kann man mit $t_{wc1}$ zu kleine Reaktionszeiten berechnen. Ebenfalls ist zu beachten, dass das Delta für die Implementationsabhängige Betrachtung nur einen Fehler angenommen hat, $t_{wc2}$ jedoch Mehrfachfehler unterstellt.

Betrachtet man die verschiedenen Fehlerszenarien, so erhält man für die Worst-Case-Reaction-Time mit mehren Fehlern das Maximum aus $t_{max}$, $t_{error\text{-}a}$ bis $t_{error\text{-}g}$.

Somit stellt

$$t_{error\text{-}g} = 4 \bullet t_{com} + 2 \bullet t_{logic} + 3 \bullet t_{cyc\text{-}out} + t_u \quad (3.32)$$

das Maximum dar.

Für den Vergleich zu $t_{wc2}$ wird hierbei die jeweilige Zeit $t_{wc}$ gleich der Worst-Case-Zeit gesetzt.

Vergleicht man dies mit dem Implementations-unspezifischen Ansatz, so erhält man für

$$t_{wc2} = t_{input|wc} + t_{com\text{-}input|wc} + t_{logic|wc} + t_{com\text{-}output|wc} + t_{output|wc} \quad (3.33)$$

wie in Gleichung (3.3) und setzt für $t_{wc}$ jeweils $2 \bullet t_x$ mit $t_x$ aus $\{t_{cyc\text{-}in}, t_{cyc\text{-}out}, t_{com}, t_{logic}\}$ ein.

$$t_{wc2} = 2 \bullet t_{cyc\text{-}in} + 2 \bullet t_{com} + 2 \bullet t_{logic} + 2 \bullet t_{com} + 2 \bullet t_{cyc\text{-}out} \quad (3.34)$$

Vergleicht man dies mit $t_{error\text{-}g}$ , so ist das Delta.

$$t_{delta} = t_{wc2} - t_{error\text{-}g} = 2 \bullet t_{cyc\text{-}in} - t_{cyc\text{-}out} \quad (3.35)$$

Da das Delta $t_{delta}$ in Abhängigkeit der Verhältnisse zwischen $t_{cyc\text{-}in}$ und $t_{cyc\text{-}out}$ auch positiv werden kann, ist hier ersichtlich, dass eine Implementations- unabhängige Berechnung der Worst-Case-Reaction-Time keinen Sinn ergibt, ja gefährlich sein kann. Insbesondere sind für die Implementierung auch weniger effiziente Verfahren als das oben gewählte möglich, die ein noch größeres negatives Delta ergeben können.

# 4 Analyse und Vergleich existierender Protokolle

In diesem Kapitel werden die Protokolle PROFIsafe, CIP-Safety und FF-SIF analysiert. Es werden die Maßnahmen der jeweiligen Protokolle den Fehlerarten und Gefahrenpotentialen gegenüber gestellt und anhand der quantitativen Analyse die Eignung zur Erkennung der Fehlerarten aufgezeigt.

## 4.1 PROFIsafe-Protokoll

Das PROFIsafe Protokoll wurde für den Safety Integrity Level 3 gemäß IEC 61508 entworfen. Dabei wird von einem Black-Channel Modell ausgegangen, dessen Qualität durch den sogenannten *SIL-Monitor* überwacht wird[105].

Als Sicherungsmechanismus zur Erkennung der Nachrichtenverfälschung kommen 24- und 32-Bit CRCs zum Einsatz, deren Wirksamkeit mit der Funktion des patentrechtlich geschützten *SIL-Monitors*[106] gekoppelt ist.

Zur zeitlichen Überwachung werden Watch-Dog-Timer, sowie ein nicht übertragener Monotoniezähler (*vconsnr*) genutzt. Zur Adressierung wird eine Id (*code-name*[107]) genutzt, die die Identität der sicheren Kommunikationsbeziehung darstellt. Die Id wird in der Nachricht nicht übertragen, sondern ist nur in der CRC-Berechnung enthalten. Die Id besteht aus den Adressen der beiden Kommunikationspartner.

Angewendet wird PROFIsafe zwischen einer Sicherheitssteuerung (F-Host) und einem Slave (F-Device). Die Sicherheitssteuerung generiert Sendungen mit jeweils erhöhtem Monotoniezähler, die vom Slave bestätigt werden. Dabei können in beiden Richtungen Nutzdaten transportiert werden.

Der Nachrichtenrahmen für PROFIsafe im V2-Mode setzt sich wie folgt zusammen[107]:

*Tabelle 4.1: PROFIsafe Nachrichtenformat*

| ***F-Data*** | ***Status Byte*** | ***CRC2*** |
|---|---|---|
| sichere Prozessdaten<br>max. 12 oder 123 Bytes | 1 Byte Statusinformationen, incl. LSB der *vconsnr*, auch Toggle Bit genannt | CRC24 oder CRC32 |

Der CRC2 wird über F-Data, Status Byte, vconsnr und die F-Parameter[107] gerechnet. Der CRC24 kommt bei Prozessdaten bis zu 12 Bytes Länge und der CRC32 bei Prozessdaten bis zu 123 Bytes Länge zum Einsatz.
Betrachtet wird in diesem Kapitel das PROFIsafe Protokoll für den Einsatz mit den Bussystemen PROFIBUS-DP und PROFINET-IO. Dabei wird der Version-2-Mode des Protokolls PROFIsafe zu Grunde gelegt, der aus der Version 1 entwickelt wurde.

Spezifiziert ist das Protokoll PROFIsafe im Standard IEC 61784-3-3 als Teil des allgemeinen Standards für Feldbusse IEC 61784 und nimmt dort für CPF[108] 3/5 und CPF 3/6 die Rolle des sicherheitsgerichteten Kommunikationsprotokolls mit der Bezeichnung FSCP[109] 3/1 ein.

In der PROFIsafe Spezifikation[110] werden die Begriffe F-Host für Sicherheitssteuerung und F-Device für Eingangs- oder Ausgangskomponenten benutzt.

---

[105] [PROFIsafeV2]

[106] [Bart99]

[107] Konfigurationsparameter des Protokolls, enthalten auch die Adressinformationen.

[108] Communication Profile Family

[109] Fail Safe Communication Profile

[110] [PROFIsafeV2]

Setzt man als Modell für die Übertragungseinrichtung den binär-symmetrischen Übertragungskanal voraus, so ergibt sich bei gegebener Hamming-Distanz die Restfehlerwahrscheinlichkeit pro Stunde unerkannt verfälschter, empfangener Nachrichten, wie bereits bekannt, mit:

$$\Lambda = R(p, N, k) \bullet 3600 \bullet \nu \bullet c \qquad (4.1)$$

Dabei sind

- $N+k$ = Anzahl Bits der Nachricht
- $k$ = Anzahl der Prüfsummenbits
- $p$ = Bitfehlerrate
- $v$ = Anzahl Nachrichten pro Sekunde
- $c$ = Anzahl der sicherheitsgerichteten Kommunikationsbeziehungen in einer Sicherheitsfunktion
- $d$ = Hamming-Distanz
- $$R(p, N, k) \approx \frac{1}{2^k} \bullet \sum_{n=d}^{N+k} \binom{N+k}{n} \bullet p^n \bullet (1-p)^{(N+k-n)} \qquad (4.2)$$

PROFIsafe benutzt ein Nachrichtenformat für bis zu 12 Nutzdaten-Bytes mit einem 24-Bit-CRC und ein zweites Nachrichtenformat für bis zu 123 Nutzdaten-Bytes mit einem 32-Bit-CRC. Damit ergeben sich maximale Nachrichtenlängen von N+k=128 Bits, bzw. N+k=1024 Bits. Das Status-Byte des Protokolls wurde für beide Nachrichtenlängen noch hinzugerechnet.

Für PROFIsafe wird weiter angesetzt: eine Bitfehlerrate $p=10^{-3}$, $v=10.000$ und $c=3$.

Die Bitfehlerrate wurde entgegen der IEC 61508-2 und der IEC 61784-3 mit $p=10^{-3}$ angesetzt, da für $p=10^{-2}$ das gewünschte Sicherheitslevel zu weit entfernt liegt. Auf die daraus resultieren Konsequenzen wird im Weiteren eingegangen.

Der Parameter $c$ repräsentiert dabei, dass eine Sicherheitsfunktion sich aus Kommunikationsstrecken für die Übertragung der Eingangswerte zur Logikverarbeitung und von dort zur Ausgabeeinheit zusammensetzt.

Der Parameter $v$ repräsentiert die maximal vom Übertragungssystem zur Verfügung gestellte Übertragungsleistung. Der Wert von 10.000 Nachrichten/Sekunde wird von der IEC 61784-3-3 als maximal zulässige Nachrichtenrate pro Verbindung angesetzt.

Als CRC Polynome werden CRC-a 0x1_5D_6DCB[111] für bis zu 12 Byte-Nutzdaten mit einer Hamming-Distanz von 6 und CRC-b 0x1_F4AC_FB13[112] für bis zu 123 Byte-Nutzdaten mit einer Hamming-Distanz von 6 eingesetzt.

Die beiden CRCs sind über die betrachteten Nachrichtenlängen proper[113].

Die Restfehlerwahrscheinlichkeit pro Stunde für CRC-a ergibt sich mit

$$R(p=10^{-3}, N=104, k=24) = \frac{1}{2^{24}} \bullet 4{,}88543 \bullet 10^{-9} = 2{,}91195 \bullet 10^{-16} \qquad (4.3)$$

zu

$$\Lambda \le R() \bullet 3600 \bullet 10.000 \bullet 3/h = 3{,}1449 \bullet 10^{-8} h^{-1} < 1 \bullet 10^{-7} h^{-1} = 1\% \ \ von \ \ SIL1 \qquad (4.4)$$

Mit der CRC-a Sicherung alleine wird offensichtlich nicht das Sicherheitslevel für SIL3 erreicht. Es wird lediglich SIL1 erzielt.

Erst wenn die hohe Nachrichtenrate von 10.000 auf 100 Nachrichten/Sekunde reduziert wird, ergibt sich aus der obigen Rechnung SIL3.

---

[111] auch bekannt als CRC-24/6.1 bei [Castagnoli93], sowie als FlexRay CRC, siehe [Flex05]. HD=6 bis zu einer Länge von 2048 Bits [Cast93].

[112] siehe 0xFA56_7D89 bei [Koop02], HD=6 bis zu einer Länge von 32768 Bits.

[113] [IEC 61784-3-3], [Cast93]

Die Anzahl an einer Sicherheitsfunktion beteiligten Knoten ist mit 4 ($c$=3) sehr gering angesetzt. Will man SIL3 erreichen, darf sie auch bei einer reduzierten Nachrichtenrate von 100 Nachrichten/Sekunde nicht erhöht werden!

Erst bei einer Bitfehlerrate von $p=10^{-4}$ erreicht das CRC-a Sicherungsverfahren unter der zugelassenen Nachrichtenrate von 10.000 eine Restfehlerwahrscheinlichkeit < 1% von SIL3.

$$R(p=10^{-4}, N=104, k=24)=\frac{1}{2^{24}}\bullet 5{,}36719\bullet 10^{-15}=3{,}19909\bullet 10^{-22} \qquad (4.5)$$

zu

$$\Lambda \le R()*3600\bullet 10.000\bullet 3h^{-1}=3{,}45502\bullet 10^{-14}h^{-1} \;<\; 1\bullet 10^{-9}h^{-1}=\; 1\%\;\; von\;\; SIL3 \qquad (4.6)$$

Bei dieser Bitfehlerrate ist es dann auch möglich die Anzahl der Verbindungen einer Sicherheitsfunktion um den Faktor 28.943,39 auf c=86.830 zu erhöhen. Dies sollte für alle realen Anwendungen lange ausreichend sein, zumal bei PROFIsafe die maximale Knotenanzahl auf 65.534 begrenzt ist[114].

Die Restfehlerwahrscheinlichkeit pro Stunde für CRC-b ergibt sich mit

$$R(p=10^{-3}, N=992, k=32)=\frac{1}{2^{32}}\bullet 6{,}64689\bullet 10^{-4}=1{,}5476\bullet 10^{-13} \qquad (4.7)$$

zu

$$\Lambda \le R()\bullet 3600\bullet 10.000\bullet 3h^{-1}=1{,}67141\bullet 10^{-5}/h \;>\; 1\bullet 10^{-7}h^{-1}=\; 1\%\;\; von\;\; SIL1 \qquad (4.8)$$

Mit der CRC-b Sicherung alleine wird das Sicherheitslevel für SIL3 nicht erreicht. Auch dann, wenn die hohe Nachrichtenrate auf 100 reduziert wird, ändert sich nichts daran, dass SIL3 nicht erreicht wird! Zudem wurde wie bei der Betrachtung des kurzen Nachrichtenformats die Anzahl der an einer Sicherheitsfunktion beteiligten Knoten mit 4 ($c$=3) sehr gering angesetzt, so dass sich die Restfehlerwahrscheinlichkeit in bestimmten Applikationen noch erhöht.

Erst bei einer Bitfehlerrate von $p=10^{-4}$ erreicht das CRC-b Sicherungsverfahren unter den zugelassenen Nachrichtenrate von 10.000 eine Restfehlerwahrscheinlichkeit < 1% von SIL3.

$$R(p=10^{-4}, N=992, k=32)=\frac{1}{2^{32}}\bullet 1{,}44621\bullet 10^{-9}=3{,}36722\bullet 10^{-19} \qquad (4.9)$$

zu

$$\Lambda \le R()\bullet 3600\bullet 10^{5}\bullet 3h^{-1}=3{,}6366\bullet 10^{-11}h^{-1} \;<\; 1\bullet 10^{-9}h^{-1}=\; 1\%\;\; von\;\; SIL3 \qquad (4.10)$$

Bei dieser Bitfehlerrate ist es dann möglich die Anzahl der Verbindungen einer Sicherheitsfunktion um den Faktor 27,49, auf $c$=82, zu erhöhen. Dies sollte für viele Anwendungen ausreichend sein, bleibt jedoch eine zu beachtende Einschränkung.

Um die reduzierte Bitfehlerrate von $10^{-4}$ ansetzen zu können geht PROFIsafe den Weg die Übertragungsqualität der Übertragungseinrichtung fortwährend zu messen. Dies wird als „SIL-Monitor“[115] bezeichnet und ist in den Varianten A und B definiert.

Beide Varianten beruhen darauf, dass es bei angenommener Bitfehlerrate eine Annahme über die je Zeitintervall vom Standard-Bussystem empfangenen, unerkannt verfälschten Nachrichten gibt. Überschreitet die gemessene Anzahl diese Annahme, so erfolgt die Sicherheitsreaktion.

Bei Variante A ist das Zeitintervall endlich und von dem geforderten SIL und der Nachrichtenlänge abhängig. Im Zeitintervall ist eine Nachricht mit falschem CRC zulässig.

---

[114] [PROFIsafeV2]

[115] [IEC 61784-3-3], sowie patentrechtlich geschützt nach [Bart99]

*Tabelle 4.2: SIL-Monitor Variante A* [116]

| *SIL* | *CRC* | *Nachrichtenlänge* | *Zeitintervall* |
|---|---|---|---|
| 3 | 24 Bit | ≤ 16 Bytes | 10 Std. |
| 2 | 24 Bit | ≤ 16 Bytes | 1 Std. |
| 3 | 32 Bit | ≤ 128 Bytes | 10 Std. |
| 2 | 32 Bit | ≤ 128 Bytes | 1 Std. |

Für Nachrichten bis zu 16 Bytes, gesichert mit CRC24, ist bei SIL2 Anwendungen eine verfälschte, vom Standardbussystem durchgelassene Nachricht in einer Stunde zulässig. Bei SIL3 Anwendungen verschärft sich die zulässige Rate, entsprechend der zulässigen Restfehlerwahrscheinlichkeiten der beiden SILs, um eine Größenordnung.

*Bemerkung: Die Nachrichtenlänge 16 Bytes ergibt sich aus 12 Bytes Nutzdaten, 1 Byte Status und 3 Bytes CRC. Die Nachrichtenlänge 128 Bytes berechnet sich aus 123 Bytes Nutzdaten, 1 Byte Status und 4 Bytes CRC.*

Bei Variante B ist das Zeitintervall unendlich und folglich wird beim ersten Empfang einer fehlerhaften, vom Standardbussystem nicht erkannten Nachricht die Sicherheitsreaktion eingeleitet. Dies ist die empfohlene Einsatzart[117]. Da der „SIL-Monitor“ nur für Sicherheitssteuerungs vorgesehen ist und Eingangs-/Ausgangskomponenten bei jedem Fehler mit dem Verbindungsabbruch reagieren, macht die Verwendung von Variante A ohnehin kaum Sinn.

Die Erkennung einer vom Standard-Bussystem unerkannt verfälschten Nachricht soll durch das jeweils verwendete PROFIsafe CRC-Polynom geschehen.

Dieser Ansatz stellt jedoch, unabhängig von Variante A und B, die folgenden Voraussetzungen an den Einsatz:

1. Beginnt[118] der Betrieb einer Kommunikationsverbindung, so muss deren Kommunikationsstrecke zu diesem Zeitpunkt eine Bitfehlerrate ≤ $10^{-4}$ aufweisen. Dies ist erforderlich, da das Modell von einem Erkennen des Anstiegs der Bitfehlerrate durch den verwendeten PROFIsafe CRC-a oder CRC-b ausgeht und bei Erreichen des entsprechenden Grenzwerts abschaltet. Dies ist jedoch nur mit hinreichender Wahrscheinlichkeit für SIL3 gegeben, wenn die Bitfehlerrate ≤ $10^{-4}$ ist.
2. Die Bitfehlerrate darf nicht sprunghaft über $10^{-4}$ ansteigen. Dies ist erforderlich, da die Erkennung von verfälschten Nachrichten durch den PROFIsafe CRC erst bei Bitfehlerraten unter $10^{-4}$ mit SIL3-Qualität funktioniert.

In den Vorschriften der IEC 61784-3-3 für die PROFIsafe Protokollimplementierung wird gefordert, dass schon bei der ersten als verfälscht erkannten Nachricht die Sicherheitsreaktion eingeleitet wird (Variante B). Dennoch bleiben die beiden Voraussetzungen problematisch, da es dem Anwender obliegt diese einzuhalten. Dies stellt eine Aufgabe dar, die der Anwender im normalen Betrieb nicht leisten kann.

Ein weiteres Problem bekommt dieses Verfahren in gestörten Umgebungen, da dann bei PROFIBUS-DP mit einer Wahrscheinlichkeit von ca.$10^{-5}$ (16 Bit CRC) verfälschte Nachrichten unerkannt vom Standardbussystem an PROFIsafe weitergegeben werden und in Folge die Verfügbarkeit herabgesetzt wird.

---

[116] aus [IEC 61784-3-3]

[117] gemäß [IEC 61784-3-3]

[118] Start oder Wiederanlauf

In die gleiche Richtung geht die Reaktion auch dann, wenn das Standardübertragungssystem, die Busanschaltung, Fehler aufweist, die dann, auch bei PROFINET, eine entsprechende Reduktion der Verfügbarkeit hervorrufen.

Die Abkehr vom Black-Channel Prinzip wäre gegebenenfalls eine sinnvollere Alternative, da dann jedes Endgerät mindestens eine, zugegebenermaßen ungetestete, weitere unabhängige CRC-Sicherungseinheit (gilt für PROFIBUS-DP, wie auch für PROFINET-IO) für die sicherheitstechnische Betrachtung einbringt.

Das PROFIsafe Protokoll definiert eine Verbindung als Kommunikation zwischen einem F-Host und einem F-Device. Wegen dem verwendeten unterlagerten Transportprotokollen PROFINET-IO und PROFIBUS-DP werden Nachrichten, mit enthaltenen PROFIsafe Anteilen, unabhängig vom Zustand des PROFIsafe Protokolls versendet.

Das heißt im technischen Sinne ist eine Kommunikationsverbindung unabhängig vom Zustand vorhanden. Auf der logischen Ebene besitzt PROFIsafe jedoch eine Menge (a) von Zuständen, in denen das Protokoll eine Anlaufsequenz erwartet und eine Menge (b) von Zuständen, in denen aktuelle, als gültig markierte Prozesswerte verarbeitet werden.

Zwecks einfacherer Beschreibung ist die Zustandsmenge (a) die Situation, in der keine PROFIsafe Verbindung existiert und die Zustandsmenge (b) die Situation, in der eine solche Verbindung existiert.

Das Öffnen der Verbindung ist als Übergang von der Zustandsmenge (a) in die Menge (b) definiert. Der Übergang von (b) nach (a) wird als Schließen definiert.

Zur Erkennung von unerwünschter Nachrichtenwiederholungen verwendet PROFIsafe das virtuelle Nachrichtenfeld „virtual consecutive number“ (kurz vconsnr). Virtuell deshalb, weil die Information in der Nachricht nicht übertragen wird, sondern durch das Einbeziehen in die CRC-Berechnung wirksam wird[119]. Eine falsche vconsnr äußert sich daher durch einen CRC-Fehler.

Bei der vconsnr handelt es sich um einen streng monoton steigenden, umlaufenden 24 Bit-Zähler. Er wird für jede neue Nachricht um eins erhöht. Dabei wird die 0 ausgespart, da diesem Wert eine besondere Bedeutung beim Anlauf nach CRC-Fehler oder *F_WD_Time* Timeout des Protokolls zukommt.

„Neue Nachricht“ könnte in diesem Zusammenhang etwas irreführend sein, da es gestattet ist, bei Wiederholungen von Nachrichten aktuellere Daten, als die beim vorherigen Senden, in der Nachricht zu übertragen[119].

Beim Einsatz von PROFIsafe ist zu beachten, dass die eingesetzten Übertragungseinrichtungen PROFIBUS-DP und PROFINET-IO eine Nachricht im Allgemeinen mehrfach übertragen. Das heißt, eine PROFIsafe Nachricht wird gemäß dem Zyklus des Standardbussystems solange wiederholt übertragen, bis eine neue PROFIsafe Nachricht zur Übertragung bereit gestellt wird.

Daher stellt es für PROFIsafe den Normalfall dar, dass Nachrichten wiederholt werden. Erkannt wird die Wiederholung am unverändertem Toggle-Bit, da dessen Erwartungshaltung nach einem gültigen Empfang invertiert wird.

Die als Duplikat erkannten Nachrichten werden vom PROFIsafe Empfänger verworfen, wenn sie einen korrekten CRC haben. Ist der CRC nicht korrekt, so wird die Verbindung geschlossen[120].

Bei PROFIsafe gibt es zwei relevante Anlaufsequenzen. Erstens, die nach dem PowerON (i) der Sicherheitssteuerung und zweitens, die Sequenz nach erkanntem CRC- oder *F_WD_Time*-Timeout-Fehler (ii), die zu einer gefährlichen Situation führen können. Die PROFIsafe Verbindung ist dann geschlossen. Die gefährliche Situation mündet in beiden Fällen darin, dass in einer Netzwerkkomponente gespeicherte, veraltete und dann wiederholte Prozessdaten wirksam werden können.

---

[119] [PROFIsafeV2]

[120] siehe [PROFIsafeV2] F-Host state diagram T6, bzw. F-Device state diagram T34

Im Fall (i) setzt die Parametrierung der Eingangs-/Ausgangskomponenten die vconsnr der Eingangs-/Ausgangskomponenten zurück auf `0xfffff0` und die Sicherheitssteuerung beginnt mit der festgelegten vconsnr `0xfffff0`.

Im Rahmen des PROFIsafe Protokolls folgen von der Sicherheitssteuerung nun 3 Nachrichten, die dazu führen, dass die Eingangs-/Ausgangskomponente ihre „Fail-Safe-Values" ausgibt. Mit der 4. Nachricht ist es erstmals möglich Prozesswerte durch die Eingangs-/Ausgangskomponente ausgeben zu lassen[121].

Das heißt mit der unerwünschten Wiederholung genau dieser 4 Nachrichten ist es möglich eine nicht sichere Reaktion einer Ausgangskomponente zu initiieren. Damit dieses Szenario anwendbar ist, muss die Eingangs-/Ausgangskomponente parametriert worden sein, z.B. nach PowerON der Eingangs- / Ausgangskomponenten, bevor die unerwünschte Wiederholung beginnt.

Im Fall (ii) sendet die Sicherheitssteuerung mit der R_cons_nr=1 den Reset für die vconsnr der Eingangs-/Ausgangskomponente und einem Indikator für einen „Operator-Acknowledge-Requested". Danach folgen 2 weitere Sicherheitssteuerung-Nachrichten, während die Eingangs-/Ausgangskomponente noch „Fail-Safe-Values" ausgibt. Ab der 4. Nachricht können nun Prozesswerte durch die Eingangs- / Ausgangskomponente ausgegeben werden[120].

Um die Eingangs- / Ausgangskomponente in einen für dieses Szenario empfänglichen Zustand zu setzen, gibt es verschiedene Möglichkeiten. Neben dem Ablaufen der *F_WD_Time* der Eingangs-/Ausgangskomponente, führt insbesondere auch das Einfügen von Nachrichten die Eingangs-/Ausgangskomponente in diesen Zustand.

> *Bemerkung: Eingefügte Nachrichten werden bei erwartetem Toggle-Bit wie verfälschte Nachrichten betrachtet und damit der sichere Zustand hergestellt und mittels PROFIsafe Nachricht signalisiert und eine Anlaufsequenz erwartet (Verbindung geschlossen). Dies gilt gleichermaßen für die Sicherheitssteuerung und für die Eingangs-/Ausgangskomponente.*[120]

Wird vor der genannten 4er Sequenz nur eine weitere ungültige PROFIsafe Nachricht an die Eingangs-/Ausgangskomponente geschickt, so ist sie in eben diesem Zustand „Verbindung geschlossen". Wenn es zur unerwünschten Wiederholung von Nachrichtensequenzen kommt, ist dieser Umstand sehr wahrscheinlich, da es sich um die selbe Fehlerursache handelt.

Die obige Nachrichtensequenz bestehen aus Sicht der Eingangs-/Ausgangskomponente aus 4 Nachrichten, bis erstmalig Prozesswerte ausgegeben werden und der Fehler wirksam wird.

Setzt man die erwartete Fehlerrate[122] für das Wiederholen einer Nachricht mit $10^{-6}\,h^{-1}$ an, so ergibt sich für 4 Nachrichten eine Restfehlerwahrscheinlichkeit pro Stunde von

$$\left(10^{-6}\right)^4 h^{-1} \;=\; 10^{-24}\,h^{-1} \;\le\; 1 \bullet 10^{-9}\,h^{-1} = \;1\%\;\; von\;\; SIL3 \qquad (4.11)$$

Damit erreicht die Maßnahme der vconsnr mit vorgegebener Anlaufsequenz den Level SIL3. Die Betrachtung, mit welcher Wahrscheinlichkeit die passende Nachrichtensequenz gespeichert wird, ist hier nicht mehr notwendig.

Der Verlust einer Nachricht wird durch die Erwartungshaltung der strengen Monotonie der vconsnr oder an der Zeiterwartung für den Empfang von Nachrichten erkannt.

Betrachtet wird hierbei für PROFIsafe das Einfügen von Nachrichten während einer bestehenden Verbindung. Der Fall, dass keine Verbindung besteht, wurde schon oben bei der Diskussion über die Wiederholung von Nachrichtensequenzen beim Starten bzw. Wiederanlaufen betrachtet.

---

[121] [IEC 61784-3-3]

[122] Annahme der Fehlerrate für einen Ethernet-Switch-Baustein mit 100 FIT, d.h. $10^{-7}$/h. Siehe auch [PROFIsafeV2]. Im Rahmen dieser Arbeit geht man immer von 10 beteiligten Swich-Bausteinen, also $10^{-6}$/h aus. Sollte dies in der Praxis nicht zutreffen, so ist die Fehlerrate anzupassen.

Für eine empfangene Nachricht prüft die Eingangs-/Ausgangskomponente den CRC mit der erwarteten vconsnr und schließt die Verbindung, wenn der CRC nicht passt[123]. Die Nachricht wird bei passendem CRC und unverändertem Toggle-Bit als Duplikat erkannt.

Eine Nachricht wird nicht als eingefügt erkannt, wenn die vconsnr zufällig passt oder wenn der CRC zufällig passt und das Toggle-Bit geändert ist. Da die vconsnr kleiner gleich als die beiden CRC-a und CRC-b ist, kann ausschließlich die vconsnr betrachtet werden. Die Wahrscheinlichkeit, dass sie zufällig passt, ist zusammen mit dem Toggle-Bit, $2^{-25}$. Das Toggle-Bit darf auch bei *CRC-a* (24-Bit) angesetzt werden, da es in der realen Nachricht übertragen wird und nicht ausschließlich durch den CRC kodiert (gesichert) ist.

Setzt man die erwartete Fehlerrate für das Wiederholen einer Nachricht mit $10^{-6}\,h^{-1}$ an, so ergibt sich Restfehlerwahrscheinlichkeit pro Stunde von

$$10^{-6}h^{-1} \bullet 2^{-25} = 5{,}960 \bullet 10^{-14}h^{-1} \leq 1 \bullet 10^{-9}h^{-1} = 1\% \ \ von \ \ SIL3 \qquad (4.12)$$

Damit erreicht die Maßnahme, eine eingefügte vconsnr über den CRC-a oder b) zu erkennen, den Level SIL3.

Das Einfügen von Nachrichten, die für andere Eingangs-/Ausgangskomponenten erstellt wurden, wird im später betrachtet.

Eine falsche Nachrichtenreihenfolge wird durch die Erwartungshaltung des Toggle-Bits und der strengen Monotonie der vconsnr erkannt. Die Restfehlerwahrscheinlichkeit für die Behandlung falscher Nachrichtenreihenfolgen entspricht der Behandlung von eingefügten Nachrichten.

Bei PROFIsafe gibt es jedoch einen Sonderfall, da es erlaubt ist, bei unveränderter vconsnr die „gleiche“ Nachricht mit veränderten, aktuelleren Prozessdaten erneut zur versenden[124]. Dies erfolgt um eine bessere Reaktionszeit im fehlerfreien Fall zu erzielen[125].

Eine Nachricht wird bei passendem CRC und unverändertem Toggle-Bit zwar als Duplikat erkannt, jedoch werden die Prozessdaten übernommen[126]. Bei einer als Duplikat erkannten Nachricht kann es sich um eine bezüglich der Reihenfolge vertauschten Nachricht handeln. Das heißt eine mit gleicher vconsnr bereits empfangene Nachricht kann nach der gerade empfangen gebildet und versendet worden sein. Dies ist insofern problematisch, dass eine Reihenvertauschung von Nachrichten mit gleicher vconsnr, aber geänderten Prozessdaten nicht erkannt wird und im Fehlerfall die alten Daten der letzten Nachricht verarbeitet werden, obwohl zuvor schon neuere vorlagen.

Dies ist insbesondere für einige Aktoren problematisch, da bei obiger Reihenfolgevertauschung zum Beispiel bei einer Abschaltung[127], der Aktor sehr kurz mit 0 angesteuert und danach wieder mit 1 angesteuert und schlussendlich dann wieder mit 0 angesteuert wird. Für Aktoren, die bei diesem Szenario eine Fehlfunktion oder erhöhten Verschleiß erfahren, müssen sich mittels Latch-Funktion selber gegen diese Fehler schützen.

Die Fragmentierung von Nachrichten, d.h. das Zusammensetzen einer Nachricht aus Teilen, z.B. aus anderen Nachrichten, bzw. die Reihenfolgevertauschung von Teilen einer Nachricht, wird von PROFIsafe durch den CRC über die gesamte Nachricht erkannt.

---

123 [PROFIsafeV2]

124 [PROFIsafeV2] Kapitel 9.3.2

125 private Diskussion

126 siehe [PROFIsafeV2] Transitionen T4 und T27

127 Wechsel des Prozessdatums von 1 nach 0

Zur Bestimmung der inakzeptablen Nachrichtenverzögerung, beziehungsweise des maximalen Nachrichtenalters, wird gemäß PROFIsafe wird die „Safety-Function-Respone-Time“ wie folgt berechnet:

$$SFRT = \left( \sum_{i=1}^{n} WCDT_i \right) + max_{i=1..n} \left( WDTime_i - WCDT_i \right) \qquad (4.13)\ ^{124}$$

Dabei sind:

$WCDT_i$ die maximale Verzögerung[128] der Komponente i in einer Kette bestehend aus Eingangskomponente, sichere Kommunikationsstrecke zur Sicherheitssteuerung, die Sicherheitssteuerung selber, sicherer Kommunikationsstrecke zur Ausgangskomponente und schlussendlich die Ausgangskomponente selber. Diese Zeiten enthalten keine Reaktion auf Fehler, sondern nur die maximale Verzögerung im fehlerfreien Fall.

$WDTime_i$ die Zeit, die nach dem Empfang einer gültigen Nachricht vergeht, bis auf eine danach auftretende Überwachungszeitüberschreitung in der Komponente eine sichere Reaktion ausgelöst wird.

Dabei wird für die maximale Verzögerung der Sicherheitsfunktion auch das Vorhandensein eines Fehlers betrachtet. Für die Eingangs-, Ausgangskomponente und die Sicherheitssteuerung ist die *WDTime* eine Angabe, die vom Hersteller der Komponente gemacht werden muss.

Bei der Kommunikationsstrecke wird die Zeit mit der Summe von *F_WD_Time1*[129] und *F_WD_Time2*[130] des PROFIsafe Protokolls, der $WCDT_i$ der Kommunikationsstrecke und der Zykluszeit der Sicherheitssteuerung bzw. Ausgangskomponente eingesetzt.

Für die Zeit zwischen Eingangskomponente und Sicherheitssteuerung ergibt sich
$WDTime_2 = F\_WD\_Time1 + WCDT_2 + t_{logic}$. *(4.14)*

Für die Zeit zwischen Sicherheitssteuerung und Ausgangskomponente ergibt sich
$WDTime_4 = F\_WD\_Time2 + WCDT_4 + t_{out}$. *(4.15)*

Für $t_{logic}$ und $t_{out}$ werden dabei die maximalen Zeiten ohne Fehler angenommen.

Setzt man für die Eingangs-/Ausgangskomponenten und die Sicherheitssteuerung das im Kapitel 3 beschriebene Verfahren des Modell-Protokolls ein und setzt

$$F\_WD\_Time1 = F\_WD\_Time2 = t_{com} \qquad (4.16)$$

und $t_{wc\text{-}xxx}$ für die Worst-Case-Zeit[131] mit Fehler, so erhält man:

*Tabelle 4.3: PROFIsafe WDTime und WCDT der aktiven Komponenten*

| ***Zeit*** | ***Eingangskomponente*** | ***Sicherheitssteuerung*** | ***Ausgangskomponente*** |
|---|---|---|---|
| $WDTime_i$ | $t_{com}+2 \bullet t_{wc\text{-}cyc\text{-}in}$ | $t_{com}+2 \bullet t_{wc\text{-}logic}$ | $t_{com}+2 \bullet t_{wc\text{-}cyc\text{-}out}$ |
| $WCDT_i$ | $2 \bullet t_{cyc\text{-}in}$ | $2 \bullet t_{logic}$ | $2 \bullet t_{cyc\text{-}out}$ |
| $WDTime_i\text{-}WCDT_i$ | $t_{com}+2 \bullet (t_{wc\text{-}cyc\text{-}in} - t_{cyc\text{-}in})$ | $t_{com}+2 \bullet (t_{wc\text{-}logic} - t_{logic})$ | $t_{com}+2 \bullet (t_{wc\text{-}cyc\text{-}out} - t_{cyc\text{-}out})$ |

[128] orginal: „worst case delay time“

[129] F_WD_Time zwischen Eingangskomponente und Sicherheitssteuerung

[130] F_WD_Time zwischen Sicherheitssteuerung und Ausgangskomponente. WD_Time2 ist nicht zu verwechseln mit der neuerdings in [PROFIsafeV2] eingeführten F_WD_TIME_2 (secondary F-Watchdog time)

[131] $t_{wc\text{-}xxx}$ *aus* $\{ t_{wc\text{-}cyc\text{-}in}, t_{wc\text{-}cyc\text{-}out}, t_{wc\text{-}logic} \}$

Die Werte für die Kommunikationsbeziehungen ergeben sich zu[132]:

*Tabelle 4.4: PROFIsafe WDTime und WCDT der Kommunikationsverbindungen*

| ***Zeit*** | ***Kommunikationsbeziehung Eingangskomponente - Sicherheits-steuerung*** | ***Kommunikationsbeziehung Sicherheitssteuerung - Ausgangs-komponente*** |
|---|---|---|
| $WDTime_i$ | $= F\_WD\_Time1+WCDT_2+t_{logic}$[133]<br>$= t_{com}+WCDT_2+t_{logic}$ | $= F\_WD\_Time2+WCDT_4+t_{cyc\text{-}out}$[134]<br>$= t_{com}+WCDT_4+t_{cyc\text{-}out}$ |
| $min\ WCDT_i$ | $= WCDT_2$<br>$= DAT + 2 \bullet Bus\text{-}Cycle + HAT$<br>$= t_{cyc\text{-}in} + 2 \bullet Bus\text{-}Cycle + t_{logic}$ | $= WCDT_4$<br>$= HAT + 2 \bullet Bus\text{-}Cycle + DAT$<br>$= t_{logic} + 2 \bullet Bus\text{-}Cycle + t_{cyc\text{-}out}$ |
| $WDTime_i\text{-}WCDT_i$ | $t_{com}+t_{logic}$ | $t_{com}+t_{cyc\text{-}out}$ |

Bus-Cycle ist die Rate, mit der Nachrichten der Verbindung vom unterlagerten Standardbussystem übertragen werden. Für PROFINET könnte hier z.B. 2 ms zur Anwendung kommen, wobei $F\_WD\_Time = t_{com}$ durch den Einfluss von $t_{logic}$ (z.B. = 50 ms) deutlich größer angesetzt werden muss. Dabei sind die vom Anwender eingestellten, Verfügbarkeits-erhöhenden größeren Werte für $t_{com}$ nicht berücksichtigt.

Korrekterweise müsste $DAT$[135] und $HAT$[136], nicht wie in PROFIsafe[137] beschrieben, nur die Zeit von dem Beginn der Verarbeitung einer empfangenen Nachricht bis zum Versenden der darauf folgenden sein, sondern zusätzlich die Zeit nach dem Eintreffen der Nachricht bei der Komponente, die vergeht, bis die Komponente mit der Bearbeitung der Nachricht beginnt. Dies ist in der eingesetzten Implementierungsvariante jeweils ein Zyklus der Komponente und führt zu den korrigierten $DAT_k$ und $HAT_k$. Hierbei nicht berücksichtigt sind die Zeiten, die durch die spezifische Busanschaltung und die Komponenten interner Kommunikationsstrecken entstehen. Diese müssten zusätzlich noch zu $DAT_k$ und $HAT_k$ hinzugerechnet werden.

*Tabelle 4.5: Korrigierte PROFIsafe WCDT und minimale F_WD_Time*

| ***Zeit*** | ***Kommunikationsbeziehung Eingangskomponente - Sicherheits-steuerung*** | ***Kommunikationsbeziehung Sicherheitssteuerung - Ausgangs-komponente*** |
|---|---|---|
| $min\ F\_WD\_Time$ | $= WCDT_2$<br>$= DAT_k + 2 \bullet Bus\text{-}Cycle + HAT_k$<br>$= 2 \bullet t_{cyc\text{-}in} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{logic}$ | $= WCDT_4$<br>$= HAT_k + 2 \bullet Bus\text{-}Cycle + DAT_k$<br>$= 2 \bullet t_{logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out}$ |

[132] [PROFIsafeV2]

[133] konsequenter Weise müsste [PROFIsafeV2] an dieser Stelle $t_{wc\text{-}logic}$ nutzen.

[134] konsequenter Weise müsste [PROFIsafeV2] an dieser Stelle $t_{wc\text{-}cyc\text{-}out}$ nutzen.

[135] Device Acknowledge Time [PROFIsafeV2]

[136] Host Acknowledge Time [PROFIsafeV2]

[137] [PROFIsafeV2] Kapitel 9.3.3

Die PROFIsafe Spezifikation[138] setzt der Erhöhung von $t_{com}$=*F_WD_Time* in ihrer aktuellen Version (Draft-V2.5c) jedoch eine empfohlene obere Grenze mit maximal 30% über der minimalen *WCDT* und setzt weiterhin voraus, dass *DAT*, *HAT* und *Bus-Cycle* garantiert werden können. Die negativen Auswirkungen auf die Verfügbarkeit sind offensichtlich.

Für eine funktionierende Parametrierung muss mindestens gelten:

$$t_{com} > 2 \bullet t_{cyc\text{-}in} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{logic} \qquad (4.17)$$

beziehungsweise für die Verbindung zur Ausgangskomponente

$$t_{com} > 2 \bullet t_{logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out} \qquad (4.18)$$

Berücksichtigt man weiter, dass die Verwendung einer typischen Zeit ($t_{cyc\text{-}in}$ , $t_{cyc\text{-}out}$, $t_{logic}$) nicht zulässig ist, ersetzt diese mit den jeweiligen Worst-Case Werten und nimmt für diese Werte an, dass im Allgemeinen $t_{logic} > x$ mit $x$ aus $\{2\bullet(t_{wc\text{-}cyc\text{-}in} - t_{cyc\text{-}in}),\ 2\bullet(t_{wc\text{-}logic} - t_{logic}),\ t_{cyc\text{-}out}\}$ ist, ergibt sich das Maximum in der *SFRT*-Gleichung zu:

$$max_{i=1..n} = WDTime_2 - WCDT_2 = t_{com} + t_{logic} \qquad (4.19)$$

Die *SFRT* mit einem Fehler berechnet sich für obige Annahme dann aus:

$$\begin{aligned} SFRT_1 \;=\; & 2 \bullet t_{cyc\text{-}in} + && \rightarrow WCDT_1 \\ & ( 2 \bullet t_{cyc\text{-}in} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{logic} ) + && \rightarrow WCDT_2 \\ & 2 \bullet t_{logic} + && \rightarrow WCDT_3 \\ & ( 2 \bullet t_{logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out} ) + && \rightarrow WCDT_4 \\ & 2 \bullet t_{cyc\text{-}out} + && \rightarrow WCDT_5 \\ & ( t_{com}+t_{logic} ) && \rightarrow max_i \\ =\; & 4 \bullet t_{cyc\text{-}in} + 4 \bullet Bus\text{-}Cycle + 7 \bullet t_{logic} + 4 \bullet t_{cyc\text{-}out} + t_{com} && (4.20) \end{aligned}$$

Falls im zu betrachtenden Szenario der Praxis die Annahme bezüglich $t_{logic}$ nicht zutrifft, so ist das jeweilige Maximum einzusetzen. Dies kann z.B. dann der Fall sein, wenn

$$2 \bullet t_{logic} < t_{wc\text{-}logic} \qquad (4.21)$$

ist.

Der Ansatz von PROFIsafe ist, das die maximale Reaktionszeit einer Komponente zur Berechnung der *SFRT* mit herangezogen wird. Wie zuvor ausgeführt, kann sich die maximale Reaktionszeit, sicherheitstechnisch unerkannt, bis auf die Worst-Case Reaktionszeit der Komponente ausdehnen.

Der größte Unterschied ergibt sich dabei typischerweise für die Werte von *Bus-Cycle* zu $t_{com}$ und von $t_{logic}$ zu $t_{wc\text{-}logic}$. Liegen Faktoren zwischen den jeweiligen Wertepaaren, so ist offensichtlich, dass die *SFRT* nicht eingehalten wird, wenn sich die realen Zeiten in Richtung der Worst-Case-Zeiten $t_{wc\text{-}logic}$ ,$t_{wc\text{-}cyc\text{-}in}$ ,$t_{wc\text{-}cyc\text{-}out}$ und insbesondere von *WDCT* zu *F_WD_Time* hin, entwickeln.

Vergleicht man die *SFRT* mit der Worst-Case1 Reaktionszeit

$$t_{wc1} \geq t_{error\text{-}a} = 4 \bullet t_{com} + 2 \bullet t_{wc\text{-}logic} \qquad (4.22)$$

und setzt für

$$t_{com} = (2 \bullet t_{logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out}) \qquad (4.23)$$

ein, so erhält man mit der Annahme $t_{cyc\text{-}in} \leq t_{cyc\text{-}out}$ :

$$t_{error\text{-}a} = 4 \bullet (2 \bullet t_{logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out}) + 2 \bullet t_{wc\text{-}logic} \qquad (4.24)$$

[138] [PROFIsafeV2]

$$\begin{aligned} t_{error\text{-}a} - SFRT_1 &= 8 \bullet t_{logic} + 8 \bullet Bus\text{-}Cycle + 8 \bullet t_{cyc\text{-}out} + 2 \bullet t_{wc\text{-}logic} - \\ &\quad 4 \bullet t_{cyc\text{-}in} - 4 \bullet Bus\text{-}Cycle - 7 \bullet t_{logic} - 4 \bullet t_{cyc\text{-}out} - t_{com} \\ &= (t_{wc\text{-}logic} - t_{logic}) + t_{wc\text{-}logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out} - 4 \bullet t_{cyc\text{-}in} \\ &\geq (t_{wc\text{-}logic} - t_{logic}) + t_{wc\text{-}logic} + 2 \bullet Bus\text{-}Cycle - 2 \bullet t_{cyc\text{-}in} \end{aligned} \quad (4.25)$$

Hierbei wurde angenommen, dass $t_{cyc\text{-}cin} \leq t_{cyc\text{-}out}$ ist.

Sobald die Zeit der Zyklen der Logik, der Buszyklus und der Output-Zyklus den Zyklus der Eingangskomponente dominieren, berechnet die *SFRT* gemäß PROFIsafe Spezifikation[139] einen zu kleinen Wert. Dies dürfte für die Mehrheit der Anwendungen zutreffen, da die Logikzeiten die Eingangszeiten überragen.

Vergleicht man die *SFRT* mit der Worst-Case2 Reaktionszeit

$$t_{wc2} \approx t_{error\text{-}g} = 4 \bullet t_{com} + 2 \bullet t_{wc\text{-}logic} + 3 \bullet t_{cyc\text{-}out} + t_u \quad (4.26)$$

und setzt

$$t_{com} = (2 \bullet t_{logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out}) \quad (4.27)$$

ein, so erhält man mit der Annahme $t_{cyc\text{-}in} \leq t_{cyc\text{-}out}$:

$$\begin{aligned} t_{error\text{-}g} &= 4 \bullet (2 \bullet t_{logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out}) + 2 \bullet t_{wc\text{-}logic} + 3 \bullet t_{wc\text{-}cyc\text{-}out} + t_u \\ &= 8 \bullet t_{logic} + 8 \bullet Bus\text{-}Cycle + 8 \bullet t_{cyc\text{-}out} + 2 \bullet t_{wc\text{-}logic} + 3 \bullet t_{wc\text{-}cyc\text{-}out} + t_u \end{aligned} \quad (4.28)$$

Kann der Anwender die Ein-Fehlerannahme der *SFRT* für PROFIsafe nicht sicherstellen, so müsste sich die $SFRT_m$ wie folgt berechnen[140]:

$$SFRT_m = \sum_{i=1}^{5} WDTime_i \quad \text{mit mehreren Fehlern} \quad (4.29)$$

$$\begin{aligned} SFRT_m &= (t_{com} + 2 \bullet t_{wc\text{-}cyc\text{-}in}) + && \rightarrow WDTime_1 \\ &\quad (t_{com} + 2 \bullet t_{cyc\text{-}in} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{logic} + t_{logic}) + && \rightarrow WDTime_2 \\ &\quad (t_{com} + 2 \bullet t_{wc\text{-}logic}) + && \rightarrow WDTime_3 \\ &\quad (t_{com} + 2 \bullet t_{logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out} + t_{cyc\text{-}out}) + && \rightarrow WDTime_4 \\ &\quad (t_{com} + 2 \bullet t_{wc\text{-}cyc\text{-}out}) && \rightarrow WDTime_5 \\ &= 2 \bullet t_{cyc\text{-}in} + 2 \bullet t_{wc\text{-}yc\text{-}in} + \\ &\quad 4 \bullet Bus\text{-}Cycle + \\ &\quad 2 \bullet t_{wc\text{-}logic} + 5 \bullet t_{logic} + \\ &\quad 5 \bullet t_{com} + \\ &\quad 2 \bullet t_{wc\text{-}cyc\text{-}out} + 3 \bullet t_{cyc\text{-}out} \\ &= 2 \bullet t_{cyc\text{-}in} + 2 \bullet t_{wc\text{-}cyc\text{-}in} + \\ &\quad 14 \bullet Bus\text{-}Cycle + \\ &\quad 2 \bullet t_{wc\text{-}logic} + 15 \bullet t_{logic} + \\ &\quad 2 \bullet t_{wc\text{-}cyc\text{-}out} + 13 \bullet t_{cyc\text{-}out} \end{aligned} \quad (4.30)$$

$$t_{error\text{-}g} - SFRT_m = t_{wc\text{-}cyc\text{-}out} - 7 \bullet t_{logic} - 6 \bullet Bus\text{-}Cycle - 1 \bullet t_{wc\text{-}cyc\text{-}in} - 2 \bullet t_{cyc\text{-}in} - 5 \bullet t_{cyc\text{-}out}$$

(hier ist $t_u$ gegen $t_{cyc\text{-}in}$ aufgerechnet worden) (4.31)

---

139 [PROFIsafeV2]

140 [PROFIsafeV2] setzt in seinen Berechnungen nur die $SFRT_1$ an.

Bei der Annahme von Common-Cause Fehlern, berechnet die an der PROFIsafe Spezifikation[141] angelehnte Berechnung der $SFRT_m$ gegenüber einer optimierten Implementierung $t_{error\text{-}g}$ und $t_{wc2}$ eine deutlich zu große Reaktionszeit.

Vergleicht man die von PROFIsafe definierte Berechnung gemäß $SFRT_1$, so ergibt sich ein Delta von:

$$\begin{aligned} t_{error\text{-}g} - SFRT_1 &= (8 \bullet t_{logic} + 8 \bullet Bus\text{-}Cycle + 8 \bullet t_{cyc\text{-}out} + 2 \bullet t_{wc\text{-}logic} + 3 \bullet t_{wc\text{-}cyc\text{-}out} + t_u) \\ &\quad - (4 \bullet t_{cyc\text{-}in} + 4 \bullet Bus\text{-}Cycle + 7 \bullet t_{logic} + 4 \bullet t_{cyc\text{-}out} + t_{com}) \\ &= 1 \bullet t_{logic} + 4 \bullet Bus\text{-}Cycle + 4 \bullet t_{cyc\text{-}out} + 2 \bullet t_{wc\text{-}logic} + 3 \bullet t_{wc\text{-}cyc\text{-}out} + t_u \\ &\quad - 4 \bullet t_{cyc\text{-}in} - t_{com} \end{aligned} \qquad (4.32)$$

Setzt man wie oben $t_{com} = (2 \bullet t_{logic} + 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out})$, so erhält man:

$$\begin{aligned} t_{error\text{-}g} - SFRT_1 &= 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out} + 2 \bullet t_{wc\text{-}logic} + 3 \bullet t_{wc\text{-}cyc\text{-}out} + t_u - 1 \bullet t_{logic} - 4 \bullet t_{cyc\text{-}in} \\ &\geq 2 \bullet Bus\text{-}Cycle + 2 \bullet t_{cyc\text{-}out} + 2 \bullet t_{wc\text{-}logic} + 3 \bullet t_{wc\text{-}cyc\text{-}out} - 1 \bullet t_{logic} - 3 \bullet t_{cyc\text{-}in} \\ &\geq 2 \bullet Bus\text{-}Cycle + 1 \bullet t_{wc\text{-}logic} + 2 \bullet t_{wc\text{-}cyc\text{-}out} \end{aligned} \qquad (4.33)$$

Dabei wurde wieder angenommen, dass $t_{cyc\text{-}cin} \leq t_{cyc\text{-}out}$ und damit auch $t_{cyc\text{-}cin} \leq t_{wc\text{-}cyc\text{-}out}$ ist und daraus folgt auch, dass $t_{cyc\text{-}cin} = t_u$ ist.

Das bedeutet, PROFIsafe berechnet für den Mehrfehler- oder Common-cause Fehlerfall eine deutlich zu kleine „Safety-Function-Response-Time“.

Der Berechnungsmethode der *SFRT* in PROFIsafe macht, abgesehen von der Ein-Fehlerannahme, eine weitere wesentliche falsche Annahme. Für die Betrachtung wird in der PROFIsafe V2-Spezifikation[141] die Nutzung von sicherheitstechnisch nicht limitierten Zeiten ( *Bus-Cycle, DAT, HAT*, $t_{logic}$, $t_{cyc\text{-}in}$, $t_{cyc\text{-}out}$ ) herangezogen. Diese können sich, wie bereits erwähnt, sicherheitstechnisch unerkannt verändern und damit zu einer falschen *SFRT* führen.

PROFIsafe verwendet für die Authentizität einer Nachricht, die 16-Bit Adresse der Sicherheitssteuerung und die 16-Bit Adresse der Eingangs-/Ausgangskomponente. Diese werden mit in die Generierung des Presets[142] zur CRC-Berechnung einbezogen. Da die Berechnung für den Preset nur 16 Bit berücksichtigt, können durch den Preset maximal 65.536 verschiedene Sicherheitssteuerung – Eingangs-/Ausgangskomponentenpaare identifiziert werden. Diese werden nochmals um zwei Paare reduziert, um die Werte 0 und 0xffff auszusparen.

Die Wahrscheinlichkeit, dass ein Adresspaar zufällig den gleichen Preset berechnet, multipliziert mit der Auftrittswahrscheinlichkeit für falsches Routing[143], berechnet sich mit:

$$(10^{-6})h^{-1} \bullet 2^{-16} = 1{,}525 \bullet 10^{-11} h^{-1} \leq 1 \bullet 10^{-9} h^{-1} = 1\% \ \ von \ \ SIL3 \qquad (4.34)$$

Da der Preset nicht nur von den Adresspaaren abhängig ist, kann selbst das Engineering-Tool keine geeigneten Maßnahmen treffen, dass nur eineindeutige Adresspaare zum Einsatz kommen.

Ein falsches Routing während der Verbindungsaufnahme wird an den dabei übertragenen Adresspaaren erkannt, so dass die Fehlerbetrachtung für nach der Verbindungsaufnahme stattfindenden Routing-Fehler anzuwenden ist.

Das hier zur Anwendung kommende patentrechtlich geschützte Verfahren[144] weist in dieser Anwendungsart Schwachstellen auf, weil

1. der CRC-Preset kürzer als die Adressinformation ist und damit die Restfehlerwahrscheinlichkeit gegenüber der originären Übertragung der Adressinformation erhöht.

---

[141] [PROFIsafeV2]

[142] In [PROFIsafeV2] CRC1 genannt

[143] In Ermangelung von Vergleichszahlen wurde hier die Versagenswahrscheinlichkeit von Switches gemäß [PROFIsafeV2] eingesetzt.

[144] [Bart99b]

2. in den CRC-Preset weitere Informationen, die F-Parameter, mit eingerechnet werden und durch die zusätzliche Streuung nicht eineindeutige CRC-Presets entstehen.

Die Wahrscheinlichkeit, dass z.B. in einer Fertigungsanlage mit heute üblicher Anzahl von 500 oder gar 700 F-Modulen ein bestimmter Preset für die CRC Rechnung entsteht, ist, wenn Gleichverteilung angenommen wird, 1%. Das heißt, dass mit ca. 0,1% Wahrscheinlichkeit ein Preset zweimal in einer Anlage verwendet wird.

Falls eine solche Situation auf Grund der Anlagenkonfiguration gegeben sind, so besteht für die Kommunikationsverbindungen mit identischem Preset die Gefahr, dass die PROFIsafe Nachrichten nicht mehr eineindeutig einer Verbindung zugeordnet werden können. Die Wahrscheinlichkeit für zufällig gleiche Presets ist damit gleich 1 für ein solches Paar von F-Modulen. Somit hängt die Auftrittswahrscheinlichkeit nur noch vom falschen Routing der nicht sicherheitsgerichteten Standardübertragungsverfahren ab. Ebenso ist es denkbar, dass die PROFIsafe Nachrichten und die zugehörigen F-Parameter von unterschiedlichen F-Modulen stammen, da ein F-Modul zwar die zu ihm passenden F-Parameter anhand der darin enthaltenen F-Adresse prüfen kann, aber die Authentizität der PROFIsafe Nachrichten eben nur durch den Preset für die CRC Rechnung erkennbar ist.

Es gibt einen Sonderfall, bei dem der CRC-Preset aller F-Module immer unterschiedlich ist. Dazu darf im Netzwerk nur ein F-Host definiert sein und die F-Parameter der F-Devices dürfen sich nur in der F-Device-Adresse unterscheiden. Parameter, wie z.B. F_WD_Time, müssen bei allen Verbindungen zu den F-Modulen gleich sein. Weiterhin dürfen dann die F-Module keinen iPAR-CRC verwenden. Da sich in diesem Fall nur die 16 Bit F-Device-Adressen in den F-Parametern unterscheiden, berechnet ein 16-Bit CRC immer unterschiedliche Ergebnisse, d.h. einen unterschiedlichen CRC-Preset.

Eine weitere Schwachstelle der Adressierungstechnik von PROFIsafe ist, dass die „Adresse“, d.h. der CRC-Preset für Nachrichten von Sicherheitssteuerung an Ein-/Ausgangskomponente und von dieser zurück zur Sicherheitssteuerung gleich ist. Damit kann die Sicherheitssteuerung eine von ihr generierte Nachricht nicht von der von einer Komponente generierten Nachricht unterscheiden, wenn die Datenlänge für Eingangsdaten und Ausgangsdaten gleich ist. In diesem Fall ist die Nachricht der Sicherheitssteuerung eine gültige Antwort der Komponente. Die Konsequenz ist, dass die Sicherheitssteuerung falsche Eingangsdaten benutzt und dass die F_WD_Time nicht abläuft, obwohl keine Nachricht vom F-Modul der Ein-/Ausgangskomponente bei der Sicherheitssteuerung eintrifft[145].

Für die Ein-/Ausgangskomponente stellt im oben genannten Fall ihre eigene Nachricht solange eine gültige Wiederholung dar, bis die F_WD_Time abgelaufen ist oder eine neue Nachricht Sicherheitssteuerung empfangen wird. Die neue Antwort auf die Nachricht der Sicherheitssteuerung stellt jedoch wieder eine gültige Wiederholung dar. Da PROFIsafe es zulässt, dass die Daten einer Wiederholung geändert sein dürfen und diese geänderten Daten verwendet werden dürfen, benutzt die Komponente falsche Ausgangsdaten.

Entstehen kann eine solche Situation durch einen Fehler im Standard-Übertragungssystem, wozu auch die Backplane und die Busanschaltung der Sicherheitssteuerung und Ein-/Ausgangskomponenten gehören. Sehr einfach ist der Fehler z.B. im PROFINET/PROFIBUS Protokollstack eines Standard-Feldbuskopplers denkbar, bei dem die Adresse, die für die PROFINET/PROFIBUS Output-Daten des F-Moduls verwendet werden soll, fälschlicherweise auf die PROFINET/PROFIBUS Input-Daten gesetzt wird, während die PROFIsafe Verbindung etabliert ist. Dabei muss es sich nicht einmal zwingend um Software Fehler handeln, auch die Busanschaltungen mit einem PROFIBUS-Chip oder einem PROFINET-Chip können derartige Fehler verursachen, da auch in ihnen die Daten für F-Input und F-Output gespeichert sind.

[145] Dieses Verhalten wurde durch einen entsprechenden Versuchsaufbau mit einem modifizierten PROFINET-Device-Stack und einem F-Host in der Sicherheitssteuerung Siemens S7-315F / FW 2.6.7 bestätigt.

Eine weitere Fehlerursache kann durch die Diagnosefunktionen von Ethernet-Bausteinen entstehen. Die Bausteine ermöglichen es zu Testzwecken die ausgehenden Nachrichten nach innen zu spiegeln. Somit könnte z.B. die Sicherheitssteuerung eine von ihrer PROFINET Anschaltung versendeten Nachricht wieder empfangen und wenn nun auch die PROFINET Mechanismen die Nachricht gültig erscheinen lassen, wird sie dem PROFIsafe Protokoll zugestellt. Aus sicherheitstechnischer Sicht dürfen die Mechanismen von PROFINET jedoch für die Betrachtung nicht herangezogen werden.

Abschließend bleibt festzustellen, dass die Adressierungssicherheit bei PROFIsafe nicht mit SIL3 Qualität gegeben ist.

Im betrachteten Einsatzgebiet von PROFIsafe gibt es eine Mischung von Safety- und Standardnachrichten. Somit ist ein Schutz zur Unterscheidung der beiden Klassen von Nachrichten erforderlich. Dazu verwendet PROFIsafe folgende Techniken.

Bei der CRC Rechnung für Sicherheitsverbindungen wird ein spezieller Preset für den CRC-Algorithmus verwendet.

Da der Preset in den nicht sicherheitsgerichteten Komponenten nicht bekannt ist, ist nicht anzunehmen, dass diese eine korrekte Prüfsumme für eine sicherheitsgerichtete Nachricht erzeugen können. Gleichbedeutend mit der Verwendung eines anderen Presets, ist die zufällige Berechnung derselben Prüfsumme, wie dies durch den für den sicheren Verbindungstyp richtigen Preset erfolgen würde. Zusätzlich berechnet PROFIsafe den CRC über das Datenpaket von hinten nach vorne, was sehr unüblich ist und damit ein weiteres Unterscheidungsmerkmal darstellt.

Zweitens wirkt die Dynamik der erwarteten vconsnr im CRC. Um jeweils korrekte CRCs zu berechnen, muss die vconsnr bekannt sein.

Damit trotz dieser Maßnahmen ein Masquerading nicht erkannt wird, muss zu einem gegebenen Datenpaket zufällig ein Datenwort als CRC eingefügt werden, das zum Datenpaket und dem erwarteten Preset passt und das Toggle-Bit und activate_FV=0 müssen ebenfalls passen.

Der Wertebereich für CRC + 2 Bits erfüllt eine Restfehlerwahrscheinlichkeit pro Nachricht von

$$CRC-a: \frac{1}{2^{(24+2)}} = 1{,}490 \bullet 10^{-8} \qquad (4.35)$$

$$CRC-b: \frac{1}{2^{(32+2)}} = 5{,}820 \bullet 10^{-11} \qquad (4.36)$$

Durch die Abschaltung der Verbindung durch die Sicherheitssteuerung – Eingangs-/Ausgangskomponenten und der für den Wiederanlauf erforderlichen manuellen Quittierung für die Sicherheitssteuerung ist anzunehmen, dass die erste empfangene Nachricht eines Nicht-Sicherheits-Geräts zur Sicherheitsreaktion führt.

Beim Einsatzgebiet von PROFIsafe handelt es sich um eine geschlossene Übertragungseinrichtung, sofern diese ordnungsgemäß aufgebaut wird. Da dies jedoch in den Bereich der vorhersehbaren Fehlbedienung beim Aufbau oder Umbau fällt, werden die Fehlermöglichkeiten offener Kommunikationseinrichtungen nachfolgend betrachtet.

Betrachtet werden hier Bedienungsmöglichkeiten die dazu führen könnten, dass die Mechanismen von PROFIsafe außer Kraft gesetzt werden. Die Bedieneingriffe bei PROFIsafe beschränken sich im Wesentlichen auf den Aufbau der Kommunikationseinrichtungen, sowie deren Parametrierung.

Um diesbezügliche Schutzmaßnahmen zu erreichen, wird dem Anwender von PROFIsafe Installationen empfohlen, geeignete Zugangsmaßnahmen und Zugriffsschutz für die Knoten und deren Netzwerk zu realisieren. Dies liegt jedoch außerhalb der Betrachtungsebene von PROFIsafe und leitet sich aus den allgemeinen Anforderungen der IEC 61508 ab.

Offene Übertragungssysteme haben, neben der Gefährdung des Zugriffs, auch die Eigenschaft nicht definieren zu können, welche anderen Protokolle benutzt werden und müssen daher die Aspekte des Masqueradings betrachten.

Das Nachrichtenformat von PROFIsafe nutzt keine in der Standardübertragung PROFINET und PROFIBUS-DP eingesetzten Mechanismen. Da der PROFIsafe CRC-a jedoch auch in Standardbussystemen mit dem Flexray-Protokoll[146] eingesetzt wird, könnte hier ein Gefährdungspotential vorhanden sein, wenn Flexray im selben Netzwerk eingesetzt wird. Im Gegensatz zum Flexray-Protokoll[147] berechnet PROFIsafe den CRC-a jedoch nicht in der Bitorder, wie das Paket auf dem Netzwerk übertragen wird, sondern berechnet ihn von hinten nach vorne[148]. Zusätzlich unterscheiden sich die Berechnungsmethoden noch durch ihren Preset.

Durch die Verwendung der dynamischen vconsnr ist eine zusätzliche Wahrscheinlichkeit gegeben, dass eine Nachricht einen, aus PROFIsafe Sicht falschen, CRC aufweist und damit die Sicherheitsreaktion einsetzt.

In PROFIsafe wird für den V2-Mode keine Einschränkung der Verwendung von Sicherheits-CRCs innerhalb des Black-Channels gefordert, wie dies im Gegensatz für den V1-Mode gilt[148]. Da dies jedoch vor dem Hintergrund der verbesserten Bitfehlererkennung geschah, sollte überdacht werden, ob auch für den V2-Mode eine Einschränkung angebracht ist, dass Standard-Komponenten im selben Netzwerk keine Sicherheits-CRCs und die selbe Berechnungsmethode einsetzten dürfen.

So wie im vorangegangenen Abschnitt die versehentliche Mischung von Standard- und Safety-Nachrichten ausgeschlossen wurde, so ist sie doch als absichtliche Beeinflussung durchaus zu betrachten.

Da PROFIsafe nicht für offene Übertragungseinrichtungen ausgelegt wurde, wurden keine spezifischen Mechanismen, zum Schutz vor absichtlicher Unterminierung der Sicherheitsmechanismen des Protokolls in den Funktionsumfang des Protokolls aufgenommen.

Wenn der Angreifer die Hochfahrsequenz aufzeichnet und daraus den iPar-CRC und den CRC1 direkt ermittelt[149], kann er das Hochfahren durch das Einschleusen einer ungültigen Nachricht erreichen, auf die dann das Wiederanlaufverfahren ausgelöst wird. Dabei macht sich der Angreifer folgendes Verhalten des F-Devices zu nutze.

Den Anlauf, bzw. Wiederanlauf, signalisiert ein F-Host durch das Setzen des Steuerbits zum Rücksetzen der Consecutive-Number (Flag: R_cons_nr). In diesem Fall wird die Consecutive-Number 0 verwendet und vom F-Device ungeachtet der zu diesem Zeitpunkt gültigen Consecutive-Number als gültig erachtet. Wenn eine solche Nachricht während einer bestehenden Verbindung das F-Device erreicht und in dieser Nachricht ist das Steuerbit Use-Fail-Safe-Data (Flag: activate_FV) nicht gesetzt, so gibt das F-Device die in der Nachricht enthaltenen Daten aus, obwohl nicht gesichert ist, dass es sich um einer Nachricht handelt, die in korrekter Reihenfolge beim F-Device ankommt. Auf die Spitze getrieben, würde das F-Device sogar eine beliebig lange Folge von Nachrichten mit Reset-Consecutive-Number und den darin enthaltenen Prozessdaten akzeptieren und unberechtigter Weise ausgeben. Hier sollte die PROFIsafe Spezifikation vom F-Device fordern, dass die Fail-Safe-Daten ausgegeben werden, wann immer Reset Consecutive-Number (Flag: R_cons_nr) gesetzt ist.

Das Szenario stellt im eigentlichen Sinne keine Anforderung an PROFIsafe, sondern an den Zugriffsschutz des Systems als Ganzes. Hierbei ist einerseits der physikalische Zugriffsschutz der sicherheitsgerichteten Übertragungseinrichtungen, aber auch der Schutz über externe Übertragungseinrichtungen zu betrachten.

---

[146] [Flex05]

[147] [Flex05] Kapitel 4.5

[148] [PROFIsafeV2]

[149] Dies kann mit etwas mehr Aufwand während der normalen Verbindung geschehen.

Der Zugriffsschutz des Systems als Ganzes spielt auch bei der absichtlichen fehlerhaften Konfiguration eine entscheidende Rolle.

Ebenfalls zu betrachten ist die Unterminierung der Sicherheitsmechanismen, die Sabotage der Funktion als Ganzes. Dafür kommen Techniken für den Denial-of-Service Angriff auf ein oder mehrere Geräte im System durch die Generierung von Überlast mittels vieler Nachrichten pro Zeiteinheit in Frage oder noch einfacher, wie oben angedeutet, durch das Zusenden ungültiger Nachrichten.

Ebenfalls denkbar ist eine Unterbindung der PROFIsafe Funktion durch MAC Spoofing, um die Unicast Kommunikationsverbindungen durch reguläre Switch-Funktionen zu stören oder ganz zu unterbinden.

Die fehlerhafte Konfiguration bei PROFIsafe kann in verschiedenen Bereichen geschehen.

Als Erstes ist dies die fehlerhafte Adressierung der Geräte. Werden baugleiche Eingangs- / Ausgangskomponenten bezüglich der F-Adresse vertauscht, so kann PROFIsafe dies nicht aufdecken. Hier ist der Anwender gefordert, bei der Inbetriebnahme, eine entsprechende Verifikation durchzuführen. Die nicht sicherheitsgerichteten Adressierungstechniken helfen jedoch ebenfalls diese Fehler aufzudecken.

Eine weitere Gefährdung besteht, wenn der Anwender die *F_WD_Time* zu hoch einstellt. Wie in der allgemeinen Betrachtung bereits ausgeführt, kann der Anwender dies nicht durch seine Inbetriebnahmetests, sondern nur die organisatorische Maßnahmen der Projektierungsverifikation aufdecken.

Zur Inbetriebnahme und zu Wartungszwecken von PROFIsafe Anlagen, kann die Prüfung von iParametern ausgeschaltet werden, bzw. in einer der nächsten Versionen von PROFIsafe die *F_WD_Time_2* aktiviert werden. Unterlässt der Anwender dies nach der Inbetriebnahme oder nach der Wartung zurückzusetzen, so gefährdet er damit seine Anlage.

Ein Missbrauch der Verwendung von PROFIsafe ist insbesondere durch eine absichtliche fehlerhafte Konfiguration der Überwachungszeiten (*F_WD_Time*) vorstellbar.

Der Anwender hat die Möglichkeiten das Protokoll PROFIsafe zu umgehen, indem er die unterlagerten Standardbussysteme PROFINET-IO oder PROFIBUS-DP nutzt.

Diese Möglichkeiten des Missbrauchs kann natürlich nur durch entsprechende Maßnahmen des Anwenders und gegebenenfalls durch die überwachende Stelle und nicht durch das Protokoll verhindert werden.

Absehbare Fehlbedienung ist vorstellbar durch:

1. Verwendung nicht eineindeutiger Adressen, durch Hinzufügen solcher PROFIsafe Geräten in ein Übertragungssystem.
2. Zusammenschalten von zwei oder mehr Systemen/Teilsystemen mit identischen Adressen, wodurch ein Gesamtsystem mit nicht eineindeutigen Adressen gebildet wird.

Gegenüber den im Abschnitt zum Masquerading aufgeführten Gefahren, die Prüfungen von iParametern und *F_WD_Time* zu unterlassen, sind für das relevante Einsatzgebiet keine weiteren Betrachtungen erforderlich.

Zusammenfassend ergeben sich für den Anwender neben allgemeinen Anforderungen, die aus dem Einsatz von sicherheitsgerichteten Anwendungen herrühren, dafür zu sorgen, dass:

1. der physikalische Zugriffsschutz auf die sicherheitsgerichtete Übertragungseinrichtung gewährleistet ist,
2. der externe (Netzwerk) Zugriffsschutz auf die Mittel des Systems, nur berechtigten Personen Zugriff zu gewähren, wirksam ist,
3. der CRC1 verschiedener F-Devices in einem Netzwerk eineindeutig ist, damit eine fehlerhafte Adressierung erkannt werden kann,

4. die Eingangs- und Ausgangsdatenlängen unterschiedlich sind, solange die mögliche falsche Adressierung nicht durch ein verbessertes PROFIsafe Protokoll behoben wird,
5. das externe Netzwerk ausreichend geschützt ist, so dass von außen keine Angriffe und oder Fehlbedienungen auf dieses erfolgen können.
6. die Überwachungszeiten *F_WD_Time* passend zur sicherheitsgerichteten Anwendung parametriert sind,
7. die Übertragungseinrichtung eine Bitfehlerrate von $\leq 10^{-4}$ aufweist.

Die Entwicklung und Verifikation von PROFIsafe wurde durch eine Konzept-Prüfung durch den TÜV SÜD und die BGIA[150] verifiziert[151].

Die Entwicklung und Verifikation von Sicherheitssteuerung- und Eingangs-/Ausgangskomponenten-Software ist für die Qualifizierung nach IEC 61508 / SIL3 im Rahmen einer IEC 61508 / SIL3 konformen Produktentwicklung vorzunehmen und in einem Knoten einzusetzen, der aus Hardware- und Software-technischer Sicht nach den genannten Gesichtspunkten entwickelt wurde.

## 4.2 CIP-Safety-Protokoll Edition 1.1

CIP-Safety ist eines der wenigen sicherheitsgerichteten Protokolle, die eine sichere Multi-Cast Kommunikation ermöglichen. Das für CIP[152] übliche Kommunikationsmodell von Provider und Consumer kann auch für die Sicherheitsfunktionen eingesetzt werden.

Eine weitere Besonderheit von CIP-Safety ist die zeitliche Überwachung durch die Anwendung einer sicheren Zeitsynchronisation. Dazu wird vom Provider seine Zeit an die Consumer seiner Daten verteilt. Innerhalb der Datennachricht befindet sich der Zeitstempel des Providers, der den Entstehungszeitpunkt der Daten definiert. Wird die Nachricht mit den Daten beim Consumer empfangen, so kann er mit dem bei ihm geführten Zeitstempel des Providers prüfen, ob die Nachricht nicht zu alt ist. Zur weiteren zeitlichen Überwachung kommt, wie üblich, ein Watch-Dog-Timer zum Einsatz.

Zur Aufdeckung von Nachrichtenverfälschung werden CRCs, wie auch die doppelte Übertragung der Daten innerhalb einer Nachricht verwendet.

Für kurze Nachrichten, mit 1 oder 2 Byte Prozessdaten, verwendet CIP-Safety folgenden Nachrichtenrahmen:

*Tabelle 4.6: CIP-Safety kurzes Nachrichtenformat[153]*

| ***Actual Data*** | ***Mode Byte*** | ***CRC-S1*** | ***CRC-S2*** |
|---|---|---|---|
| sichere Prozessdaten<br>1 oder 2 Bytes | 1 Byte Status-<br>informationen | CRC8 | CRC8 |

Der CRC-S1 wird über die nicht übertragene Provider-Adresse, das Mode Byte, sowie die Prozessdaten berechnet. Der *CRC-S2* wird über das *Mode-Byte* und die invertierten Prozessdaten berechnet.

[150] heute IFA

[151] [SDSS07]

[152] Common Industrial Protocol

[153] [CIP5]

Für lange Nachrichten mit 3 bis 250 Byte Prozessdaten verwendet CIP-Safety folgenden Nachrichtenrahmen:

*Tabelle 4.7: CIP-Safety langes Nachrichtenformat*[153]

| ***Actual Data*** | ***Mode Byte*** | ***CRC-S3*** | ***Actual Data*** | ***Mode Byte*** | ***CRC-S3*** |
|---|---|---|---|---|---|
| sichere Prozessdaten 3 bis 250 Bytes | 1 Byte Status-informationen | CRC16 | inverse Prozessdaten | inverses Mode Byte | inverser CRC-S3 |

Der CRC-S3 wird über die nicht übertragene *Provider-Adresse*, das *Mode Byte* und die Prozessdaten berechnet. Die Elemente *Actual Data*, *Mode Byte* und *CRC-S3* werden doppelt übertragen.

Die zukünftigen Formate für den CIP-Safety Nachrichtenrahmen, auch solche für die Zeitsynchronisation, werden später vorgestellt.
Hier wird die CIP-Safety Variante für EtherNet/IP, gemäß Ed.1.1 der Spezifikation, betrachtet. Eine Betrachtung der Edition 2.2 von CIP-Safety folgt im Kapitel 4.3.

Zum besseren Verständnis werden hier die wesentlichen Sicherheitsmechanismen von CIP-Safety vorgestellt.

Für Prozessdatennachrichten mit 1-2 Byte Nutzdaten werden die Daten einmalig übertragen und durch 2 verschiedene 8-Bit CRCs, CRC-S1 und CRC-S2, gesichert. Zu den über die beiden CRCs gesicherten Daten gehören auch die Bits des Mode-Bytes.

Für Prozessdatennachrichten mit 3-250 Byte Nutzdaten werden die Daten doppelt, davon einmal invers, übertragen und durch einen 16-Bit CRC, CRC-S3, gesichert. Der CRC wird über die einfachen Nutzdaten incl. Mode-Byte gerechnet und selber ebenfalls doppelt, invers übertragen.

In jeder Prozessdatennachricht wird ein 16-Bit Time-Stamp mitgeschickt, der selbst wieder durch einen 8-Bit CRC-S1, in den der Producer-Identifier (PID) als Preset mit eingeht, gesichert ist. Die Time-Stamp Einheit ist 125 Mikrosekunden und hat einen darstellbaren Wertebereich von 8,3886 s[154].

Dieser Time-Stamp dient dem Empfänger dazu, das Alter einer Prozessdatennachricht zu bewerten.

Bei Single-Cast Verbindung repräsentiert der Time-Stamp-Wert eine Zeit bezüglich des Consumers. Bei Multi-Cast Verbindungen repräsentiert er eine Zeit relativ zum Producer.

Damit eine Altersabschätzung möglich ist, verwendet CIP-Safety ein Synchronisationsverfahren für die Time-Stamps, das in regelmäßigen Abständen (Ping-Intervall) wiederholt wird. Das zur Anwendung kommende Verfahren stellt sicher, dass die Verzüge und Ungenauigkeiten des Zeitabgleichs immer in die sichere Richtung gehen. Bezogen auf das Alter bedeutet dies, dass eine Nachricht im Zweifelsfall älter bewertet wird, als sie eigentlich ist.

Die Zeitsynchronisation wird durch den Producer mit einem neuen Ping-Count initiiert, worauf der angesprochene Consumer mit einer Time-Coordination-Nachricht antwortet. Dieser überträgt seine lokale Zeit im Time-Stamp-Format, zusammen mit dem vom Producer generierten 2-Bit Ping-Count. Time-Stamp und Ping-Count werden durch einen 16-Bit CRC (S3) gesichert, in den auch der Consumer-Identifier (CID) durch den Preset eingeht.

Der zweite Teil der Zeitsynchronisation ist für Single- und Multi-Cast unterschiedlich realisiert. Bei Single-Cast wird, wie bereits angedeutet, mit jeder Prozessdatennachricht der Time-Stamp mitgeschickt, der sich auf die lokale Zeit des Consumers bezieht. Bei Multi-Cast wird mindestens einmal je Ping-Intervall der Versatz zwischen Producer- und Consumer-Zeit für einen speziellen Consumer mit der Datennachricht mitgeschickt. Damit kann der Consumer die empfangene lokale Zeit des Producers in seine lokale Zeit umrechnen. Das bedeutet, dass im Multi-Cast Fall neben dem Time-Stamp des

[154] 128 Mikrosekunden • $2^{16}$

Producers, immer auch ein Korrektur-Time-Stamp, samt Consumer-Id für ***einen*** Consumer, in die Prozessdatennachricht gepackt wird. Diesen Teil der Nachricht nennt man Time-Correction. Er ist durch einen 16-Bit CRC-S3, in den auch die PID durch den Preset eingeht, abgesichert.

Das Expected Packet Interval (EPI) ist der zeitliche Abstand, mit dem Prozessdatennachrichten versendet werden. Gleichzeitig werden von dieser Größe zahlreiche andere Zeiten, z.B. das Ping-Intervall, abgeleitet. Das maximal verwendbare EPI ist gemäß CIP-Safety 1.000.000 Mikrosekunden. Da jedoch die Zeiten bei CIP-Safety in der Regel als Vielfache von 128 Mikrosekunden dargestellt werden, ergibt sich ein maximales EPI von 999.936 Mikrosekunden, was jedoch der Übersichtlichkeit wegen nicht weiter betrachtet wird.

Die Network Time Expectation (NTE) stellt bei CIP-Safety das maximal erlaubte Alter der Prozessdaten dar. Damit gleichbedeutend stellt sie auch die Zeit dar, nach der, nach einem Fehler, auf diesen reagiert wird. Der Wertebereich liegt zwischen 0 und 45313 in 128 Mikrosekunden, d.h. bis zu 5,800064 Sekunden.

In Abhängigkeit von der Implementierung, z.B. bei zum EPI asynchroner Prozessdatenverarbeitung, kommt noch ein Alter von 1 mal EPI zur NTE für die Prozessdaten hinzu.

Die Restfehlerrate pro Stunde empfangener, unerkannt verfälschter Nachrichten, ergibt sich mit

$$\Lambda = R(p, N, k) \cdot 3600 \cdot \nu \cdot c \qquad (4.37)$$

dabei sind

- $R(p,N,k)$ = Restfehlerrate bzgl. einer Nachricht
- $n+k$ = Anzahl Bits der Nachricht, bei CIP-Safety inklusive des Mode-Bytes

Der Wert von $c$ repräsentiert dabei die Ausführungen, dass sich eine Sicherheitsfunktion aus insgesamt $c$ Kommunikationsstrecken für die Übertragung der Eingangswerte zur Logikverarbeitung und von dort zur Ausgabeeinheit zusammen setzt.

Der Parameter $v$ repräsentiert die maximal angenommene Übertragungsrate in der Einheit Nachrichten je Sekunde. Im Gegensatz zu anderen Betrachtungen wurde hier mit $v$=100 zwar ein häufig nicht überschrittener Wert angesetzt, jedoch ist es mit bestehender Technologie durchaus möglich auch Systeme mit 5.000 bis 10.000 Nachrichten je Sekunde zu realisieren.

Beim realen Einsatz ist daher darauf zu achten, dass die Restfehlerrate bei gegebenem EPI nicht überschritten wird.
Setzt man als Modell für die Übertragungseinrichtung, den binär-symmetrischen Übertragungskanal voraus, so ergibt sich für CIP-Safety die bekannte Summenformel in zwei Ausprägungen.

## Lange Nachrichten

Die Nutzdaten für „lange“ Nachrichten bestehen aus mindestens 3 und maximal 250 Byte Prozessdaten[155], 3 Bits des Mode-Bytes und den 2 Bytes vom CRC-S3. Die gesamten Informationen werden bei langen Nachrichten in einer Nachricht doppelt übertragen.

Die Restfehlerwahrscheinlichkeit ergibt sich somit zu:

$$R_{lang}(p^2, N, k) \approx \frac{1}{2^k} \cdot \sum_{n=d}^{N+k} \binom{N+k}{n} \cdot (p^2)^n \cdot (1-p^2)^{(N+k-n)} \qquad (4.38)$$

Für CIP-Safety werden dabei für lange Nachrichten $p^2=(10^{-3})^2$, $N$=27...2003 Bit, $k$=16 Bit, $d$=4[156] angesetzt. Da die Daten bei CIP-Safety für lange Nachrichten zweifach übertragen werden, wird beim

[155] Bei Multi-Cast maximal 248 Bytes; dies wird jedoch nicht weiter betrachtet.

[156] Daten stammen von Rockwell-Automation, eine zusätzliche Verifikation der Hamming-Distanz aus der Literatur liegt nicht vor.

Empfang ein Vergleich der beiden Datensätze durchgeführt. Dies bewirkt im obigen Modell, dass $p$ bei unterstellter Bitfehlerrate von $10^{-3}$ in der Restfehlerrechnung mit $p^2 = 10^{-6}$ angesetzt werden kann. Der verwendete Faktor $1/2^k$ wird mit dem Hinweis auf den verwendeten *properen* CRC-S3 eingesetzt.

Somit folgt die Restfehlerrate pro Stunde mit

$$\Lambda_{lang} \le 1{,}0516 \cdot 10^{-17} \cdot 3600 \cdot 100 \cdot 62\,h^{-1} = 2{,}3462 \cdot 10^{-10}\,h^{-1} < 1 \cdot 10^{-9}\,h^{-1} = 1\%\ \ von\ \ SIL3 \qquad (4.39)$$

Mit dieser Abschätzung ergibt sich bei langen Nachrichtenlängen ein Faktor 4,26 für die Sicherheitsmarge. Da die Menge der Verbindungen hoch angesetzt ist, ergibt sich für reale Anwendungen mit 10 Verbindungen ein Faktor 26,4 als Sicherheitsmarge. Diese zeigt jedoch auch, dass für lange Nachrichtenlängen bei Erhöhung der Nachrichtenrate um mehr als eine Zehnerpotenz, Vorsicht geboten ist.[157]

### Kurze Nachrichten

Die Nutzdaten für „kurze“ Nachrichten bestehen aus 1 bis 2 Datenbytes für die eigentlichen Prozessdaten, 3 Bits des Mode-Bytes[158] und den beiden CRC8 Bytes vom CRC-S1 und CRC-S2.

$$R_{kurz}(p, N, k) = \sum_{n=1}^{N+k} A_n^{N+k} \cdot p^n \cdot (1-p)^{(N+k-n)} \qquad (4.40)$$

$A_n^{N+k}$ = *die Anzahl nicht erkannter verfälschten Nachrichten der Länge N+k mit n Bitfehlern*

Für kurze Nachrichten werden $p$=$10^{-3}$, $N$=19 Bit, $k$=16 Bit angesetzt. Für kurze Nachrichten werden die Daten nur einfach übertragen, daher bleibt $p$ bei $10^{-3}$. Es wird $k$=16 Bit gesetzt, da 2 unterschiedliche 8-bit CRCs zur Anwendung kommen.

Für kurze Nachrichten (=2 Nutzdatenbytes) wurden die Gewichte $A_n^{N+k}$ durch Simulation[159] der möglichen Bitfehler ermittelt. Mit diesen folgt $R(p,N,k)$=$7{,}02 \cdot 10^{-20}$ und somit

$$\Lambda_{kurz} \le 7{,}02 \cdot 10^{-20} \cdot 3600 \cdot 100 \cdot 62\,h^{-1} = 1{,}56 \cdot 10^{-12}\,h^{-1} < 1 \cdot 10^{-9}\,h^{-1} = 1\%\ von\ SIL3 \quad (4.41)$$

Mit dieser Abschätzung ergibt sich bei kurzen Nachrichtenlängen ein Faktor 641 für die Sicherheitsmarge. Da die Menge der Verbindungen hoch angesetzt ist, ergibt sich für reale Anwendungen mit 10 Verbindungen ein Faktor 3974 als Sicherheitsmarge.[160]

Die Erwartungshaltung für einen Time-Stamp liegt im Zeitintervall vom Time-Stamp, der zuletzt gültig empfangenen Nachricht und der aktuellen Zeit des Empfängers[161]. Dies trifft für Multi- und Single-Cast in gleicher Weise zu, nur dass der empfangene Time-Stamp bei Multi-Cast nach dem Empfang auf die lokale Zeit anzupassen ist. Dieses Intervall kann bis zur Network Time Expectation anwachsen, werden für diese Zeit keine gültigen Nachrichten empfangen.

Im Worst-case ergibt sich aus den maximalen Werten, für Network-Time-Expectation=5,8 s[162] und Time-Stamp-Wertebereich=8,3886 s[163], eine Wahrscheinlichkeit von 0,6914, dass ein mehr als

---

[157] Anmerkung: Zum Zeitpunkt der Entwicklung von CIP-Safety wurde von den Prüfbehörden nicht die heute vorwiegend benutzte Bitfehlerrate $10^{-2}$, sondern $10^{-3}$ für den Black-Channel verwendet.
Setzt man oben eine Bitfehlerrate von $10^{-2}$ ein, so erhält man für lange Nachrichten eine Restfehlerrate $\Lambda_{lang}$ pro Stunde von 0,01924, die für eine sicherheitsgerichtete Kommunikation für SIL1 nicht geeignet ist.

[158] Die Bits lauten Run_Idle, TBD1_Bit, TBD2_Bit.

[159] Werte stammen aus dem Rockwell Automation Dokument [X0323]

[160] Anmerkung: Setzt man auch hier die Bitfehlerrate von $10^{-2}$ an, so zeigt sich, dass auch das Sicherungsverfahren für kurze Nachrichten nicht für SIL1 geeignet ist.

[161] Time-Stamps dürfen nicht in der Zukunft liegen

[162] 128 Mikrosekunden • 45313

[163] 128 Mikrosekunden • $2^{16}$

8,3886 s alter Time-Stamp und damit auch die zugehörigen Prozessdaten, unerwünscht akzeptiert werden.

### Szenario A

Nimmt man den etwas realistischeren Fall für das Auftreten von Nachrichtenverlusten an, dass das Fenster für die Akzeptanz 3•EPI groß ist, so ergeben sich für EPI=10 ms, EPI=100 ms und EPI=1000 ms, Wahrscheinlichkeiten von $3{,}5763 \cdot 10^{-3}$, $3{,}5763 \cdot 10^{-2}$ bzw. $3{,}5763 \cdot 10^{-1}$. Der Faktor 3 ist auf die Annahme zurück zu führen, dass durch Nachrichtenverfälschungen, Nachrichten in einem Zeitintervall verloren gehen[164] und dann eine unerkannt verfälschte oder alte Nachricht eintrifft.

Der maximale Faktor für das Verwerfen von Nachrichten, bevor die Verbindung geschlossen wird, berechnet sich aus NTE/EPI.

*Soll z.B. eine Reaktionszeit eines Prozesses 1 s sein, so kann man für die Input-Daten die Network-Time-Expectation auf 300 ms setzen, die Sicherheitssteuerung Logikverarbeitung[165] auf 200 ms und wiederum für die Network-Time-Expectation 300 ms für die Output-Daten. Mit einer EPI von 100 ms ergibt sich somit der Faktor 3. Die in obiger Rechnung fehlenden 200 ms, ergeben sich aus der unterstellten, zum EPI asynchronen Verarbeitung der Sicherheitssteuerung.*

### Szenario B

Für eine von Nachrichtenverfälschungen bzw. anderen Fehlern unabhängige Betrachtung, ist das Zeitfenster mit 1 EPI anzunehmen. Daraus ergeben sich für die Akzeptanz eines beliebigen Time-Stamps die Wahrscheinlichkeiten $1{,}1321 \cdot 10^{-3}$ bis $1{,}1321 \cdot 10^{-1}$ für EPI=10 ms bis 1000 ms.

Hierbei wurden die durch die Zeitsynchronisation hervorrufen Verschiebungen des Time-Stamps nicht betrachtet. Diese Betrachtung machen ihn, wie schon erwähnt, älter, d.h. das Akzeptanzintervall wird größer als 1 EPI. Der Offset ist von der konkreten Anwendung und von der konkreten Implementierung abhängig. Da jedoch 1.000 ms als EPI sehr groß angenommen wurde, soll dieser Offset hier nicht weiter betrachtet werden.

Der Time-Stamp Teil der Nachricht besteht aus dem 16 Bit Time-Stamp und den restlichen 5 Bits des Mode-Bytes und 1 Byte CRC8 (CRC-S1). Damit ergibt sich $N$=21 und $k$=8. Die Größen $p$, $c$ und $v$ bleiben wie oben auf $p=10^{-3}$, $c$=62 bzw. $v$=100, da der Time-Stamp zusammen mit der Datensendung verschickt wird.

Aus der Simulation[166] der Bitfehler für die übertragenen 29 Bits, ergibt sich

$$R(p,N,k) = 1{,}10 \cdot 10^{-10}. \qquad (4.42)$$

Da jedoch eine gewisse Erwartungshaltung bzgl. des Time-Stamps beim Empfänger existiert, kann die Restfehlerrate weiter reduziert werden.

### Szenario C

Für den Fall a) mit 3 • EPI = 30 ms ergibt sich eine Restfehlerrate pro Nachricht von $3{,}5763 \cdot 10^{-3} \cdot 1{,}10 \cdot 10^{-10} = 3{,}9339 \cdot 10^{-13}$ und daher ein $\Lambda_{TS}$ von $8{,}78 \cdot 10^{-6}$ $h^{-1}$ ($v$=100).

Für den Fall b) mit 3 • EPI = 300 ms ergibt sich eine Restfehlerrate pro Nachricht von $3{,}5763 \cdot 10^{-2} \cdot 1{,}10 \cdot 10^{-10} = 3{,}9339 \cdot 10^{-12}$ und somit ein $\Lambda_{TS}$ von $8{,}78 \cdot 10^{-6}$ $h^{-1}$ ($v$=10).

---

[164] Z.B. durch die Ethernet-CRC Prüfungen der Übertragungseinrichtung

[165] 2 Zyklen

[166] Werte stammen von Rockwell Automation Dokument [X0323]

## Szenario D

Für Fall c) mit EPI = 10 ms bis 1000 ms ergeben sich die Restfehlerraten pro Nachricht von $1{,}1321 \cdot 10^{-3} \cdot 1{,}10 \cdot 10^{-10} = 1{,}2453 \cdot 10^{-13}$ bis $1{,}1321 \cdot 10^{-1} \cdot 1{,}10 \cdot 10^{-10} = 1{,}2453 \cdot 10^{-11}$ und daher ein $\Lambda_{TS}$ von $2{,}78 \cdot 10^{-6}$ $h^{-1}$.

## Worst-case

Beim Worst-case d) der Network-Time-Expectation ergibt sich eine Restfehlerrate pro Nachricht von $0{,}6914 \cdot 1{,}10 \cdot 10^{-10} = 7{,}6054 \cdot 10^{-11}$ und ein $\Lambda_{TS}$ von $1{,}6975 \cdot 10^{-5}$ $h^{-1}$ (v=1).

Die Fälle a) bis d) zeigen, dass selbst bei der hier angesetzten Bitfehlerrate von $10^{-3}$ die 1% der SIL3 Qualität auf diese Weise nicht erreicht wird.

Erst wenn zusätzlich die Auftrittswahrscheinlichkeit $P_d$ für eine verzögerte Nachricht ausreichend klein ist, erreicht man das Ziel von $10^{-9}$ $h^{-1}$.

Der von Rockwell Automation gewählte Ansatz, diese Auftrittswahrscheinlichkeit mit 1 Fehler je Nachricht, je Stunde führt bei einer Bitfehlerrate von $10^{-3}$ zu:

a) $P_d = 1/(100 \cdot 3600)$; $\Lambda_{TS} = 2{,}43 \cdot 10^{-11}$.

b) $P_d = 1/(10 \cdot 3600)$; $\Lambda_{TS} = 2{,}43 \cdot 10^{-10}$.

c) $P_d = 1/(100..1 \cdot 3600)$; $\Lambda_{TS} = 7{,}7210^{-12} ... 7{,}7210^{-10}$.

d) $P_d = 1/(1 \cdot 3600)$; $\Lambda_{TS} = 4{,}72 \cdot 10^{-9}$.

und kommt somit zu einer Qualität besser als 1% SIL3.

Nicht ganz unproblematisch ist einerseits die Definition der Auftrittswahrscheinlichkeit selbst und, noch viel problematischer, die unterstellte Unabhängigkeit der Einzelwahrscheinlichkeiten. So lässt sich in realen Umgebungen ein Zusammenhang zwischen der Auftrittswahrscheinlichkeit von Bit-Fehlern und der Verzögerung von Nachrichten nachweisen. Da die CIP-Safety unterlagerten Transportschichten, z.B. WLAN, Wiederholmechanismen und damit Verzögerungen aufgrund der Bit-Fehler, gemäß ihrer Sollfunktion hervorrufen.[167]

Eine Variation des Problems wird nachfolgend nochmals aufgegriffen.

Die Time Correction Daten werden nur bei Multicast genutzt und bestehen aus dem Multicast-Byte incl. Parity, einem 16 Bit Time-Stamp, einem 16 Bit CRC-S4 und einer kodierten Kopie des Multicast-Bytes, die jedoch nicht in den CRC eingeht. Damit ergibt sich N=24, k=16, sowie zusätzlich das kodierte Multicast-Prüfbyte.

Bezüglich der Nachrichtenverfälschung verhalten sich Time Correction und Time Coordination gleich. Die Nachrichtenverfälschung der Time-Coorection wird daher im nächsten Abschnitt mit betrachtet. Lediglich die Nachrichtenhäufigkeit ist eine andere und ergibt somit gegenüber der Time-Coordination Nachricht ein anderes Lambda.

$$\Lambda_{TCorr}(v = 100, c = 1) = 1{,}06 \cdot 10^{-11}\ h^{-1} \qquad (4.43)$$

*c* wurde mit 1 belegt, da die Time-Correction immer nur an einen Consumer geht. Dies genügt 1% von SIL3, auch wenn das EPI auf 1 ms reduziert wird, wodurch sich die Anzahl der Nachrichten verzehnfacht.

*Anmerkung: Für die Bitfehlerrate von $10^{-2}$ ist die angestrebte Restfehlerrate nicht für SIL3 geeignet.*

Die Time Coordination Nachricht besteht aus dem Ack-Byte incl. Parity, einem 16 Bit Time-Stamp, einem 16 Bit CRC-S4 und einer kodierten Kopie des Ack-Bytes, die jedoch nicht in den CRC eingeht. Damit ergibt sich N=24, k=16, sowie zusätzlich das kodierte Ack-Prüfbyte.

---

[167] Anmerkung: Für die Bitfehlerrate von $10^{-2}$ ist die angestrebte Restfehlerrate in den hier aufgezeigten Szenarien nicht für SIL3 geeignet.

Aus der Simulation[168] der Bitfehler für die übertragenen 48 Bits[169] ergibt sich für das Ack/Multi-cast-Byte ein $R(p,N,k)$ von $3{,}07 \cdot 10^{-23}$ und für den Time-Stamp ein $R(p,N,k)$ von $2{,}97 \cdot 10^{-17}$.

Für die Anzahl Nachrichten je Sekunde ergeben sich mit den CIP-Safety Regeln mit 19 • EPI für Single-cast und 100 • EPI für Multi-Cast also 1/19-tel bzw. 1/100-tel der Prozessdatennachrichtenmenge.

Somit ergibt sich $\Lambda_{TCoord}(v=100/19) = 3{,}48 \cdot 10^{-11}\ h^{-1}$ und für $\Lambda_{TCoord}(v=100/100) = 6{,}62 \cdot 10^{-12}\ h^{-1}$. Dies genügt 1% von SIL3, auch wenn das EPI auf 1 ms reduziert wird, wodurch sich die Anzahl der Nachrichten verzehnfacht.[170]

Zusammenfassend ergibt sich die Summe der Restfehlerraten der Nachrichtenverfälschung aus den einzelnen Fehlerraten.

$$\Lambda_{Summe\text{-}Multi\text{-}Cast} = \Lambda_{Data} + \Lambda_{TS} + \Lambda_{TimeCorr} + \Lambda_{TimeCoord}. \quad (4.44)$$

$$\Lambda_{Summe\text{-}Multi\text{-}Cast} = 2{,}34 \cdot 10^{-10}\ h^{-1} + 8{,}78 \cdot 10^{-6}\ h^{-1} + 1{,}06 \cdot 10^{-11}\ h^{-1} + 6{,}62 \cdot 10^{-12}\ h^{-1}$$

$$= 8{,}78 \cdot 10^{-6}\ h^{-1} \quad (4.45)$$

mit $p=10^{-3}$ und $v=100$

Die Restfehlerrate genügt damit selbst bei einer Bitfehlerrate von $10^{-3}$ nicht 1% von SIL3.

*Anmerkung: Im Falle von Single-cast entfällt $\Lambda_{TCorr}$.*

Nachrichtenverluste innerhalb der Network-Time-Expectation sind für CIP-Safety nicht relevant, solange die Zeiterwartung nicht verletzt wird. Wird diese verletzt, wird die Verbindung geschlossen. Damit beherrscht CIP-Safety die relevanten Verluste.

Zur Erkennung von Nachrichtenwiederholungen verwendet CIP-Safety das Nachrichtenfeld Time-Stamp.

### Unerwünschte Wiederholungen von einzelnen Nachrichten

Beim CIP-Safety Empfänger einer Nachricht besteht die Erwartungshaltung bezüglich des Time-Stamps. Das heißt, ein zu akzeptierender Time-Stamp muss größer als der zuletzt akzeptierte sein. Damit kann dieser Mechanismus mehrfach empfangene Nachrichten als solche erkennen. Die als Duplikat erkannten Nachrichten werden vom CIP-Safety Empfänger verworfen.

### Unerwünschte Wiederholungen von Nachrichtensequenzen und Wiederanlauf

Unterstellt man nun, dass eine Verbindungsaufbau-Nachrichtensequenz in den verwendeten Kommunikationseinrichtungen gespeichert wurde und nach dem Schließen der Verbindung wieder eingespielt wird, so kann der Empfänger nicht erkennen, dass es sich um Nachrichten einer alten Verbindung handelt. Eine Technik zur Erkennung von Nachrichten alter Verbindungen existiert in CIP-Safety nicht.

Für die Restfehlerwahrscheinlichkeit, dass eine solche Sequenz erkannt wird, ergibt sich 1 • Wahrscheinlichkeit für die Akzeptanz des Time-Stamps. Dieser Time-Stamp wird durch den Consumer vorgegeben und muss demnach zum in der gespeicherten Sequenz verwendeten Time-Stamp passen.

Bei Single-Cast ist dieser Time-Stamp so lange zulässig, als dass er sich im Zeitintervall bzgl. der lokalen Zeit des Consumers und der NTE befindet. Damit ist die Wahrscheinlichkeit, dass die Sequenz akzeptiert wird, NTE/8,388608 s Für NTE=100 bis 1000 ms ergeben sich Wahrscheinlichkeiten von 0,011 bis 0,11.

Bei Multi-Cast ist es nicht der Time-Stamp des Producers, sondern der Time-Stamp im TimeCorrection-Teil der Nachricht. Dafür gelten die gleichen Bedingungen, wie bei Single-Cast Time-Stamp. Es

[168] Werte stammen von Rockwell Automation Dokument [X0323]

[169] Identisch zu Time Correction

[170] Anmerkung: Für die Bitfehlerrate von $10^{-2}$ ist die angestrebte Restfehlerrate nicht für SIL3 geeignet.

ergeben sich auch hier Wahrscheinlichkeiten von 0,01 bis 0,11, dass eine Nachrichtensequenz zufällig passt.

Setzt man die erwartete Fehlerrate[171] mit $10^{-6}$ $h^{-1}$ an, so ergibt sich eine Restfehlerwahrscheinlichkeit pro Stunde von

$$0{,}11 \cdot 10^{-6} h^{-1} = 1{,}1 \cdot 10^{-7} h^{-1} \quad bzw. \quad 1{,}1 \cdot 10^{-8} h^{-1} > 1 \cdot 10^{-9} h^{-1} = 1\% \ von \ SIL3 \qquad (4.46)$$

1% von SIL3 ist damit nicht erreicht.

Bis jetzt noch nicht betrachtet wurde, dass sich die Verbindung im Zustand Offline befinden muss, damit ein Verbindungsaufbau durchgeführt werden kann. Dabei ist Wahrscheinlichkeits-technisch jedoch Vorsicht geboten, da die Unabhängigkeit nicht als gegeben angesehen werden darf. Wiederholungen finden insbesondere nach Störungen im Netz statt, die zuvor zum Schließen einer Verbindung geführt haben können.

Dass eine Nachrichtensequenz einer anderen Verbindung zu der hier betrachten Verbindung passt, hat zusätzlich eine Wahrscheinlichkeit, das der Unique-Network-Identifier (nachfolgend UNID) zufällig passt; was jedoch ein Konfigurationsfehler des Netzwerks wäre. Dies wird daher nicht weiter betrachtet.

Betrachtet wurden hier für CIP-Safety das Einfügen von Nachrichten während einer bestehenden Verbindung. Für den Zustand, dass keine Verbindung besteht, wurde schon oben, die Diskussion über die Wiederholung von Nachrichtensequenzen beim Starten bzw. Wiederanlaufen geführt.

Bei bestehender Verbindung können die Nachrichten aus vorangegangenen Verbindungen, wie auch aus der aktuell bestehenden bzw. sogar aus ganz anderen Verbindungen, herrühren.

Für die Nachrichten, die zu der bestehenden Verbindung gehören oder gehörten, ergibt sich die Restfehlerwahrscheinlichkeit, wie schon oben erläutert, in Abhängigkeit von der verwendeten EPI mit $1{,}1321 \cdot 10^{-3}$ bis $1{,}1321 \cdot 10^{-1}$ für EPI=10 ms bis 1.000 ms. Dem Sachverhalt, dass die eingefügte Nachricht mindestens 8,3 s alt sein muss, wird hier die Wahrscheinlichkeit 1 zugemessen.

Für Nachrichten aus anderen Verbindungen multipliziert sich die Wahrscheinlichkeit zusätzlich mit dem Preset für die CRC Berechnung ($=1{,}5268 \cdot 10^{-5} = 2^{-16}$), da der Producer mit seiner PID zufällig den gleichen Preset generiert haben muss.

Setzt man die erwartete Fehlerrate für das Wiederholen einer mindestens 8,3 s alten Nachricht mit $10^{-6}$ $h^{-1}$ an, so ergeben sich für Nachrichten der Verbindung Restfehlerwahrscheinlichkeiten pro Stunde von

$$\begin{aligned} 1{,}1321 * 10^{-3} \cdot 10^{-6} h^{-1} &= 1{,}1321 \cdot 10^{-9} h^{-1} > 1 \cdot 10^{-9} h^{-1} = 1\% \ von \ SIL3 \\ 1{,}1321 * 10^{-1} \cdot 10^{-6} h^{-1} &= 1{,}1321 \cdot 10^{-7} h^{-1} > 1 \cdot 10^{-9} h^{-1} = 1\% \ von \ SIL3 \end{aligned} \qquad (4.47)$$

SIL3 wird damit nicht erreicht.

1% SIL3 wird bezüglich dieser Maßnahme erst ab einer Fehlerrate in der Größenordnung von $10^{-7}$ $h^{-1}$ bis $10^{-9}$ $h^{-1}$ erreicht. Dies ist, zumindest für größere EPIs, bei den eingesetzten Übertragungseinrichtungen eine äußerst fragwürdige Annahme.

Hierbei ist noch unberücksichtigt, dass die Netzwerkkomponenten typischerweise in einer Linienstruktur aufgebaut werden und damit die Fehlerrate je Komponente, zwischen 10 bis 100 Stück, zu addieren ist. Bei 100 Komponenten müsste für SIL3 je Standard-Komponente eine Fehlerrate von $10^{-11}$ $h^{-1}$ angesetzt werden, was völlig utopisch ist.

Geht man weiterhin vom „Black-Channel“ Ansatz aus, so ist auch eine Fehlerrate von $10^{-6}$ $h^{-1}$ zu hinterfragen. Ein eher konservativer Ansatz mit 1 Fehler je Stunde ist hier aus sicherheitstechnischer Sicht angebracht, auch wenn damit EtherNet/IP-Safety nicht verfügbar arbeiten kann.

---

[171] Annahme der Fehlerrate für einen Ethernet-Switch-Baustein mit 100 FIT. Siehe auch [PROFIsafeV2]. Damit ein praxistauglicher Ansatz gewährleistet ist, werden 10 Ethernet-Switch-Bausteine angenommen.

Bei Nachrichten aus anderen Verbindungen ergibt sich entsprechend eine Wahrscheinlichkeit von $1{,}727 \cdot 10^{-12}$, also ausreichend für 1 % von SIL3.

Wenn Daten und Time-Stamp innerhalb einer empfangenen Nachricht, nicht zusammengehören, wird dies von CIP-Safety nicht erkannt. Der Fragmentierungsschutz von CIP-Safety ist daher ungeeignet.

Eine falsche Nachrichtenreihenfolge wird von CIP-Safety durch die Erwartungshaltung des Time-Stamps erkannt. Die Restfehlerrate, dass dies nicht erkannt wird, entspricht der Restfehlerrate eingefügter Nachrichten aus der gleichen Verbindung.

Die wiederholte Zeitsynchronisation von CIP-Safety deckt *schleichenden* zeitlichen Drift auf. Die verwendete Zeitsynchronisation ist derart ausgeführt, dass sie die anzunehmenden Fehler bei der Zeitsynchronisation jeweils zu Ungunsten des Nachrichtenalters heranzieht.

Die Adressierung wird durch den CID und den PID, die in den UNID eingehen, beim Verbindungsaufbau und den Konfigurations-CRC geprüft. Die Adressdaten werden in die Bestimmung des Startwerts für die CRC Berechnung mit eingebunden. Somit wird mit der Qualität der CRCs auch die Qualität der Adresssicherheit hergestellt.

Im betrachteten Einsatzgebiet von CIP-Safety gibt es eine Mischung von Safety- und Standardnachrichten. Somit ist ein Schutz zur Unterscheidung der beiden Klassen von Nachrichten erforderlich. Dazu verwendet CIP-Safety ein sehr spezielles Nachrichtenformat, dass sich von Standardformaten unterscheiden sollte. Zusätzlich wird die CRC Berechnung durch einen nicht übertragenen Informationsanteil (CID/PID) gegen Mischung geschützt.

Beim Einsatzgebiet von CIP-Safety handelt es sich um eine geschlossene Übertragungseinrichtung, sofern diese ordnungsgemäß aufgebaut wurden. Da dies jedoch in den Bereich der vorhersehbaren Fehlbedienung beim Aufbau oder Umbau betrachtet werden kann, werden die Fehlermöglichkeiten offener Kommunikationseinrichtungen nachfolgend betrachtet.

Die Forderung nach einer geschlossen Übertragungseinrichtung fehlt in der CIP-Safety Spezifikation[172].

Betrachtet werden hier Bedienungsmöglichkeiten, die dazu führen könnten, dass die CIP-Safety Mechanismen ausgehebelt werden. Die Bedieneingriffe bei CIP-Safety beschränken sich im Wesentlichen auf den Aufbau der Kommunikationseinrichtungen, Restart der Kommunikation, sowie deren Parametrierung.

Offene Übertragungssysteme haben, neben der Gefährdung des Zugriffs, auch die Eigenschaft, nicht definieren zu können, welche anderen Protokolle benutzt werden.

Die wesentliche signifikante Eigenschaft von CIP-Safety ist die Verwendung von spezifischen CRC16/CRC8 Polynomen und eines spezifischen Presets aus CID/PID, die soweit bekannt, nicht für andere Kommunikationszwecke angewendet werden. Zudem werden die CRCs bei langen Nachrichten nicht, wie üblich, am Ende der Nachricht, sondern inmitten der sicheren Nachricht übertragen. Somit müsste ein anderes Protokoll die selben CRCs, den selben Preset und die selben Positionen innerhalb der Nachricht nutzten.

Des Weiteren müssen noch die oben genannten Mechanismen, wie Time-Stamp, Mode-Byte, doppelte invertierte Nachrichteninhalte und Adressierung zur Erwartungshaltung des Empfängers passen, damit eine Nachricht eines anderen Protokolls als sichere Nachricht akzeptiert wird. Dies wird aus ausreichend komplex angesehen, dass es nicht zu der in diesem Kapitel angesprochenen Gefährdung kommen kann.

So wie im vorangegangenen Absatz die versehentliche Mischung von Standard- und Safety-Nachrichten ausgeschlossen wurde, so ist sie doch bei absichtlicher Beeinflussung durchaus zu betrachten.

---

[172] [CIP5]

Da CIP-Safety nicht für offene Übertragungseinrichtungen ausgelegt wurde, gibt es keine spezifischen Mechanismen, die zum Zweck des Schutzes vor absichtlicher Unterminierung von Sicherheitsmechanismen des Protokolls in den Funktionsumfang des Protokolls aufgenommen wurden.

Jedoch ist es trotzdem nicht einfach z.B. eine Nachricht in das Übertragungssystem einzuschleusen, die vom Empfänger als korrekt akzeptiert wird. Voraussetzungen dazu sind die Kenntnis der Funktionsweise von CIP-Safety, der physikalischer Zugang mit einem Mirroring-fähigen Switch in das Übertragungssystem und ein Kommunikationsknoten, der in der Lage ist den Nachrichtenverkehr in Echtzeit zu analysieren.

Insbesondere die Echtzeitfähigkeit des angreifenden Kommunikationsknotens stellt hohe technische Anforderungen an diesen, da der Knoten einen Time-Stamp generieren muss, der dem aktuellen Erwartungsstand des Empfängers entspricht. Trifft der Kommunikationsknoten nicht das Zeitfenster, in dem der Time-Stamp gültig ist, wird die Nachricht vom Empfänger verworfen.

Eine Unterminierung der Sicherheitsmechanismen von CIP-Safety ist am wahrscheinlichsten durch den Einsatz einer CIP-Safety Strategie umsetzenden Software in einem externen Gerät. Damit dies Aussicht auf Erfolg hat, reicht es nicht aus die Funktionsweise von CIP-Safety zu kennen, vielmehr müssen auch die etwaigen Parametrierung der zu kompromittierenden Geräte bzw. der daraus abgeleiteten CRC-Presets, bekannt sein. Möglich ist dies durch Aufzeichnen des Netzverkehrs, Herausnehmen eines CIP-Safety Knotens und Ersetzen desselben durch ein passendes Gerät.

Damit stellt dieses Szenario im eigentlichen Sinne keine Anforderung an CIP-Safety, sondern an den Zugriffsschutz des Systems als Ganzes. Hierbei ist einerseits der physikalische Zugriffsschutz der sicherheitsgerichteten Übertragungseinrichtungen, aber auch der über externe Übertragungseinrichtungen zu betrachten.

Der Zugriffsschutz des Systems als Ganzes spielt auch bei der absichtlichen fehlerhaften Konfiguration eine entscheidende Rolle.

Wahrscheinlicher als die Unterminierung der Sicherheitsmechanismen, ist die Sabotage der Funktion als Ganzes. Dafür in Frage kommen Techniken für den Denial-of-Service Angriff auf ein oder mehrere Geräte im System, durch die Generierung von Überlast mittels vieler Nachrichten pro Zeiteinheit.

Ebenfalls denkbar ist eine Unterbindung der CIP-Safety Funktion durch MAC Spoofing, um die Unicast Kommunikationsverbindungen durch reguläre Switch-Funktionen zu stören oder ganz zu unterbinden.

Am einfachsten ist es jedoch CIP-Safety-Nachrichten mit falschem CRC an einen Knoten zu schicken, der dann, den Regeln von CIP-Safety gehorchend, ab einer maximalen Rate[173] die Verbindung schließt.

Diese Arten der Sabotage führen jedoch nicht zum Versagen von CIP-Safety Sicherheitsmechanismen, sondern zum Schließen der Verbindung und damit zur sicherheitsgerichteten Reaktion.

Beim CIP-Safety liegt es in der Verantwortung des Anwenders jedem Gerät eine eineindeutige Adresse UNID zuzuteilen.

Eine Parametrierung von für den zu steuernden/überwachenden Prozess unzulässig große Network-Time-Expectation ist ebenfalls als fehlerhafte Konfiguration zu werten. Diese würden im Fehlerfall zu einer unzulässig langen Reaktionszeit führen.

Beide Möglichkeiten der fehlerhaften Konfiguration sind durch organisatorische Maßnahmen durch den Anwender zu beherrschen.

Ein Missbrauch der Verwendung von CIP-Safety ist insbesondere vorstellbar durch eine absichtliche fehlerhafte Konfiguration der Überwachungszeiten.

---

[173] Max_Fault_Count aus [CIP5-2.2]

Möglichkeiten das Protokoll CIP-Safety im vorgesehenen Einsatzgebiet zu umgehen, hat der Anwender nur dann, wenn die eingesetzten Geräte auch einen Standardzugriff mittels EtherNet/IP (ohne Safety) ermöglichen.

Beide Möglichkeiten des absehbaren Missbrauchs sind durch organisatorische Maßnahmen durch den Anwender zu beherrschen.

Eine absehbare Fehlbedienung ist vorstellbar durch:

1. Verwendung nicht eineindeutiger Adressen, beim Hinzufügen weiterer Geräte in ein Übertragungssystem
2. Zusammenschalten von zwei oder mehr Systemen mit identischen Adressen, wodurch ein Gesamtsystem mit nicht eineindeutigen Adressen gebildet würde.

Beide Möglichkeiten sind durch organisatorische Maßnahmen durch den Anwender zu beherrschen.

Zusammenfassend hat der Anwender, neben allgemeinen Anforderungen, die aus dem Einsatz von sicherheitsgerichteten Anwendungen herrühren, dafür zu sorgen, dass:

1. der physikalische Zugriffsschutz auf die sicherheitsgerichtete Übertragungseinrichtung gewährleistet ist,
2. der externe (Netzwerk) Zugriffsschutz Mittel des Systems einsetzt, die nur berechtigten Personen Zugriff gewähren,
3. das externe Netzwerk ausreichend geschützt ist, als das von außen Angriffe oder Fehlbedienungen auf dieses erfolgen können,
4. die Network-Expectation-Time passend zur sicherheitsgerichteten Anwendung parametriert ist.

### Zwischenergebnis

Auf Anregung des Autors wurde der Einsatz des CIP-Safety Kommunikationsprotokolls gemäß CIP-Vol-5/Ed1.1 für die Anwendung mit EtherNet/IP Übertragungssystemen eingeschränkt auf:

1. Verwendung von EPIs kleiner als 100 ms. Dies hat zum Ziel, die Restfehlerrate für Einfügungen zu reduzieren, indem die speichernden Komponenten besser mit Nachrichten durchflutet werden und veraltete Nachrichten nicht so lange darin verweilen können.
2. Verwendungen von speichernden Netzwerkkomponenten, die gegen interne Fehladressierung möglichst robust sind. Dies betrifft insbesondere Geräte mit SW basierten Kommunikationstechniken, wie Router oder nicht sicherheitsgerichtete Ethernet-Anschaltungen der sicheren Geräte.
3. Verwendung von Übertragungseinrichtung, die eine Nachrichtenfragmentierung an ungeraden Byte-Grenzen von Nachrichten nicht verursachen können, da sichere Daten und Time-Stamp immer an einer ungeraden Byte-Grenze aneinander stoßen.

## 4.3 CIP-Safety Änderungsvorschlag und Edition 2.2

Vorschlag des Autors für die Beherrschung der Schwächen des CIP-Safety Protokolls Edition 1.1[174]:

1. Einfügung von Nachrichten
2. Verfälschung des Time-Stamp Teils der Prozessdatennachricht
3. unerkannte Fragmentierung der Prozessdatennachricht

[174] [CIP5]

## Kurzes Nachrichten Format

| Data[16-536] | PingCount[2] | RunIdle[1] | TimeStamp[13] | CRC[32] |
|---|---|---|---|---|

*Abbildung 4.1: Änderungsvorschlag kurze Nachricht*

Die Zahlen in [] bezeichnen die Bits der jeweiligen Information.

***Beschreibung der Nachrichten-Felder:***

| | |
|---|---|
| Data[] | 16 bis 536 Bit Prozessdaten (2-67 Bytes) |
| PingCount | 2 Bit Wert mit der selben Bedeutung, wie das Mode-Byte in CIP-Safety Ed. 1.1 |
| RunIdle | 1 Bit Wert mit der selben Bedeutung, wie das Mode-Byte in CIP-Safety Ed. 1.1 |
| TimeStamp[13] | Niederwertige 13 Bits des 27 Bit Time-Stamps |
| CRC | 32 Bit CRC des properen Polynoms 0xA814_498F[175] in gespiegelter Form, mit der bekannten Hamming-Distanz von 8 für Nachrichten bis 1024 Bit Länge[176]. Die CRC Berechnung wird über Data[16-536] + PingCount[2] + RunIdle[1] + TimeStamp[13 low part] + { zero[2] + TimeStamp[14 high part] } durchgeführt. Teil in {} wird nicht übertragen. |

Für den Startwert der CRC Berechnung wird der selbe Algorithmus verwendet, wie in CIP-Safety Ed. 1.1.

Die Restfehlerrate berechnet sich mit:

$$\Lambda(N+k, d, p, \nu, c) \approx R(p, N+k, d) \cdot 3600 \cdot \nu \cdot c = 9{,}52526 \cdot 10^{-10} \qquad (4.48)$$

und bleibt damit unter 1% von SIL3.

Dabei wird eingesetzt

- $N+k$ = Anzahl Bits der Nachricht = 584
- $k$ = Anzahl der Prüfsummenbits = 32
- $p$ = Bitfehlerrate = $10^{-3}$
- $\nu$ = Anzahl Nachrichten pro Sekunde = 100
- $c$ = Anzahl der sicherheitsgerichteten Kommunikationsbeziehungen = 63
- $d$ = Hamming-Distanz = 8
- $$R(N+k, d, p) \approx \frac{1}{2^k} \cdot \sum_{n=d}^{N+k} \binom{N+k}{n} \cdot p^n \cdot (1-p)^{(N+k-n)} \qquad (4.49)$$

  für gerade n, da das Polynom den Faktor (x+1) enthält und somit alle ungeradzahligen Bitfehler sicher erkennt[177].

*Anmerkung: Bei einer Bitfehlerrate von $10^{-2}$ ist auch dieses Verfahren nicht für SIL3 Anwendungsbereiche geeignet ($\Lambda(10^{-2})=7{,}70782 \cdot 10^{-4}$). Wie später beim TSP1 Protokoll gezeigt wird, sind dazu Nachrichten mit maximal 48-Bit Data und einen für die Nachrichtenlänge von 96-Bit optimierten CRC erforderlich.*

[175] CRC32/8 aus [Cast83a] mit der ungespiegelten Normaldarstellung `0x1_F192_2815`

[176] [Cast83a]

[177] [Pete81]

## Langes Nachrichten Format

Der nachfolgend beschriebene Nachrichtenrahmen wird doppelt übertragen.

| Data[544-1960] | PingCount[2] | RunIdle[1] | TimeStamp[13] | CRC[32] |
|---|---|---|---|---|

*Abbildung 4.2: Änderungsvorschlag lange Nachricht*

Die Zahlen in [] bezeichnen die Bits der jeweiligen Information.

***Beschreibung der Nachrichten-Felder:***

Data[] 544 bis 1960 Bit Prozessdaten (68-245 Bytes)
Die Grenze 245 Bytes resultiert in der maximalen Länge von 510 für lange Nachrichten, das mit 2 Nachrichtenformaten gefüllt wird (=2•(245+6)=502) und eine Time Correction Nachricht (=8 Bytes für Multi-cast). Zwecks Harmonisierung wird für Multi-cast und Single-cast ein einziges Nachrichtenformat vorgeschlagen.

Die anderen Felder haben die gleiche Bedeutung wie beim kurzen Format. Das zweifache Übertragen resultiert aus der anzunehmenden Bitfehlerrate und der maximalen Nachrichten-Länge. Für Nachrichten bis zur Länge 2046 Bit hat das CRC Polynom 0xA814_498F eine Hamming-Distanz von mindestens 4 [176]. Damit ist es nicht der beste bekannte CRC für solche Längen, jedoch lag das Ziel des gewählten CRCs kurze Nachrichten nur einfach übertragen zu müssen und dabei ist der gewählte CRC besonders leistungsfähig.

Die Restfehlerrate berechnet sich mit:

$$\Lambda(N+k, d, p, v, c) \approx R(p, N+k, d) \bullet 3600 \bullet v \bullet c = 3{,}55925 \bullet 10^{-15} \qquad (4.50)$$

und bleibt damit unter 1% von SIL3.

Dabei wird eingesetzt

- $N+k$ = Anzahl Bits der Nachricht = 2008
- $k$ = Anzahl der Prüfsummenbits = 32
- $p$ = Bitfehlerrate = $10^{-3}$
- $v$ = Anzahl Nachrichten pro Sekunde = 100
- $c$ = Anzahl der sicherheitsgerichteten Kommunikationsbeziehungen = 63
- $d$ = Hamming-Distanz = 4
- $$R(N+k, d, p) \approx \frac{1}{2^k} \bullet \sum_{n=d}^{N+k} \binom{N+k}{n} \bullet (p^2)^n \bullet (1-p^2)^{(N+k-n)} \qquad (4.51)$$

  für gerade n, da das Polynom den Faktor (x+1) enthält und somit alle ungeradzahligen Bitfehler sicher erkennt.

*Anmerkung: Bei einer Bitfehlerrate von $10^{-2}$ ist auch dieses Verfahren nicht für SIL3 Anwendungsbereiche geeignet ($\Lambda(10^{-2})=2{,}92262 \bullet 10^{-7}$). Wie beim TSP2 Protokoll gezeigt wird, wären dazu 2 geeignete CRC Polynome zum Verfälschungsschutz erforderlich, wobei dann die doppelte Übertragung entfallen könnte.*

### Time-Coordination Nachrichten

| PingResponse[1] | PingCountReply[2] | zero[2] | **ConsumerTime[27]** | CRC[32] |
|---|---|---|---|---|

*Abbildung 4.3: Änderungsvorschlag Time-Coordination Nachricht*

***Beschreibung der Nachrichten-Felder:***

| | |
|---|---|
| PingResponse | 1 Bit, wie im Ack-Byte des CIP-Safety Ed. 1.1 |
| PingCountReply | 2 Bits, wie im Ack-Byte des CIP-Safety Ed. 1.1 |
| ConsumerTime | 27 Bit Wert der Empfängeruhr (consumer clock), entsprechend dem obigen Time-Stamp |
| zero | 2 Null Bits (reserviert) |
| CRC | Das gleiche Polynom wie oben (0xA814_498F), berechnet über PingResponse+PingCountReply+zero[2]+ConsumerTime. |

Die Restfehlerrate berechnet sich wie beim kurzen Nachrichten-Format.

### Time-Correction Nachrichten

| ActiveIdle[1] | Consumer#[4] | TimeCorrection[27] | CRC[32] |
|---|---|---|---|

*Abbildung 4.4: Änderungsvorschlag Time-Correction Nachricht*

***Beschreibung der Nachrichten-Felder:***

| | |
|---|---|
| ActiveIdle | 1 Bit, wie im MCast-Byte des CIP-Safety Ed. 1.1 |
| Consumer# | 4 Bits, wie im MCast-Byte des CIP-Safety Ed. 1.1 |
| TimeCorrection | 27 Bit Wert der Empfängeruhr (consumer clock), entsprechend dem obigen Time-Stamp |
| CRC | Das gleiche Polynom wie oben (0xA814_498F), berechnet über ActiveIdle+Consumer#+TimeCorrection. |

Die Restfehlerrate berechnet sich wie beim kurzen Nachrichten-Format.

### Time-Stamp

Das grundlegende Time-Stamp-Verfahren von CIP-Safety Ed. 1.1 wurde nicht geändert, nur an einigen wenigen Stellen angepasst. So ist die Uhrengranularität unverändert 128 Mikrosekunden, da aber 27 Bits für den Zähler benutzt werden, hat der Time-Stamp nun einen Wertebereich von 4,77 Stunden.

Der Time-Stamp wird verkürzt übertragen, damit die Nachrichtenlänge, insbesondere bei DeviceNet[178], 8 Bytes nicht überschreitet. Dies gilt sowohl für die 2-Byte Prozessdaten-Nachrichten, wie auch für Time-Correction-Nachrichten.

Mit den zuvor (Protokoll Ed. 1.1) betrachteten Annahmen für das Einfügen von Nachrichten, beherrschen die beschriebenen Mechanismen dies mit 1% von SIL3.

Dies berechnet sich mit: $W = EPI / (2^{27} \bullet 128\ Mikrosekunden)$ (z.B. $W(10\ ms) = 5{,}8 \bullet 10^{-7}$), wobei $2^{27}$ der maximale Time-Stamp-Wert ist. Werden weiterhin die Anzahl der Netzwerkkomponenten NK betrachtet, so ergibt sich eine Restfehlerrate I für unerkannte Einfügungen von:

a) $EPI = 10\ ms,\ 17179\ NK \bullet I = 5{,}8 \bullet 10^{-7} \bullet 17179 \bullet 1000\ FIT = 9{,}96 \bullet 10^{-10}\ h^{-1}$

b) $EPI = 100\ ms,\ 1717\ NK \bullet I = 5{,}8 \bullet 10^{-6} \bullet 1717 \bullet 1000\ FIT = 9{,}96 \bullet 10^{-10}\ h^{-1}$

[178] DeviceNet überträgt auf der Netzwerkebene Pakete von einer Länge von maximal 8 Bytes. Längere Daten müssen fragmentiert werden, wodurch die Effizienz sinkt.

c) $EPI = 1.000\ ms,\ 171\ NK \bullet I = 5{,}8 \bullet 10^{-5} \bullet 171 \bullet 1000\ FIT = 9{,}92 \bullet 10^{-10}\ h^{-1}$ *(4.52)*

Es darf angenommen werden, dass die Anzahl der zulässigen Netzwerkkomponenten für die allermeisten Applikationen ausreichend ist und keine reale Restriktion darstellt.

### Nachrichten Fragmentierung

Nachrichten-Fragmentierung wird durch den verwendeten CRC32 erkannt (mit dessen Qualität). Insbesondere die Berechnung des CRCs über Prozessdaten und Time-Stamp führt dazu, dass die beiden Nachrichtenteile nicht mehr unerkannt fragmentiert werden können.

### Time-Stamp Nachrichten Verfälschung

Die Qualität für das Aufdecken einer Nachrichten-Verfälschung des Time-Stamp-Anteils geschieht mit der gleichen Qualität, wie für die Prozessdaten und ist bei einer Bitfehlerrate von $10^{-3}$ ohne weitere Annahmen besser als 1% von SIL3.

### Weiterentwicklung von CIP-Safety Edition 2.2

In der später spezifizierten Version des CIP-Safety Kommunikationsprotokolls gemäß CIP-Vol-5 Edition 2.2[179] wurde zwar nicht der Vorschlag des Autors übernommen, jedoch wurde die Spezifikation abgeändert, um die in obiger Analyse zu Tage getretenen Schwächen zu beheben.

Die historischen Nachrichtenformate für kurze und lange Nachrichten wurden für spezielle Einsatzgebiete unverändert belassen. Für den allgemeinen Einsatz wurden erweiterte Nachrichtenformate definiert[180].

Die vorgelegten Änderungen in CIP-Safety Ed-2.2 betreffen im Wesentlichen zwei Kernpunkte.

1. Einsatz eines CRC24 (`0x5d_6DCB`) über sichere Daten und Time-Stamp gegen Datenverfälschung und Fragmentierung, sowie zusätzlich einen CRC16 (`0x080F`) bei den langen Nachrichtenformaten. Untersucht wurde der gewählte CRC24 von Castagnoli[181]. Er wird dort CRC24-6.1 genannt. Dieser CRC kommt auch beim Flexray-Protokoll, sowie bei PROFIsafe V2 zum Einsatz[182].
2. Verwendung eines 32-Bit Time-Stamps, von dem 16 Bits übertragen werden und die restlichen 16 Bit (Rollover count) mit in die CRC Berechnung eingehen.

Für kurze und lange Nachrichtenlängen wurde je ein „Extended“ Nachrichtenformat definiert.

*Tabelle 4.8: CIP-Safety „Extended“ kurzes Nachrichtenformat[183]*

| ***Actual Data*** | ***Mode Byte*** | ***CRC-S5*** | ***Time-Stamp*** | ***CRC-S5*** |
|---|---|---|---|---|
| sichere Prozessdaten 1 oder 2 Bytes | 1 Byte Statusinformationen | CRC24 (2 Bytes des CRCs) | 2 Bytes | CRC24 (1 Byte des CRCs) |

[179] [CIP5-2.2]

[180] [CIP5-2.2]

[181] [Cast93]

[182] [Flex05], [PROFIsafeV2]

[183] [CIP5-2.2]

*Tabelle 4.9: CIP-Safety „Extended“ langes Nachrichtenformat*[183]

| ***Actual Data*** | ***Mode Byte*** | ***CRC-S3*** | ***Actual Data*** | ***CRC-S5*** | ***Time-Stamp*** | ***CRC-S5*** |
|---|---|---|---|---|---|---|
| sichere Prozessdaten 3 bis 250 Bytes | 1 Byte | CRC16 (2 Bytes) | inverse sichere Prozessdaten | CRC24 über inverse Prozessdaten und Time-Stamp (2 Bytes des CRCs) | 2 Bytes | CRC24 (1 Byte des CRCs) |

Der CRC24/6.1 hat für das kurze Nachrichtenformat $N+k$=64 Bits eine Hamming-Distanz $d$=8 und für das lange Nachrichtenformat $N+k$=2132 Bits eine Hamming-Distanz $d$=4. [181]

Die Restfehlerwahrscheinlichkeit für kurze Nachrichten ergibt sich damit zu

$$\Lambda(p=10^{-2}) = 1{,}40598 \cdot 10^{-14} \cdot 3600 \cdot 100 \cdot 62 = 3{,}3658 \cdot 10^{-7} \qquad (4.53)$$

Den heute üblichen Anspruch aus IEC 61784-3 und IEC 61508-2, die Nachrichtenverfälschung bei einer Bitfehlerrate von $10^{-2}$ mit einem Black-Channel Modell zu beherrschen, wird mit CIP-Safety Edition 2.2 für kurze Nachrichten nicht erreicht. Die gewählten Sicherungstechniken haben dazu eine zu geringe Qualität.

Die Restfehlerwahrscheinlichkeit für lange Nachrichten entzieht sich den üblichen Rechenmethoden. Die Nachrichteninhalte sind nur zum Teil doppelt übertragen und zum Teil nur einfach. Daher ist der Ansatz, $p^2$ in der approximativen Berechnung zu verwenden, nicht ganz statthaft. Weiterhin bliebt zu untersuchen, wie sich die Unabhängigkeit der beiden CRCs S3 und S5 darstellt.

Unterstellt man die Unabhängigkeit und setzt durchgängig eine Bitfehlerrate von $p^2=10^{-4}$ ein, ergibt sich mit dem Produkt der Einzelwahrscheinlichkeiten von CRC-S3 und CRC-S5 zu

$$R_{CRC\text{-}S3}(p^2=10^{-4}) \approx 8{,}6 \cdot 10^{-10}$$

$$R_{CRC\text{-}S5}(p^2=10^{-4}) \approx 4{,}1 \cdot 10^{-12}$$

$$\Lambda(p^2=10^{-4}) \approx 3{,}526 \cdot 10^{-21} \cdot 3600 \cdot 100 \cdot 62 = 7{,}870 \cdot 10^{-14}. \qquad (4.54)$$

Im Rahmen der offenen Punkte, auf denen die Berechnung beruht, wird von CIP-Safety Ed. 2.2 für lange Nachrichten das Ziel SIL3 erreicht.

Betrachtet man jedoch die einfach übertragenen Daten Time-Stamp und CRC-S5, so erhält man alleine für diesen Teil ein $\Lambda(p=10^{-2}) \approx 7{,}42501 \cdot 10^{-9}$, was etwas oberhalb des für SIL3 zulässigen Levels liegt. Das heißt, die obige, sehr optimistische Betrachtung kann so nicht angestellt werden und eine Simulation der Tauglichkeit des Sicherungsmechanismusses ist hier angeraten.

Die Entwicklung und Verifikation von CIP-Safety wurde für die abgeänderte Version Edition 2.2[184] durch eine Konzept-Prüfung durch den TÜV-Rheinland, Köln, im Jahr 2008 verifiziert.

## 4.4 FF-SIF-Protokoll

Teilnehmer der FF-SIF Kommunikationssysteme sind Geräte in der Rolle von Slaves, die mit einer zentralen sicherheitsgerichteten Speicher-Programmierbaren-Steuerung, kurz Sicherheitssteuerung, kommunizieren[185]. In der Spezifikation FF-SIF[186] werden die Slaves *devices* und die Sicherheitssteuerung *logic-solver* genannt.

[184] [CIP5-2.2]

[185] [FFSA06]

[186] [FF-SIF]

Zum Schutz vor Nachrichtenverfälschung werden übliche Mechanismen, wie CRC32 und doppelte Übertragung innerhalb einer Nachricht genutzt. Zur zeitlichen Kontrolle werden die Mechanismen Watch-Dog-Timer, *stale counter* und eine Zeitsynchronisation zusammen mit Monotonie-Zählern genutzt[186]. Diese Mechanismen sind sehr eng mit dem unterlagerten FF Protokoll verknüpft und verhindern damit eine mögliche, vom Transportprotokoll unabhängige Nutzung von FF-SIF.

Zur Sicherung der Adressierung wird ein *connection key* genutzt, der mit in die CRC Berechnung eingeht, aber nicht in der Nachricht übertragen wird[187].

Der Nachrichtenrahmen für FF-SIF setzt sich wie folgt zusammen[186]:

*Tabelle 4.10: FF-SIF Nachrichtenformat*

| ***Original Data*** | ***MCN***[188] | ***CRC32*** | ***Original Data-2*** | ***MCN-2*** | ***CRC32-2*** |
|---|---|---|---|---|---|
| Prozessdaten 2 – 120 Bytes | Monotoniezähler | IEEE 802.3 CRC | Kopie von *Original Data* | Kopie von *MCN* | Kopie von *CRC32* |

Der CRC32 wird über die nicht übertragenen Informationen *connection key*, *object index*, sowie über *MCN* und die Prozessdaten gerechnet. Der zweite Teil der Nachricht stellt eine Kopie des ersten Teils dar.

Nachfolgend wird der bis Mitte 2010 veröffentlichte Stand der FF-SIF Protokollspezifikation betrachtet.

Setzt man als Modell für die Übertragungseinrichtung den binär-symmetrischen Übertragungskanal voraus, so ergibt sich bei gegebener Hamming-Distanz d die Restfehlerwahrscheinlichkeit pro Stunde unerkannt empfangener verfälschter Nachrichten mit Gleichung (2.4) und dabei ist für propere Polynome *R()* gemäß Gleichung (2.6) anzuwenden.

Die Bitfehlerrate $p=(10^{-2})^2$ wird wegen der doppelten Übertragung, verbunden mit dem Vergleich beim Empfang, quadriert eingesetzt.

Die maximale Nachrichtenlänge $N$=1008 berechnet sich aus den maximal übertragbaren Nutzdaten (120 Bytes) plus dem Header bestehend aus Object-Index (*OI*) (4 Bytes) und Macro-Cycle-Number (*MCN*) (2 Bytes)[189]. Nicht mit eingerechnet wird der nicht übertragene Anteil der Nachricht, der Connection-Key. Unter der Annahme, dass der Object-Index im High-Byte immer 0 überträgt, könnte die maximale Nachrichtenlänge um 16 Bits reduziert werden, was hier im Hinblick auf eine zukünftige Verwendung nicht erfolgt.

Der Wert von $c$=64+20 repräsentiert dabei die Menge an Verbindungen, die maximal für eine Sicherheitsfunktion erforderlich sind. Die Sicherheitsfunktion setzt sich aus einer Kommunikationsstrecke für die Übertragung des Eingangswerts zur Logikverarbeitung und von dort zur Ausgabeeinheit zusammen. In der Verbindungsmenge von 64 nicht eingerechnet sind die, für die Ausgabe-Devices notwendigen, Publishing Verbindungen ihrer Input-Daten, die mit 20 weiteren Verbindungen angesetzt werden.

Der Wert von $v$=10 repräsentiert die maximal vom Übertragungssystem unterstellte Übertragungsleistung je Verbindung. Dabei wird angenommen, dass der Makrozyklus auf einem H1-Link mit 5-7 Teilnehmern, die Sicherheitssteuerung nicht gerechnet, nicht kleiner als 100 ms ist. Durch die 64 + 20 Verbindungen ergeben sich dann 840 Nachrichten je Sekunde.

---

[187] FF-SIF nutzt hierbei ein Verfahren, das auch bei der PROFIsafe Entwicklung verwendet wurde. Dabei wird die Adresse für eine sicherheitsgerichtete Nachricht in die CRC-Rechnung einbezogen, aber nicht übertragen.

[188] *MCN* steht für Macro Cycle Number

[189] [FF-SIF]

Als CRC wird das IEEE-802.3 Polynom 0xedb88320 mit einer Hamming-Distanz von $d$=5 bis zu einer Länge von $N+k$=3006 Bits (incl. CRC32 Bits) eingesetzt[190]. Bleibt die Nachrichtenlänge unter $N+k$=200 Bits, so kann eine Hamming-Distanz von 7 eingesetzt werden. Bleibt sie unter $N+k=112$ Bits, so kann eine Hamming-Distanz von 8 eingesetzt werden.

Die 112 Bits sind beispielsweise für die relevanten Nachrichten eines FF-SIF-AI- und FF-SIF-DO-Blocks anwendbar.

Die maximale Restfehlerwahrscheinlichkeit pro Stunde ergibt sich dann für $d$=5 mit

$$R(p=(10^{-2})^2, N=1008, k=32)=2{,}14502 \cdot 10^{-17} \qquad (4.55)$$

zu

$$\Lambda \le R() \cdot 3600 \cdot 10 \cdot 84\,h^{-1}=6{,}48653 \cdot 10^{-11}\,h^{-1}<1 \cdot 10^{-9}\,h^{-1}=\ 1\%\ \ von\ \ SIL3 \qquad (4.56)$$

Aus diesem Wert resultiert die Sicherheitsmarge von 15,41, berechnet aus $10^{-9}$ / 6,48653 • $10^{-11}$. Dies ist auch dann noch ausreichend, wenn die sehr konservativ angenommenen Werte für die Anzahl der Kommunikationsverbindungen für eine Sicherheitsfunktion in einer konkreten Applikation erhöht und der minimale Makrozyklus reduziert wird.

Die Restfehlerwahrscheinlichkeit pro Stunde ergibt sich bei einer Bitfehlerrate von $10^{-2}$ und $d$=7 mit maximaler Nachrichtenlänge von 200 Bits ($N+k$) zu $\Lambda=1{,}58112 \cdot 10^{-19}$.
Die Restfehlerwahrscheinlichkeit pro Stunde ergibt sich bei einer Bitfehlerrate von $10^{-2}$ und $d$=8 mit maximaler Nachrichtenlänge von 112 Bits ($N+k$; incl. CRC) zu $\Lambda = 3{,}31703 \cdot 10^{-24}$.
Da derzeit im Allgemeinen kurze Nachrichten, mit einer Länge unter 112 Bits, bei der Anwendungen von FF-SIF in der Prozessindustrie zur Anwendung kommen sollen, besitzt FF-SIF sehr hohe Sicherheitsreserven bezüglich der Nachrichtenverfälschung. Zu erwarten ist allerdings, dass in Zukunft die Aggregation mehrerer Prozessgrößen in einer Nachricht zum Einsatz kommen wird, da sich damit der Overhead reduzieren und somit die Effizienz des Übertragungssystems steigern lässt.
Zur Erkennung von Nachrichtenwiederholungen verwendet FF-SIF bei Publisher/Subscriber Kommunikationsbeziehungen das Nachrichtenfeld *MCN*.

Bei der *MCN* handelt es sich um einen streng monoton steigenden, umlaufenden 16-Bit Zähler, der aus der Data-Link-Time durch ganzzahlige Division der Dauer des Macro-Zyklusses gebildet wird[191].

Beim FF-SIF Empfänger einer Nachricht, besteht eine Erwartungshaltung bezüglich der *MCN*, die sich aus seiner eigenen Data-Link-Time berechnet. Trägt die empfangene Nachricht nicht die erwartete *MCN*, so wird sie verworfen.

Damit hat FF-SIF die Fähigkeit Nachrichten aus dem vorherigen Makrozyklus zu erkennen und zu verwerfen.
Nicht erkannt werden Wiederholungen von Nachrichten innerhalb des selben Makrozyklusses. Reduziert wird die Zeitspanne für die Akzeptanz von Nachrichten dadurch, dass eine Nachricht nur dann verwendet wird, wenn sie vor der Ausführung des SIF-Function-Blocks empfangen wurde. Da die Function-Block Ausführung auch am Ende des Makrozyklusses stattfinden kann, z.B. bei FF-SIF-DO Blöcken, kann der Zeitbereich den ganzen Makrozyklus umfassen.

Dies kann jedoch als unproblematisch angesehen werden, da es sich um ein und die selbe Nachricht handelt und dies für die Verarbeitung der publizierten Daten keinen Unterschied macht.

Unterstellt man nun, dass eine Verbindungsaufbau-Nachrichtensequenz (FMSInitiate.req/.cnf + pub/sub) in den verwendeten Kommunikationseinrichtungen gespeichert wurde und nach dem Schließen der Verbindung wieder eingespielt wird, so kann der Empfänger dies an der *MCN* der Publisher / Subscriber Nachrichten erkennen.

---

[190] siehe [Fuji89], [Cast93], [Koop02a]

[191] [FF-SIF]

Die Wahrscheinlichkeit, dass die *MCN* der Sequenz zufällig zur aktuell gültigen *MCN* (berechnet aus der Data-Link-Time) passt und daher der Mechanismus versagt, liegt bei:

$$\frac{1}{2^{16}} \approx 1{,}5258 \bullet 10^{-5} \qquad (4.57)$$

Zum Beherrschen des Szenarios ist ein Sperren der sicheren Funktion des Subscribers, nach dem die Funktion in den sicheren Zustand übergegangen ist, notwendig. Diese Sperre darf vom Operator erst nach Behebung des Problems zurückgesetzt werden.

Für FF-SIF-DO Function-Blöcke ist hier der Modus LO (Local Override) zum Sperren, der Clear_Fault_State Eingang des Ressource-Blocks und der RESET_D Eingang des FF-SIF-DO Blocks zum Aufheben der Sperre vorgesehen[191].

Für den FF-SIF-Host-Function-Block (HFB) der Sicherheitssteuerung ist dies durch Hersteller spezifische Mechanismen umzusetzen.

Da die Implementierung von FF-SIF im Allgemeinen aus einem nicht sicherheitsgerichteten FF-H1 Stack und einem davon getrennt implementierten FF-SIF Stack bestehen, ist die Behandlung der FMSInitiate.req/.cnf auf den FF-H1 Stack verlagert. Es handelt sich dabei um Standard FF-H1 Funktionalitäten.

Die Trennung von FF-SIF und Standard FF-H1 bewirkt jedoch, dass auch bei nicht bestehender Verbindung FF-SIF Nachrichten akzeptiert werden, wenn sie eine gültige *MCN* aufweisen. Die Wahrscheinlichkeit berechnet sich wie oben zu $1{,}5258 \bullet 10^{-5}$. Wurde die FF-SIF Verbindung geschlossen und wurde damit die Sperre aktiviert, so stellt dies kein Problem dar.

Die Gefährdung entsteht jedoch vor bzw. beim Starten der Verbindung, da in dieser Situation keine Sperre aktiv ist. Dieser Fall entspricht den unten behandelten Einfügungen von Nachrichten bei bestehender Verbindung oder „eben gerade“ geöffneter Verbindung.

Der Verlust einer Nachricht wird durch die Erwartungshaltung, dass je Makrozyklus eine Nachricht mit passender *MCN* erwartet wird, erkannt.

Betrachtet wird hier für FF-SIF Publisher/Subscriber-Verbindungen das Einfügen von Nachrichten während einer bestehenden Verbindung. Der Fall für den Zustand, dass keine Verbindung besteht, wurde schon oben bei der Diskussion über die Wiederholung von Nachrichtensequenzen beim Starten bzw. Wiederanlaufen, betrachtet.

Bei bestehender Verbindung wird die eingefügte Nachricht anhand der unpassenden *MCN* erkannt. Die Wahrscheinlichkeit, dass der Mechanismus für Nachrichten aus anderen Makrozyklen versagt, liegt bei:

$$\frac{1}{2^{16}} \approx 1{,}5258 \bullet 10^{-5} \ , \qquad (4.58)$$

wenn es sich um eine Nachricht eines beliebigen anderen Makrozyklus der selben Verbindung handelt.

Setzt man die erwartete Fehlerrate für das Einfügen einer Nachricht mit $10^{-6}\ h^{-1}$ an, so ergibt sich eine Restfehlerwahrscheinlichkeit pro Stunde von:

$$\frac{1}{2^{16}} \bullet 10^{-6} h^{-1} = 1{,}5258 \bullet 10^{-11} h^{-1} \leq 1 \bullet 10^{-9} h^{-1} = 1\% \ \ von \ \ SIL3 \qquad (4.59)$$

SIL3 bleibt bei dieser Maßnahme bis zu einer Fehlerrate in der Größenordnung von $10^{-5}\ h^{-1}$ erhalten. Die Annahme der Fehlerrate ist jedoch bislang für FF-SIF Umgebungen nicht belegt und daher mit Vorsicht zu betrachten.

Für Implementierung von „Black-Channel"-Komponenten[192] ist es daher angeraten besondere Maßnahmen zur Vermeidung von Wiederholungen/Einfügungen zu ergreifen. Insbesondere dürften keine Mechanismen im „Black-Channel" vorgesehen werden, die regulär Nachrichten wiederholen.

Dies ist jedoch vor dem Hintergrund von Verfügbarkeitsmechanismen nicht allgemein haltbar. Daher sollte bei den Mechanismen darauf geachtet werden, dass Wiederholungen die strenge Monotonie der Nachrichtensequenz[193] nicht verletzen. Der Aspekt ist auch beim Kanal zwischen dem FF-Host-Stack und der FF-SIF Implementation der Komponente zu beachten, die ebenfalls zum „Black-Channel" gehört.

Die Anforderungen an den „Black-Channel" sollten in der FF-SIF Spezifikation aufgenommen werden, sofern nicht der nachfolgende Ansatz zur Problemlösung herangezogen wird. Dieser wurde der Foundation-Fieldbus Organisation im Rahmen der Forschungsarbeiten vorgestellt, fand jedoch keine Berücksichtigung bei der Protokolldefinition.

Ungeachtet der Bemühungen der Implementierung des „Black-Channels" verbleibt das undefinierte Restrisiko, dass eine „Black-Channel"-Komponente fehlerhafter-weise trotzdem eine „alte" Nachricht in den Nachrichtenstrom einschleust.

***Lösungsansatz:***

Ein Ansatz dem Problem entgegenzutreten, besteht darin, die gesamte Data-Link-Time mit in die CRC-Berechnung mit einfließen zu lassen. Übertragen werden muss die DLT dazu nicht, sie wäre Bestandteil der virtuellen Nachricht. Die erzielbare Qualität wird dann durch die CRC Größe plus der *MCN* definiert. Aus diesen 48 Bits ergibt sich die Versagenswahrscheinlichkeit mit

$$\frac{1}{2^{(16+32)}} \approx 3{,}5527 \bullet 10^{-15} \quad , \qquad (4.60)$$

wenn es sich um eine Nachricht eines beliebigen anderen Makrozyklus handelt.

Setzt man die erwartete Fehlerrate der Einfügung mit $10^{-6}\ h^{-1}$ an, so ergibt sich eine Restfehlerwahrscheinlichkeit pro Stunde von:

$$\frac{1}{2^{(16+32)}} \bullet 10^{-6} h^{-1} = 3{,}5527 \bullet 10^{-21} h^{-1} \leq 1 \bullet 10^{-9} h^{-1} = 1\% \ \ von \ \ SIL3 \quad . \qquad (4.61)$$

SIL3 bleibt bei dieser Maßnahme bis zu einer Fehlerrate in der Größenordnung von $281{,}4749 \bullet 10^3$ eingefügten Nachrichten $h^{-1}$ erhalten, was immerhin 40,35% der verfügbaren Bandbreite eines H1-Links darstellt.

Eine falsche Nachrichtenreihenfolge wird durch die Erwartungshaltung der *MCN* erkannt (Restfehlerwahrscheinlichkeit wie bei Einfügen, s.o.).

Die Fragmentierung von Nachrichten, d.h. das Zusammensetzen einer Nachricht aus Teilen, z.B. aus anderen Nachrichten, bzw. die Reihenfolgevertauschung von Teilen einer Nachricht, wird von FF-SIF durch den CRC über die gesamte Nachricht erkannt.

FF-SIF benutzt drei Mechanismen zur Erkennung von Nachrichtenverzögerung und Altern von Nachrichten. Die ersten beiden Mechanismen beziehen sich hierbei auf den Subscriber.

Der erste Mechanismus besteht darin, dass die *MCN* der empfangenen Nachricht zur lokal berechneten *MCN* passen muss und in jedem Makrozyklus eine Nachricht erwartet wird. Somit werden, von Einfügungen abgesehen, nur Nachrichten aus dem aktuellen Makrozyklus akzeptiert. Das Alter der Daten ist dann beim empfangenden Subscriber, ungeachtet der Verzögerung der Datenaufbereitung außerhalb von FF-SIF, kleiner als die aktuelle Dauer des laufenden Makrozyklusses.

---

[192] z.B. FF-H1 Stack in Sicherheitssteuerung und Device

[193] Das Hauptziel liegt darin, dass sich Nachrichten niemals überholen!

Der Stale-Counter definiert die Anzahl aufeinander folgender Makrozyklen, in denen keine gültige Nachricht empfangen wurde. Überschreitet der Stale-Counter ein konfiguriertes Maximum, so wird die Verbindung geschlossen und die sichere Reaktion eingeleitet. Ein Stale-Counter ist zu jeder FF-SIF-Verbindung auf Subscriber-Seite definiert[194].

Findet ein Function-Block zu seinem Ausführungszeitpunkt keine Input-Daten aus dem aktuellen Makrozyklus vor, so sind die Input-Daten als ungültig zu betrachten[195]. Im einfachen Fall bedeutet dies auch, dass der Function-Block keine Output-Daten erzeugt und in Folge keine Output-Daten publiziert werden.

Wird wieder eine aktuelle FF-SIF-Nachricht empfangen, so wird der Stale-Counter zurück auf 0 gesetzt.

Das Konzept des Stale-Counters führt dazu, dass bei den nachgeschalteten Function-Blocks der Sicherheitsfunktion ebenfalls ihr Stale-Counter und zwar noch innerhalb des selben Makrozyklusses, erhöht wird. Dies stellt ein, in anderen gängigen Sicherheitsprotokollen nicht verwendetes, aber sehr reaktionsschnelles Verfahren dar.

Erfolgt bei einem Ausgabe-Function-Block das Überschreiten des parametrierten Limits des Stale-Counters, so nimmt er seinen parametrierten Fehlerzustand ein (Sicherheitsreaktion) [194].

*Anmerkung: Nur der letzte Function-Block einer Sicherheitsfunktion nimmt eine Störaustastung mittels Stale-Counter vor. In den vor ihm liegenden Function-Blocks wird durch das „Nicht"-Publish-en eine „sofortige" Fehlerreaktion ausgelöst. Lediglich das Unterbrechen der Verbindung (sperren) ist in diesen Function-Blocks vom Stale-Counter-Limit beeinflusst. Ist der HFB der Sicherheitssteuerung der letzte Output-Function-Block, weil die Daten z.B. außerhalb von FF-SIF weiterverarbeitet werden (lokale IO), so muss der HFB bzw. die Applikation für dieses Datum die Störaustastung gemäß Stale-Counter realisieren.*

Ist das Stale-Counter-Limit auf den Wert SCL parametriert, so erfolgt die Reaktion für einen H1-Link auf einen Eingangswertewechsel bei Vorhandensein einer Kommunikationsunterbrechung durch den letzten Function-Block der Sicherheitsfunktion innerhalb von

$$t_{wc\text{-}ff} = (SCL+2) \bullet MCT. \qquad (4.62)$$

Hierbei wird angenommen, dass der erste Function-Block der Sicherheitsfunktion kurz nach seinem Publizieren die Fähigkeit verliert Nachrichten zu senden und einen Signalwechsel übertragen soll. Auf diesen Signalwechsel (AI1 → AI2) wird sicherheitsgerichtet nach $t_{wc\text{-}ff}$ reagiert.

***Beispiel:***

| | | Makrozyklus x | | | | | | Makrozyklus x+1 | | | | | | Makrozyklus x+2 | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| H1-Link | | | pub AI1 | ... | | pub DO1 | ... | | Fehler | ... | | kein pub | | | Fehler | ... | | kein pub | | |
| | | SIS-AI-FB | | | HFB | | SIS-DO-FB | SIS-AI-FB | | | HFB | | SIS-DO-FB **stale=1** | SIS-AI-FB | | | HFB | | SIS-DO-FB **stale=2** | |
| Prozesswert | AI1 | AI2 | | | | | | DO1 | | | | | | | | | | | | Fail-Safe DO |

*Abbildung 4.5: FF-SIF Fehlerreaktion mit Stale-Counter*

[194] [FF-SIF]

[195] Ausnahme bilden die wiederholt publizierten FF-SIF-DO-Block Daten

Das Beispiel zeigt ein Szenario mit Stale-Count-Limit=1, d.h. ab einem Stale-Count von 2 wird die Fail-Safe Reaktion eingeleitet[196].
Die normale Reaktionszeit in einem H1-Link von *2 • MCT* wird hier also um das *Stale-Count-Limit • MCT* verlängert.

Nicht berücksichtigt bei $t_{wc\text{-}ff}$ sind die Zeiten, die durch die Prozesswertmessung am Eingang und nachgeschaltete Filter zu Stande kommen.

Der Vorteil des Stale-Counter-Konzeptes liegt darin, dass im Vergleich zu einer Störaustastung zwischen den einzelnen Function-Blocks, die maximale Reaktionszeit minimiert wird, aber gleichzeitig ein großes Maß an Robustheit (Verfügbarkeitsreserve) vorhanden ist. Die durch das Stale-Count-Limit definierten zulässigen Störungen können sich an beliebiger Stelle der Sicherheitsfunktion auswirken.

Damit das Konzept wirksam ist, müssen die Function-Blocks, nicht vorhandene Input-Daten dazu veranlassen, selber keine davon abhängenden Output-Daten zu publizieren. Insbesondere in der Sicherheitssteuerung muss darauf geachtet werden, dass erst nach Stale-Count-Limit Zyklen mit fehlenden Daten *im nächsten Sicherheitssteuerung Zyklus* eine Reaktion stattfindet, insbesondere, wenn die Daten außerhalb von FF-SIF weiter verarbeitet werden. Die Sicherheitssteuerung sollte das Fehlen von Input-Daten ebenso durch nicht publizieren der abhängigen Output-Daten behandeln und nicht etwa durch Einstellen und Publizieren der sicheren Werte.

Der zweite Mechanismus zur Beherrschung von Nachrichtenverzögerungen und unzulässigem Nachrichtenalter besteht darin, dass der mit dem FF-SIF-DO-Block verbundene Output-Transducer-Block innerhalb der STALE_DATA_Time eine Ausgabe seines DO-Blocks erwartet. Hierzu muss vom DO Block eine gültige Nachricht empfangen und der DO Block ausgeführt werden. Trifft dies innerhalb von STALE_DATA_Time nicht zu, so geht der DO-Block in den Zustand LO Mode (Local Override)[196].

Mit den beiden Mechanismen Stale-Count-Limit und STALE_DATA_Time, wird die Übertragungsdauer, sowohl der Offset zum Zeitmaster für die *MCN*, wie auch Queuing und Drift der Übertragungsdauer nicht beherrscht.
Um der Problematik Queuing, Drift und Zeitsynchronisationsfehler zu begegnen, wurden im Rahmen dieser Arbeit der Foundation Fieldbus Organisation zwei Vorschläge gemacht, die nachfolgend erläutert werden.

### *MCN* Generator in Sicherheitssteuerung

Der erste Teil des Konzeptes besteht darin, dass die Sicherheitssteuerung die sicherheitstechnische Referenz für die *MCN* darstellt. Für von der Sicherheitssteuerung empfangene Nachrichten bedeutet dies, dass sie aus dem Makrozyklus der Sicherheitssteuerung stammen müssen, um gültig zu sein. Damit werden unzulässig verzögerte Nachrichten von FF-SIF-Geräten Richtung Sicherheitssteuerung erkannt. Dabei ist es unerheblich, ob eine Verzögerung der Nachricht erfolgt ist, oder ob der Mechanismus der Zeitsynchronisation versagt hat. Der bisherige Mechanismus der Zeitsynchronisation war bislang Bestandteil des Black-Channels.

In der FF-SIF Spezifikation müsste die Sicherheitssteuerung als Referenz der Zeitsynchronisation gefordert werden, damit obiges Problem gelöst wird. Wahrscheinlich müsste Foundation-Fieldbus Organisationen dazu eine neue oder geänderte CRC-gesicherte Zeitsynchronisationsnachricht definieren.

Der Verbesserungsvorschlag für das FF-SIF Protokolls wurde in dieser Hinsicht von der Foundation Fieldbus Organisation nicht angenommen.

---

[196] [FF-SIF]

## Überwachung des FF-SIF-DO-Block

Zweitens muss eine FF-SIF Ausgabeeinheit, z.B. FF-SIF-DO-Block, sein empfangenes Datum republizieren. Dabei ist die *MCN* des jeweiligen Macro-Zyklusses zu verwenden. Diese *MCN* stimmt mit der *MCN* des empfangenen Datums überein. Die Sicherheitssteuerung, die sich auf das von der Ausgabeeinheit publizierte Datum subscribt, überprüft durch den ordnungsgemäßen Empfang dieser Nachricht, dass die FF-SIF Ausgabeeinheit mit aktuellen Daten arbeitet. Stellt die Sicherheitssteuerung fest, dass dies nicht mehr gewährleistet ist, so publiziert sie sichere Ausgangsdaten zu dieser Ausgabeeinheit.

Das Publizieren sicherer Daten führt zu einer schnelleren Sicherheitsreaktion, als es das Einstellen der Verbindung bewirken würde, da hier noch der Stale-Count-Limit abgewartet wird.

Eine Störaustastung via Stale-Counter wird für das republizierte Datum nicht wie für jedes andere Input-Datum durchgeführt.

Es ist jedoch erforderlich, dass auch der republizierte Weg eine Störaustastung hat. Entweder wurde das republizierte Datum nicht von der Sicherheitssteuerung empfangen oder das von der Sicherheitssteuerung zu publizierende Datum wurde nicht publiziert oder wurde vom FF-SIF-DO-Block nicht empfangen. Aus letzterem folgt automatisch, dass auch der FF-SIF-DO-Block kein Datum republiziert, da gemäß den Ausführungsregeln für FF-SIF ein Function-Block nur ausgeführt wird und Daten publiziert, wenn er alle nötigen Eingangsdaten empfangen hat[196].

Aus diesem Grund muss die Sicherheitssteuerung die republizierten Daten solange das Stale-Count-Limit nicht erreicht ist, als gültig ansehen.

Ungültig wird das Datum jedoch, unabhängig vom Stale-Counter, sobald von der Sicherheitssteuerung ein Verzug erkannt wird. Wird in Makrozyklus x ein republiziertes Paket aus einem Makrozyklus ungleich *x-1* festgestellt, so wird unmittelbar die Sicherheitsreaktion eingeleitet. Der Vorteil der sofortigen Reaktion wird mit dem Nachteil erkauft, dass Einfügungen und Wiederholungen von republizierten Nachrichten zur Sicherheitsreaktion führen.

Diese Verbesserung des FF-SIF Protokolls wurde von der Foundation Fieldbus Organisation angenommen.

## Fehlerszenarien für Queuing und Verzug

***Szenario 1:***

Es entsteht ein Queuing von Host-Function-Block der Sicherheitssteuerung in Richting SIF-DO Block, aber nicht in umgekehrter Richtung, was z.B. durch Implementierungen der Standard-FF-H1-Stack-Software und der Software innerhalb der Geräte und der Sicherheitssteuerung denkbar ist.

1. Als erstes überschreitet das Queuing von Host-Function-Block Richtung SIF-DO die Zeit so sehr, dass das republizierte Datum zu spät bei der Sicherheitssteuerung ankommt.
2. Die Sicherheitssteuerung erkennt dann den Verzug und sendet die Failsafe-Daten.

Unter der Voraussetzung, dass der Host-Function-Block im FF-H1-Schedule vor dem SIF-DO Publishing geschieht, wird das Queuing nach 2 Makrozyklen erkannt. Im ersten Makrozyklus empfängt die Sicherheitssteuerung keine republizierte Nachricht und im darauf folgenden die verspätete.

***Szenario 2:***

Ab dem Makrozyklus x gehen die vom SIF-DO publizieren Daten verloren.

In Zyklus *x+1* stellt die Sicherheitssteuerung fest, dass ihr Input-Daten fehlen und zählt daher ihren Stale-Counter hoch. Dies wiederholt sich insgesamt für Stale-Counter-Limit (SCL) Makrozyklen. Dann sendet die Sicherheitssteuerung Failsafe-Daten.

1. Ohne weitere Störungen, führt dies in Stale-Counter-Limit Makrozyklen zur Einstellung der Failsafe-Daten im SIF-DO Block. Eine etwaige Prozess technische Abschaltung wird bei diesem Szenario nicht verzögert.
2. Ist der H1-Link zum SIF-DO Block jedoch mit Störungen belegt, die immer wieder Sequenzen von bis zu Stale-Counter-Limit-1 Nachrichten verfälschen, so kann die Einstellung der Failsafe-Daten auf Grund einer Prozessanforderung um Stale-Counter-Limit Makrozyklen verzögert werden. Eine etwaige Abschaltung wird bei diesem Szenario nicht mehr verzögert, als dies durch die Störungen selber auch der Fall wäre.

Die maximale Reaktionszeit für die Verarbeitung eines FF-SIF Eingangssignals bis zur FF-SIF Ausgabe beträgt in einem System ohne Fehler für eine FF-SIF-Sicherheitsfunktion mit einem H1-Link:

$$t_{max\text{-}ff} = 2 \bullet MCT \qquad (4.63)$$

In den Zeiten enthalten ist die Logikverarbeitung der Sicherheitssteuerung. Nicht betrachtet sind hierbei die Verzögerungen, die in den Geräten durch die Messung und Filterung oder Verzögerung von Abschaltungen entstehen.

***Beispiel:***

| | | Makrozyklus x | | | | | | Makrozyklus x+1 | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| H1-Link | | | pub AI1 | ... | | pub DO1 | ... | | pub AI2 | ... | | pub DO2 | | |
| | | SIF-AI-FB | | | HFB | | SIF-DO-FB | SIF-AI-FB | | | HFB | | SIF-DO-FB | |
| Prozesswert | AI1 | AI2 | | | | | | DO1 | | | | | | DO2 als Reaktion auf AI2 |

*Abbildung 4.6: FF-SIF Reaktionszeit AI-Block Eingang bis DO-Block Ausgang*

Das Beispiel zeigt ein Szenario, bei dem der Prozesswertwechsel von AI1 nach AI2 in spätestens 2 Makrozyklen eine Reaktion des SIF-DO-FBs bewirkt.
Ob ein Function-Block Output-Daten erzeugt, ist ihm überlassen. So sind z.B. Anwendungen mit Mehrheitsentscheid über die Eingangsdaten eines Funktion-Blocks, auch die nicht verfügbaren, in der Praxis möglich. Beispielsweise kann eine 2oo3 Verknüpfung solange gültige Output-Daten liefern, wie zwei gültige Eingangswerte vorliegen.

Liegt kein oder nur noch ein gültiges Input-Datum vor, so würde ein solcher FF-SIF Block keine Daten mehr publizieren.

Die maximale Reaktionszeit mit einem H1-Link als Übertragungssystem ändert sich nicht durch die Anwendung solcher Verfügbarkeitsmechanismen.

In der praktischen Anwendung werden SIF Applikationen mehr als einen H1-Link für eine Sicherheitsfunktion benutzen. Neben den Beschränkungen der Teilnehmer eines H1-Links können dafür hauptsächlich Verfügbarkeitsaspekte angeführt werden. Nachfolgend wird beispielhaft eine solche Struktur aufgezeigt.

Zum Einsatz kommen 3 bzw. 5 unterschiedliche H1-Links. Die Eingangsdaten von den AI-Blöcken werden fehlertolerant und sicher verschaltet[197], so dass sowohl ein H1-Link oder auch ein AI-Block ausfallen kann, ohne die Funktion der Sicherheitsfunktion einstellen zu müssen. Bei Ausfall des H1-

[197] typische 2oo3 Struktur in der Prozessautomatisierung

Links werden sowohl Fehler der physikalischen Übertragungseinrichtung, wie auch Fehler der Anschaltung, hier dargestellt durch verschiedene COM Baugruppen, beherrscht.

Je nach Applikation können die DO-Blocks sowohl verfügbar, wie auch sicherheitsgerichtet betrieben werden. Beim verfügbaren Betrieb wird auch hier der Fehler eines H1-Links, als auch des DO-Blocks beherrscht. Zur Erhöhung der Verfügbarkeit kann auch noch der Betrieb der DO-Blocks an eigenen H1-Links (4+5) zur Anwendung kommen.

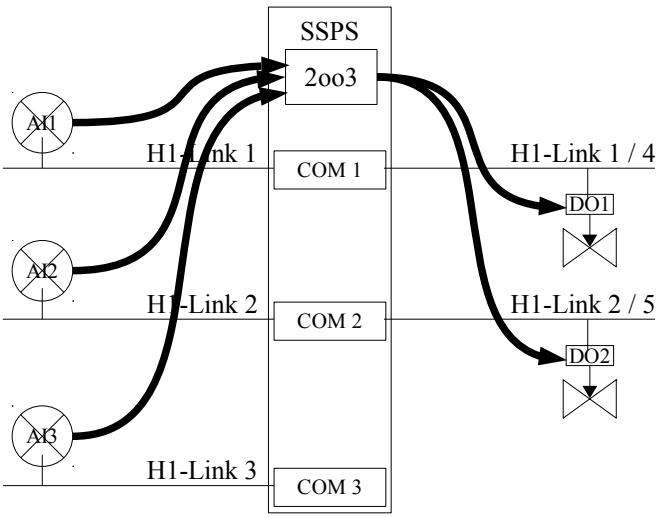


*Abbildung 4.7: FF-SIF Redundanzverknüpfung von H1-Links*

Die Verschaltungsart in Abbildung 4.7 impliziert, dass der Host-Function-Block der Sicherheitssteuerung nicht mit dem beispielhaft dargestellten 2oo3-Block übereinstimmen kann. Dies ist in den nicht dauerhaft synchronisierbaren Schedules/Makrozyklen der einzelnen H1-Links begründet. Zudem müssten die H1-Links alle die gleiche Makrozykluszeit und die gleiche Startzeit für den Host-Function-Block aufweisen, was technisch nicht sinnvoll realisierbar ist.

Damit diese für das Anwendungsgebiet Prozessautomatisierung erforderlichen Verschaltungen sicherheitsgerichtet eingesetzt werden können, wird in dieser Arbeit folgendes Vorgehen vorgeschlagen.

Aus Sicht eines H1-Links ergibt sich, dass dessen Host-Function-Block, außer dem Stale-Counter, zusätzliche Mechanismen benötigt, sich mit den anderen Host-Function-Blocks anderer H1-Links abzustimmen. Dabei sollen die folgenden Regeln angewendet werden:

1. Die Sicherheitssteuerung benutzt die Input-Daten eines Host-Function-Blocks in ihrer Logikverarbeitung erst, wenn sie gemäß Scheduling des HFB verarbeitet werden sollen. Somit wird der von FF zugesicherten Konsistenz der Daten Genüge getan. Link-übergreifende Konsistenz kann damit jedoch nicht erreicht werden.
2. Sind aktuelle Input-Daten zum Ausführungszeitpunkt des HFBs in der Sicherheitssteuerung nicht vorhanden[198], so werden die Daten als Fehlerzustand in der Logikverarbeitung der Sicherheitssteuerung übernommen. Das heißt, ein Stale>0 wirkt sich umgehend auf die Sicherheitssteuerung Logikverarbeitung aus, indem keine abhängigen Daten publiziert werden. Es findet dort keine Störaustastung statt.

[198] Stale-Counter > 0 oder es gibt zu diesem Zeitpunkt keine. Merke: der Stale-Counter muss gemäß [FF-SIF] erst erhöht werden, wenn innerhalb des Makrozyklusses keine Nachricht empfangen wurde. Dies ist insofern ein schwerer Fehler, denn der HFB muss zwar feststellen, dass für ihn kein Input aus dem aktuellen Makrozyklus zu seiner Ausführungszeit vorliegt, jedoch wird das Schließen der FF-SIF Verbindung erst nach seiner Ausführung durch ihn erkennbar und er kann darauf reagieren.

3. Sollen die FF-SIF-Input-Daten in eine Verarbeitung außerhalb von FF-SIF eingehen, so ist, sofern gewünscht, in der Logikverarbeitung eine Störaustastung zu realisieren. Für die Weiterverarbeitung in FF-SIF sollte dies jedoch nicht geschehen, da sonst die Reaktionszeit negativ beeinflusst wird.
4. Die Sicherheitssteuerung publiziert in ihrer Logikverarbeitung die Output-Daten eines HFB einmal je Makrozyklus des jeweiligen HFBs und zwar innerhalb der für die Ausführung des HFBs vorgesehenen Zeit und nachdem die Input-Daten des HFBs, sofern vorhanden, übernommen wurden.
5. In die publizierten Output-Daten gehen alle relevanten Daten mit ihrem jeweiligen Zustand ein, die zu dem Zeitpunkt des Publizierens verfügbar sind. Dies sind im Allgemeinen mehr als die Input-Daten des HFBs, sondern auch die anderer HFBs und ebenso aus nicht-FF-SIF Quellen (z.B. Sicherheitssteuerung-EAs).
6. Da die Sicherheitssteuerung Logikverarbeitung nicht synchron zu den Schedules aller FF-SIF-H1-Links ablaufen kann, wird nachfolgend eine zum FF-H1-Schedule unsynchronisierte, aber zyklische Sicherheitssteuerung-Logikverarbeitung vorausgesetzt.
7. Daher muss die geplante Ausführungsdauer der HFBs größer als 2 mal die maximale Ausführungsdauer der Sicherheitssteuerung Logikverarbeitung sein. Von der Sicherheitssteuerung wird erwartet, dass sie innerhalb der HFB Ausführungsdauer die Input-Daten des HFBs verarbeitet und die Output-Daten des HFBs publiziert.
8. Entweder ist $2 \bullet MCT$ der Input-Links < $MCT$ der Output-Links,
   oder[199]
   - $MCT$ des Output-Links „verbraucht“ keine Werte von $MCT$ des Input-Links
   - Werte von $MCT$ Input-Link bleiben für den Output-Link gültig, solange für diese auf dem Input-Link einen Stale-Counter von 0 haben.

Die maximale Reaktionszeit für die Verarbeitung eines FF-SIF Eingangssignals bis zur FF-SIF Ausgabe beträgt in einem System ohne Fehler mit Sicherheitsfunktionen mit n-Input-Links und m-Output-Links:

$$t_{max\text{-}ff} = 2 \bullet max\{MCT_{input\text{-}link}\} + 2 \bullet max\{MCT_{output\text{-}link}\} \qquad (4.64)$$

In den Zeiten enthalten ist die Logikverarbeitung der Sicherheitssteuerung. Nicht betrachtet sind dabei die Verzögerungen, die in den Geräten durch Messung, Filterung oder Verzögerung von Abschaltungen entstehen.

Die Gleichung drückt im ersten Term, die oben gezeigte maximale Reaktionszeit aus. Erst im 2. Makrozyklus des Input-Links steht der Wert für das Publizieren auf dem Output-Link zur Verfügung. Der Zeitpunkt, bei dem auf diesem publiziert werden kann, ist im ungünstigsten Fall gerade vorbei, so dass erst nach 2 Makrozyklen die Reaktion des Prozesswertes vom Input-Link bei einem FF-SIF-DO-FB des Output-Links erfolgen kann. Dies wird durch den 2. Term ausgedrückt. Dies entspricht dem Fall der Blockade im allgemeinen Reaktionszeitmodell.

Dabei sei nochmals ausdrücklich darauf hingewiesen, dass dies nur unter der Voraussetzung der mindestens zweifachen Sicherheitssteuerung-Logikverarbeitung innerhalb der HFBs gilt, d.h., durch die Sicherheitssteuerung wird gegenüber den FF-SIF Mechanismen keine zusätzliche Verzögerung erzeugt.

Solange die FF-SIF-Sicherheitsfunktion alleine durch einen HFB abgebildet wird, ändert sich die maximale Reaktionszeit gegenüber der obigen Betrachtung mit einen Link nicht. Der Unterschied entsteht, wenn mehrere Links für einer Sicherheitsfunktion betrachtet werden müssen.

[199] FF-SIF nutzt eine verbrauchende Wertesemantik, die sich jedoch nur bei synchroner Verarbeitung, d.h. eigentlich nur innerhalb eines H1-Links anwenden lässt.

Erst einmal behält der FF-SIF-DO-Function-Block seine gewohnte Störaustastungsfunktion mittels Stale-Counter. Da die Störaustastungszeit von der Makrozykluszeit, d.h. dem jeweiligen H1-Link abhängt, ist bei mehreren H1-Links nicht mehr automatisch die erwünschte Robustheit an jedem Ort der Sicherheitsfunktion gewährleistet.

Um dies zu erreichen, sollte das Stale-Count-Limit der FBs, die zu einer Sicherheitsfunktion gehören, so eingestellt werden, dass $SCL \bullet MCT_{FB}$ der FBs möglichst gleich ist und die *SCLs* ein für die Robustheit notwendiges Minimum nicht unterschreiten. Im ersten Schritt lässt sich aus der maximalen *MCT*, der am SIS-Link beteiligten FBs mal dem minimalen *SCL* die Reaktionszeit $t_{stale\text{-}max}$, wie bei einem H1-Link, berechnen. Der spezifische *SCL* eines FBs berechnet sich dann mit der Vorschrift (siehe auch Gleichung (4.62))

- wähle den größten *SCL*, so dass sein
  $$MCT \bullet (2+SCL) \leq t_{stale\text{-}max} \qquad (4.65)$$
  ist.

Die maximale Reaktionszeit auf ein Eingangssignalwechsel mit einer Kommunikationsunterbrechung ergibt sich zu

$$t_{wc\text{-}ff} = 2 \bullet max\{MCT_{input\text{-}link}\} + (2+SCL_{FF\text{-}SIF\text{-}DO}) \bullet (MCT_{FF\text{-}SIF\text{-}DO}) \qquad (4.66)$$

$MCT_{input\text{-}link}$ bezeichnet hierbei die *MCT* der anderen Links, deren FBs Input-Daten für die betrachtete Sicherheitsfunktion publizieren. Gibt es keine solche FBs, so erhält man wieder die bekannte Gleichung (4.62).

Input-Daten, die für eine Sicherheitsfunktion über eine Link-Grenze hinweg übertragen werden, verlängern die Reaktion um ihre doppelte *MCT*.

Gegenüber der maximalen durch FF-SIF bestimmten Reaktionszeit, sind für die Betrachtung der Sicherheitsfunktion auch die Zeiten der Eingangs- und Ausgangsgeräte relevant.

Hierbei werden auch die bislang ausgesparten Eingangs- und Ausgangsdatenverarbeitungen und etwaige Filterzeiten berücksichtigt.

Angenommen werden in den FF-SIF-Devices, dass sie maximal eine Zeit von 2 ihrer internen Zyklen benötigten, um einen Wert zu bestimmen bzw. diesen auszugehen. Diese Zeit wird mit $t_{max\text{-}in}$ und $t_{max\text{-}out}$ bezeichnet.

Die Filter- und Verzögerungszeiten werden mit $t_{max\text{-}filter}$ bzw. $t_{max\text{-}delay}$ bezeichnet. Bei diesen ist darauf zu achten, dass die effektiven maximalen Zeiten in Abhängigkeit von der jeweiligen Implementierung größer als die eingestellten FF-SIF Parameter sein können. Dies ist in der jeweiligen Herstellerdokumentation dargelegt. Es wird weiterhin angenommen, dass in den Filter- und Verzögerungszeiten auch die durch den Geräteaufbau bedingten Latenzen, z.B. Eingangsmessstufe und Ausgangsrelais, enthalten sind.

Die gesamte maximale Reaktionszeit bei FF-SIF-Sicherheitsfunktionen ergibt zu:

$$t_{max} = t_{max\text{-}in} + t_{max\text{-}filter} + t_{max\text{-}ff} + t_{max\text{-}delay} + t_{max\text{-}out} \qquad (4.67)$$

Für $t_{max\text{-}ff}$ ist, je nach dem, ob ein oder mehrere H1-Links in der Sicherheitsfunktion genutzt werden, der Ausdruck aus den Gleichungen (4.62) oder (4.64) ein zu setzten.

Typische Common-Cause Probleme der FF-SIF Kommunikation sind im Bereich Stromversorgung[200], EMV und Busstörungen möglich. Ebenfalls durch das Einfehlerszenario nicht abgedeckt sind transiente Busstörungen, d.h. solche, die zwar den Stale-Counter ansteigen lassen, aber unterhalb des Limits bleiben. Diese können dennoch zeitlich so eng aufeinander auftreten, dass sie eine Anforderung entlang der Komponenten einer Sicherheitsfunktion mehrfach betreffen.

[200] wird über den gemeinsamen Kommunikationsbus geliefert

Hier können mangelhafte Verkabelung, unzulässige elektromagnetische Störungen oder Ähnliches die Ursache sein.

Möchte man das Risiko eines derartigen Fehlerszenarios nicht eingehen, so berechnet sich die Worst-Case2 Reaktionszeit einer FF-SIF-Sicherheitsfunktion mit mehr als einer Fehlerauswirkung zu:

$$t_{wc2} = t_{wc\text{-}in} + t_{wc\text{-}filter} + t_{wc\text{-}ff} + t_{wc\text{-}delay} + t_{wc\text{-}out} \qquad (4.68)$$

Dabei stehen die Zeiten $t_{wc\text{-}in}$ für die Worst-Case Zeit der Eingabeverzögerung, $t_{wc\text{-}filter}$ die Worst-Case Zeit bedingt durch Filter, $t_{wc\text{-}ff}$ die Worst-Case Zeit von Foundation-Fieldbus-SIF, $t_{wc\text{-}delay}$ die Worst-Case Zeit bedingt durch die Ausgabezyklen und $t_{wc\text{-}out}$ die Worst-Case Zeit bedingt durch die Ausgabeverzögerungen.

In den Zeiten ist die Logikverarbeitung der Sicherheitssteuerung enthalten. Nicht berücksichtigt ist der zulässige Fehler des H1-Link-Schedulings, der Zeitsynchronisation und der Data-Stale-Timer und dessen Jitter. Dies wird nachfolgend diskutiert.

Die Sicherheitszeit der Sicherheitssteuerung braucht im Allgemeinen für die reinen FF-SIF Datenverarbeitung nicht weiter betrachtet werden, da ein Verzug oder Ausfall der Sicherheitssteuerung sich als „Nicht-Publizieren“ bzw. „nicht zeitgerecht Publizieren“ darstellt und somit von Stale-Counter-Konzept aufgefangen wird.

Bei gemischten Datenverarbeitungen, d.h. Nicht-FF-SIF-Input bzw. Nicht-FF-SIF-Output, sind die entsprechenden Zeiten, sowie die anzuwendenden Berechnungsformeln der Herstellerdokumentation zu entnehmen.

Weiter ist zu betrachten, wie sich kontinuierliche Busstörungen, die aber noch alle *SCL*-Zyklen eine korrekte Sendung durchlassen, auf die Reaktionszeit auswirken.

Dabei gilt für eine Sicherheitsfunktion auf einem H1-Link $t_{wc\text{-}ff} = (2 + SCL) \bullet MCT$, wieder ohne Filter und sonstige Verzögerungen. Das entspricht dem Modell, dass genau alle $(SCL+1)$ Makrozyklen genau ein gültiger Wert verarbeitet wird, dessen Alter jedoch maximal $2 \bullet MCT$ beträgt.
Für eine Sicherheitsfunktion aus mehreren H1-Links gilt (siehe auch Gleichungen (4.66), (4.68)):

$$\begin{aligned} t_{wc2} \quad = \quad & t_{wc\text{-}in} + t_{wc\text{-}filter} + \\ & 2 \bullet max\{MCT_{input\text{-}link}\} + (2+SCL_{FF\text{-}SIF\text{-}DO}) \bullet (MCT_{FF\text{-}SIF\text{-}DO}) + \\ & t_{wc\text{-}delay} + t_{wc\text{-}out} \end{aligned} \qquad (4.69)$$

Das Abtasttheorem bezüglich *SCL* kommt hier nicht zur Anwendung, da bei FF-SIF Nachrichten immer aus dem selben Macro-Zyklus stammen müssen und ihr Alter daher nicht über diese Zeit hinaus anwachsen kann.
Neben dem Stale-Counter gibt es in FF-SIF noch das Konzept des Stale-Data-Timers. Wird in der für ihn parametrierten Zeit keine gültige Nachricht empfangen, so leitet der Function-Block seine sichere Reaktion ein.

Mathematisch betrachtet könnte der Stale-Data-Timer auf $SCL \bullet MCT$ eingestellt werden. Dies ist, will man die Verfügbarkeit des parametrierten *SCLs* gewähren, problematisch. Um das Problem zu verstehen, muss man zwei aufeinander folgende Ausführungszeitpunkte eines FBs betrachten. Da die Bestimmung und Koordination des Starts der eigentlichen FB Ausführung nicht mathematisch präzise abläuft, kommt es dabei zu Schwankungen bzgl. des geplanten Zeitpunkts. Ist die Schwankung bei der ersten Ausführung gering hinter dem eigentlichen Startzeitpunkt und bei der zweiten Ausführung größer, d.h. etwas weiter hinter dem eigentlichen Startzeitpunkt, so sind diese beiden Zeitpunkte größer als die *MCT*. Eine Zeitüberwachung mit genau der *MCT*, ist also nicht realisierbar (siehe Abbildung 3.9).

Den selben Effekt erhält man bei Betrachtung der Ausführungsdauer des FBs, auch diese ist nicht konstant und schwankt im Allgemeinen in einem tolerierten Bandbereich. Auch dabei sind zwei aufeinander folgende Endzeitpunkte des FBs möglicherweise mehr als *MCT* voneinander entfernt.

Die Stale-Data-Time ist folglich auf

$$SDT=SCL \bullet MCT_{output} + Jitter_{max} \qquad (4.70)$$

einzustellen.

Daraus resultiert jedoch auch, dass dann die Stale-Data-Time die Worst-Case Reaktionszeit für die 1-Link-Betrachtung darstellt. Bei mehreren Links wird, wie oben, nochmals $2 \bullet max\ \{ MCT_{input\text{-}link} \}$ hinzugerechnet.

$$\begin{aligned} t_{wc2} \;=\; & t_{wc\text{-}in} + t_{wc\text{-}filter} + \\ & 2 \bullet max\ \{ MCT_{input} \} + SDT + MCT_{output} + \\ & t_{wc\text{-}delay} + t_{wc\text{-}out} \end{aligned} \qquad (4.71)$$

Jeder Publisher/Subscribe FF-SIF-Verbindung ist ein eineindeutiger 32-Bit Connection-Key zugeordnet. Dieser wird nicht in der Nachricht übertragen, geht aber in die CRC Berechnung[201]. Eine fehlerhafte Adressierung ist folglich nicht von einem CRC-Fehler zu unterscheiden.

Die Restfehlerwahrscheinlichkeit für das Aufdecken von Adressierungsfehlern ist unterschiedlich zur der für Nachrichtenverfälschungen, da die doppelte Übertragung hierbei nicht zur Geltung kommt.

Die Restfehlerwahrscheinlichkeit für die Akzeptanz einer falsch adressierten Nachricht ist $2{,}3283 \bullet 10^{-10}$. Hier wirkt nur noch der Faktor des CRCs von $1/2^{32}$.

Somit werden die Ansprüche bezüglich der Prüfung/Aufdeckung der Authentizität hinreichend erfüllt.

Neben Fehlern des Black-Channels und dass ein CRC über die Nachricht zufällig zum Connection-Key passt, ist als hauptsächliche Ursache für die falsche Adressierung das Engineering-Tool zu sehen. Dieses steht in der Verantwortung eineindeutige Connection-Keys zu generieren und dann den richtigen beteiligten Kommunikationspartnern mitzuteilen. Bei der Zuteilung des „richtigen“ Kommunikationspartners kommt auch dem Anwender eine entscheidende Verantwortung zu, da er für die korrekte Adressierung der Geräte zuständig ist.

Der Sicherheitssteuerung sollte die Aufgabe zukommen, alle ihr bekannten Connection-Keys auf Eineindeutigkeit zu prüfen oder, besser noch, die aus Connection-Key und Object-Index berechneten CRCs auf Eineindeutigkeit zu prüfen. Diese berechneten CRCs kommen als Startwert für die Nachrichten CRC-Berechnung zum Einsatz.

Der Object-Index kommt bei der Betrachtung der Authentizität nicht zur Anwendung, da er, mindestens bei den Geräten gleicher Bauart kein Unterscheidungsmerkmal darstellt.

Im betrachteten Einsatzgebiet von FF-SIF kommt es eine Mischung von Safety- und Standardnachrichten. Somit ist ein Schutz zur Unterscheidung der beiden Klassen von Nachrichten erforderlich. Dazu verwendet FF-SIF folgende Technik an:

Das Datenpaket innerhalb der FF-SIF Nachricht hat den Aufbau $2 \bullet \{FF\text{-}Daten + MCN + CRC\}$. Dabei wird angenommen, dass einerseits Standard-FF Daten nicht diesem Aufbau entsprechen und dass andererseits Standardkomponenten dieses Nachrichtenformat nicht erzeugen. Vor dem Hintergrund von FF-SIF-C/S Verbindungen durch nicht sichere Komponenten (Engineering-Tool) trifft dies jedoch nicht zu. Hier müssen Connection-Key und gegebenenfalls die Data-Link-Time Erweiterung des CRCs zur Unterscheidung herangezogen werden.

Unterschieden werden die Nachrichten an den Stellen *MCN+CRC*, sowie der Kopie der beiden. Die FF-Daten und ihre Kopie werden nicht weiter betrachtet, da sie keine spezielle Wertecharakteristik aufweisen, die zur Identifikation gewinnbringend angesetzt werden können.

[201] [FF-SIF]

Damit trotz dieser Maßnahmen ein Masquerading nicht erkannt wird, müssen *2 • {MCN+CRC}* versagen. Dies kommt einem Fehler gleich, der genau einen 12 Byte großen Wert an der jeweiligen Stelle, in der Standardkommunikationssoftware erzeugt. Allein der Wertebereich ergibt eine maximale Restfehlerwahrscheinlichkeit pro Stunde von

$$\frac{1}{2^{96}} \bullet 3600 \bullet 840 = 3{,}8168 \bullet 10^{-23} \leq 1 \bullet 10^{-9} = \ 1\% \ \ von \ \ SIL3 \qquad (4.72)$$

Dabei ist 3600•840 die angesetzte Anzahl der Nachrichten je Stunde auf allen H1-Links. Dass der betrachtete Fehler zusätzlich an den richtigen Stellen in den Nachrichten der Standardkommunikationssoftware auftreten muss, trägt zur weiteren Reduktion der Restfehlerwahrscheinlichkeit bei.

Bezieht man die FF-SIF-C/S Verbindungen unsicherer Komponenten mit ein, so fällt die Datenverdopplung zur Unterscheidung aus und es verbleiben nur noch *MCN+CRC*. Daraus ergibt sich eine Restfehlerwahrscheinlichkeit von $1{,}0743 \bullet 10^{-8}$. Dies entspricht nicht SIL3. Daraus folgt, dass den FF-SIF-C/S Verbindungen nicht erlaubt werden darf, das gleiche Protokoll zu benutzten.

Beim Einsatzgebiet von FF-SIF handelt es sich um eine geschlossene Übertragungseinrichtung, sofern diese ordnungsgemäß aufgebaut wurde. Da dies jedoch in den Bereich der vorhersehbaren Fehlbedienung beim Aufbau oder Umbau fällt, werden die Fehlermöglichkeiten offener Kommunikationseinrichtungen nachfolgend betrachtet.

Die Forderung nach einer geschlossen Übertragungseinrichtung fehlt in den FF-SIF-Dokumenten[202].

Betrachtet werden hier Bedienmöglichkeiten, die dazu führen können, dass die FF-SIF-Protokoll Mechanismen ausgehebelt werden. Die Bedieneingriffe bei FF-SIF beschränken sich im Wesentlichen auf den Aufbau der Kommunikationseinrichtungen, der Parametrierung der beteiligten Geräte und der Sicherheitssteuerung, Entsperren nach Fehlerreaktion (LO) und dem Asset-Management durch C/S-Verbindungen über Linking-Devices. Eine Sicherheitssteuerung kann auch ein Linking-Device darstellen.

Der Zugriff auf ein FF-SIF-Gerät wird über das Write-Lock gesteuert[202]. Ist dies gesetzt, so können keine FF-SIF-Parameter und Modi geändert werden. Das Rücksetzen des Write-Locks ist jedoch ohne weitere Zugriffsschutzmaßnahmen möglich und gibt dann die FF-SIF Parameter und Modi zum Ändern frei. Beim Aufheben des Write-Locks wird keine sichere Reaktion eingeleitet, sondern lediglich ein Alarm abgesetzt.

Damit diesen Gefahren begegnet werden kann, muss der Anwender einen ausreichenden Zugangsschutz zum H1-Link gewährleisten. Hierbei können beispielsweise Berechtigungsmechanismen der eingesetzten Sicherheitssteuerungen und Linking-Devices, so vorhanden, verwendet werden.

Ist ein solcher Zugangsschutz in diesen nicht vorhanden, so sind die Mechanismen auf höheren Ebenen umzusetzen. Dabei sind jedoch meist wesentlich umfangreichere Maßnahmen erforderlich, da die nächste Ebene i.d.R. Ethernet als Kommunikationsmittel verwendet.

Offene Übertragungssysteme haben neben der Gefährdung des Zugriffs auch die Eigenschaft nicht definieren zu können, welche anderen Protokolle benutzt werden. Für den Anwendungsbereich FF-SIF wird nur vom FF-H1-Standardprotokoll und aufgesetztem FF-SIF-Protokoll ausgegangen.

Andere Protokolle, wie z.B. PROFIBUS-PA/PROFIsafe, könnten die selbe Übertragungseinrichtung nutzen, nimmt man fehlerhafte Netzwerkverkabelung oder Parametrierung an.

Masquerading von PROFIBUS Standardnachrichten wird mit der selben Qualität, wie das von FF-H1-Standardformaten beherrscht. PROFIsafe unterscheidet sich in seinen Sicherungsmechanismen und kann bzgl. Masquerading ebenso wie PROFIBUS-Standardnachrichten betrachtet werden.

---

[202] [FF-SIF]

Da das FF-H1 Übertragungssystem jedoch über Linking-Devices von Ethernet aus erreichbar sein kann, wird ein CRC verwendet, der auch bei Standardprotokollen zum Einsatz kommt. Dies verringert die Fähigkeit FF-SIF-Nachrichten von Standardnachrichten zu unterscheiden. Daher sollte für das FF-SIF Protokoll ein CRC zur Anwendung kommen, der nicht in Standardübertragungssystemen (Ethernet) verwendet wird.

So wie zuvor die versehentliche Mischung von Standard- und Safety-Nachrichten ausgeschlossen wurde, so ist sie doch bei absichtlicher Beeinflussung zu betrachten.

Da FF-SIF nicht für offene Übertragungseinrichtungen ausgelegt wurde, gibt es keine spezifischen Mechanismen, die zum Zweck des Schutzes vor absichtlicher Unterminierung von Sicherheitsmechanismen des Protokolls in den Funktionsumfang des Protokolls aufgenommen wurden.

Ziel einer solchen Unterminierung könnte es sein, eine Nachricht in das Übertragungssystem einzuschleusen, die vom Empfänger als korrekt akzeptiert wird. Voraussetzungen dazu sind die Kenntnis der Funktionsweise von FF-SIF, der physikalischer Zugang, z.B. mit einem Linking-Device, in das Übertragungssystem und ein Kommunikationsknoten, der in der Lage ist, den Nachrichtenverkehr in Echtzeit zu analysieren.

Eine Unterminierung der Sicherheitsmechanismen von FF-SIF ist am wahrscheinlichsten durch den Einsatz einer FF-SIF Strategie umsetzenden Software. Damit dies Aussicht auf Erfolg hat, reicht es nicht aus die Funktionsweise von FF-SIF zu kennen, vielmehr müssen auch die Connection-Keys der einzelnen Verbindungen bekannt sein. Diese können durch Rückrechnung des CRCs aus mitgehörten Nachrichten ermittelt werden. Ebenso möglich ist das Auslesen des Keys aus dem FF-SIF-Link-Object[202]. Dies stellt keinen wirklichen Schutz dar.

Dieses Szenario wird von den FF-SIF-Protokollmechanismen nicht beherrscht, sondern bedarf des Zugriffsschutzes des Systems als Ganzes. Hierbei ist einerseits der physikalische Zugriffsschutz der sicherheitsgerichteten Übertragungseinrichtungen, aber auch der über externe Übertragungseinrichtungen, z.B. ein indirekter Zugang über ein HSE Linking-Device, zu betrachten.

Eine weitere Möglichkeit ist die Sabotage der Funktion als Ganzes. Dafür in Frage kommen neben der physischen Beschädigung, Techniken für die Störung der Übertragungseinrichtung durch Senden außerhalb des Schedules oder Versenden von falschen Time-Distribution-Nachrichten.

Die unberechtigte Änderung der Parameter der Geräte stellt eine weitere wesentliche Gefährdung der FF-SIF-Geräte dar, da die Auswirkung im Gegensatz zu Unterbindung der Funktion als Ganzes, die sicherheitsgerichteten Funktionen direkt[203] gefährlich beeinflussen kann.

Auch bei diesen Szenarien ist der Zugriffsschutz des Systems als Ganzes der entscheidende Faktor.

Beim FF-SIF ist die Fehlerfreiheit der Konfiguration besonders zu beachten. Von der FF-SIF Spezifikation wird gefordert, dass die Konfiguration durch das FF-SIF C/S-Protokoll in die Geräte geladen wird[204]. Dieser Anspruch schadet der Konfiguration zwar nicht, stellt aber ein Gefühl der Sicherheit her, die an dieser Stelle nicht angebracht ist. Betrachtet werden muss, dass die Konfigurationsinformationen von Standardgeräten (Engineering-Tool-PC) generiert und von diesen, mit dem FF-SIF-Protokoll, in die Geräte transportiert wird. Hier fehlt ein Konzept, dass die Integrität der gesamten FF-SIF Konfiguration eines Gerätes sicher stellt.

Ebenso wird gefordert, dass sich der Anwender von der ordnungsgemäßen Funktion/Parametrierung der Geräte vor der Inbetriebnahme überzeugt. Dies muss vor dem Hintergrund von Gerätetauschszenarien als äußerst fragwürdiges Konzept angesehen werden, da damit zu rechnen ist, dass eben diese Verifikation der ordnungsgemäßen Parametrierung absichtlich oder fahrlässig unterlassen wird.

---

[203] Filter-Zeiten, Fehlerhaltezeit, Ersatzwert, Simulation

[204] [FF-SIF]

Es liegt in der Verantwortung des Anwenders jedem Gerät die richtige eineindeutige Adresse zuzuteilen. Das FF-SIF-Protokoll kann nicht erkennen, wenn z.B. zwei Geräte/Blöcke vertauscht wurden, insbesondere dann nicht, wenn es sich um baugleiche Geräte handelt.

Eine Parametrierung von für den zu steuernden/überwachenden Prozess unzulässig große *MAX_STALE_COUNT, STALE_DATA_Time* und *FAULT_Time* Zeiten sind ebenfalls als fehlerhafte Konfiguration zu werten. Diese führt im Fehlerfall zu einer unzulässig langen Reaktion auf diese Fehler.

Ein Missbrauch der Verwendung von FF-SIF ist insbesondere durch eine absichtliche fehlerhafte Konfiguration der Überwachungszeiten vorstellbar.

Möglichkeiten das Protokoll FF-SIF im vorgesehenen Einsatzgebiet zu umgehen, hat der Anwender dann, wenn die eingesetzten Geräte ihre Funktion sowohl via FF-H1-Standard, wie auch via FF-SIF anbieten.

Eine weitere Gefährdung ergibt sich, wenn der Anwender die Ausgabeeinheiten im Simulationsmodus (Manual Override) betreibt und damit die Sicherheitsfunktion aushebelt.

Eine absehbare Fehlbedienung ist vorstellbar durch:

1. die Verwendung nicht eineindeutiger Adressen, durch Hinzufügen solcher Geräte in ein Übertragungssystem, insbesondere, wenn mehr als ein Gerät zeitgleich getauscht werden muss,
2. die unterlassene Verifikation der Geräteparametrierung bei Gerätetausch.

Den letzten Punkt könnte durch das oben angesprochene Konzept der Integrität der Konfiguration wesentlich verbessert werden. Würde das Gerät eine Signatur über seine Parametrierung rechnen, so könnte es einerseits Fehler beim Konfigurationsdownload besser aufdecken und andererseits hätte der Anwender durch Auslesen der Signatur die Gewähr, die richtige Konfiguration im Gerät zu haben und das, ohne das Gerät beim Tausch erneut vollständig prüfen zu müssen.

Gegenüber den vorherigen Kapiteln sind für das relevante Einsatzgebiet keine weiteren Betrachtungen erforderlich.

Zusammenfassend verbleiben für den Anwender neben allgemeinen Anforderungen, die aus dem Einsatz von sicherheitsgerichteten Anwendungen herrühren, dafür zu sorgen, dass,

1. der physikalische Zugriffsschutz auf die sicherheitsgerichtete Übertragungseinrichtung gewährleistet ist,
2. der externe (Netzwerk) Zugriffsschutz die Mittel des Systems, nur berechtigten Personen Zugriff zu gewähren, einsetzt,
3. das externe Netzwerk ausreichend geschützt ist, so dass von außen keine Angriffe und oder Fehlbedienungen auf dieses erfolgen können,
4. die Überwachungszeiten passend zur sicherheitsgerichteten Anwendung parametriert sind,
5. dass für den Gerätetausch ein geeignetes Prozedere definiert und eingehalten wird,
6. dass Bedieneingriffe definierten Regularien unterliegen und diese kontrolliert werden.

Die Entwicklung und Verifikation von FF-SIF ist im Rahmen einer IEC 61508 / SIL3 konformen Produktentwicklung vorzunehmen. Der dabei zum Einsatz gebrachte Entwicklungsprozess muss die Anforderungen der IEC 61508 erfüllen. Eine Konzeptprüfung für FF-SIF durch den TÜV Rheinland liegt vor, ist jedoch noch an die weitere Entwicklung des Protokolls anzupassen.

# 5 Transport Safety Protokoll

Die folgenden Darlegungen spezifizieren das Transport Safety Protokoll (TSP) für den Einsatz mit unterschiedlichen Bussystemen, das im Rahmen dieser Forschungsarbeit entwickelt wurde.

Das TSP Protokoll ist für den Einsatz der sicherheitsgerichteten Datenübertragung innerhalb von Automatisierungssystemen vorgesehen. Das TSP Protokoll soll bis zum Level SIL3 nach IEC 61508 und bis zur Kategorie CAT4/PL-e nach IEC 13849 eingesetzt werden können.

Das Transport Safety Protokoll weist folgende Leistungsmerkmale auf:

- 8/238 Byte maximale Nutzdatenmenge bei Bitfehlerraten von $10^{-2}$
- 120/238 Byte maximale Nutzdatenmenge bei Bitfehlerraten von $10^{-3}$
- 6/12 Bytes Protokoll-Overhead
- 4094/65534 Kommunikationsverbindungen
- IEC 61508-SIL3, auch bei Bitfehlerraten von $10^{-2}$ bzw. $10^{-3}$
- IEC 13849-CAT4/PL-e
- Black-Channel-Prinzip
  (keine speziellen Übertragungseinrichtungen oder Redundanzen erforderlich)
- Automatischer Wiederanlauf des Protokolls ist gestattet, vorausgesetzt die Applikation gestattet dies
- Master-Slave-Betrieb
- Multi-Master-Betrieb (ein Slave-Knoten kommuniziert gleichzeitig mit mehreren Mastern)
- TSP-Gateway-Betrieb (ein Slave-Knoten ist gleichzeitig Master für andere Slave-Knoten)
- TSP darf in geschlossenen Übertragungssystemen betrieben werden, in denen auch nicht sicherheitsgerichtete und andere sicherheitsgerichtete Protokolle eingesetzt werden.

Hauptziel des Designs, der Modellirung und der Entwicklung von TSP war der Entwurf eines sicherheitsgerichtetes Protokolls, das ein möglichst gutes Verhältnis von Nutzdaten zu Protokoll-Overhead aufweist.

Des Weiteren war ein Ziel, dass auch größere Datenmengen sicherheitsgerichtet übertragen werden können und das Protokoll aber eine möglichst einfache Implementierung ermöglicht.

Da sich vor dem Hintergrund der sicherheitsgerichteten Datenübertragung die Ziele geringer Protokoll-Overhead und größere Datenmengen ausschließen, wurden zwei Nachrichtenformate TSP1 und TSP2 festgelegt. Die Mechanismen des TSP konnten ungeachtet der beiden Formate annähernd gleich gehalten werden und so eine Implementierung vereinfachen.

### TSP1 Nachricht

Als Nachrichtenformat mit geringem Protokoll-Overhead wird das TSP1 Format definiert.

| Connection-ID [12 Bit] | OkBit [1 Bit] | Event1 [2 Bit] | Seq-No-LSB [1 Bit] | Nutzdaten [8-960 Bit] | CRC1 [32 Bit] |
|---|---|---|---|---|---|

*Abbildung 5.1: TSP1 Nachrichtenformat*

**Nachrichtenfelder:**

| | |
|---|---|
| Connection-ID | Identifikation einer sicheren Kommunikationsverbindung zwischen zwei Knoten Master und Slave. Wertebereich von 1 bis 4094; 0 und 4095 sind reserviert. |
| OkBit | beim Prozessdatenaustausch: 1 = keine Fehlersignalisierung der Applikation, 0 = Fehlersignalisierung; die Bedeutung obliegt der Applikation. |

| | |
|---|---|
| | beim Verbindungsaufbau: 1 = Fragment-Ende, 0 = Nachricht stellt Fragment dar (siehe Open Handling) |
| Event1 | definiert die Art des Datenpakets, Codes für TSP1 Pakete (s.u.) |
| Seq-No-LSB | Least significant Bit der 32 Bit Sequence-Number |
| Nutzdaten | die zu übertragenden Nutzdaten (minimal 1 Byte, maximal 120 Bytes je nach Bitfehlerrate) |
| CRC1 | die Prüfsumme für das TSP1 Paket |

Aus diesem Aufbau resultieren Nachrichtenlängen von 56 bis 1008 Bits. Zwecks Unterstützung des Open-Konzepts (s.u.), muss die Nutzdatenlänge je Richtung mindestens 1 Byte sein.

Als 32 Bit CRC1 wird das im Anwendungsbereich von bis zu 1008 Bits propere Polynom 0x1_F192_2815 (natürliche Darstellung) verwendet.

Berechnet wird der CRC ausgehend von einem Startwert S über die nicht übertragenen 32 Bits der Sequence-Number, der Connection-ID, OkBit, Event1 und Seq-No-LSB und schlussendlich über die Nutzdaten. Der CRC wird in seiner gespiegelten Form[205] für die Berechnung genutzt.

Das Speicherabbild, über das der CRC1 gerechnet wird, ist folgendermaßen definiert.

| **Byte -4** | **Byte -3** | **Byte -2** | **Byte -1** | **Byte 0** | **Byte 1** | **Byte 2 bis n-4** [206] |
|---|---|---|---|---|---|---|
| Bit31-24 Seq-No | Bit23-16 Seq-No | Bit15-8 Seq-No | Bit7-0 Seq-No | Die ersten beiden Bytes des Datenpakets | | Nutzdaten (kann auch ein ungenutztes Byte sein) |

*Abbildung 5.2: TSP1 CRC Berechnung*

*Anmerkungen:*

- *Die Bytes -4 bis -1 werden nicht im Datenpaket übertragen.*
- *Das LSB der Sequence-Number ist aus Gründen der einfacheren Implementierung doppelt in der CRC1 Berechnung enthalten.*
- *Alle Informationen des TSP1 Pakets, mit Ausnahme der Nutzdaten, müssen in dem für Kommunikationsprotokolle üblichen Big-Endian Format verwendet werden.*
- *Das Endian-Format für die Nutzdaten ist in der Applikation festzulegen.*

Für das Nachrichtenfeld Event1 sind definiert:

*Tabelle 5.1: TSP Codes für das Nachrichtenfeld Event1*

| ***Code*** | ***Name*** | ***Bedeutung*** |
|---|---|---|
| 2#01 | OUTPUT_DATA_INDICATION | Daten des Masters für Output-Daten des Slaves und gleichzeitige Anfrage des Masters beim Slave nach Input-Daten |
| 2#10 | RESPONSE | Antwort des Slaves an den Master auf Anfrage 2#01 oder 2#11 |
| 2#11 | OPEN_INDICATION | Verbindungsaufbau-Anfrage des Masters an den Slave |
| 2#00 | TSP2 Format | Nachrichten-Format-Kennung |

[205] 0xA814_498F ist gespiegelte Darstellung von 0x1_F192_2815

[206] der Byte-Index n ist das letzte Byte des CRC1

## TSP2 Nachricht

Ein Ziel beim Entwurf des TSP2 Nachrichtenformats war es, größere Nutzdatenmengen, auch bei schlechten Bitfehlerraten, sicherheitsgerichtet transportieren zu können.

Dieser Nachrichtenrahmen ist für eine Nachricht im TSP2 Format wie folgt definiert:

| Conection-ID-High [12 Bit] | OkBit [1 Bit] | Event1 [2 Bit] | Seq-No-LSB [1 Bit] | Event2 [8 Bit] | reserved [4 Bit] | Connection-ID-Low [4 Bit] | Nutzdaten [8-1904 Bit] | CRC2 [32 Bit] | CRC3 [32 Bit] |
|---|---|---|---|---|---|---|---|---|---|

*Abbildung 5.3: TSP2 Nachrichtenformat*

**Nachrichtenfelder:**

| | |
|---|---|
| Connection-ID-High | die 12 höherwertigen Bits der Connection-ID |
| Connection-ID-Low | die 4 niederwertigen Bits der Connection-ID<br>Wertebereich der Connection-ID: 1 bis 65534; 0 und 65535 sind reserviert |
| OkBit | wie TSP1 |
| Event1 | Wert immer 2#00, kennzeichnet die Nachricht als TSP2-Format |
| Event2 | Art des Datenpakets (s.u.) |
| Seq-No-LSB | wie TSP1 |
| reserved | Reserviert, Bits werden mit 0 übertragen |
| Nutzdaten | Nutzdaten (minimal 1 Byte, maximal 238 Bytes) |
| CRC2 | ist die erste Prüfsumme für das TSP2 Paket |
| CRC3 | ist die zweite Prüfsumme für das TSP2 Paket |

Daraus resultieren Nachrichtenlängen von 104 bis 2000 Bits. Zwecks Unterstützung des Open-Konzepts (s.u.), muss die Nutzdatenlänge je Richtung mindestens 1 Byte sein.

Als 32 Bit CRC2 wird das propere Polynom 0x1_3258_3499 (natürliche Darstellung) verwendet. Das Polynom von CRC2 ist unabhängig vom CAN-CRC-Polynom und unabhängig vom beim Ethernet verwendeten IEEE-802 Polynom.

Als 32 Bit CRC3 wird das propere Polynom 0x1_F6AC_FB13 (natürliche Darstellung) verwendet. Das Polynom von CRC3 ist unabhängig vom CRC2 und unabhängig vom CAN-CRC-Polynom und unabhängig vom beim Ethernet verwendeten IEEE-802 Polynom.

Berechnet werden die CRCs ausgehend von einem Startwert S über die nicht übertragenen 32 Bits der Sequence-Number und dann über das übertragene Paket (ohne CRC2/3).

Die CRCs werden in ihrer gespiegelten[207] Form für die Berechnung genutzt.

Das Speicherabbild, über das die CRCs gerechnet werden, ist folgendermaßen definiert.

| **Byte -4** | **Byte -3** | **Byte -2** | **Byte -1** | **Byte 0-Byte 3** | **Byte 4 bis n[208]-8** |
|---|---|---|---|---|---|
| Bit31-24 Seq-No | Bit23-16 Seq-No | Bit15-8 Seq-No | Bit7-0 Seq-No | Die 4 Bytes des Datenpakets | Nutzdaten (kann auch ein ungenutztes Byte sein) |

*Abbildung 5.4: TSP2 CRC Berechnung*

[207] 0x1_992C_1A4C = gespiegelt: 0x3258_3499
0x1_F6AC_FB13 = gespiegelt: 0xC8DF_356F

[208] der Byte-Index n ist das letzte Byte des CRC3

*Anmerkungen:*

- *Die Bytes -4 bis -1 werden nicht im Datenpaket übertragen.*
- *Das LSB der Sequence-Number ist aus Gründen der einfacheren Implementierung doppelt in der CRC2 und CRC3 Berechnung enthalten.*
- *Alle Informationen des Datenpakets, mit Ausnahme der Nutzdaten, müssen in dem für Kommunikationsprotokolle üblichen Big-Endian Format verwendet werden.*
- *Das Endian-Format für die Nutzdaten ist in der Applikation festzulegen.*

Die Codes für das Nachrichtenfeld Event2 sind wie folgt definiert:

```
0x01              OUTPUT_DATA_INDICATION
0x02              RESPONSE
0x03              OPEN_INDICATION
0x00, 0x04-0xff   Werte sind reserviert
```

Die Bedeutung ist identisch zu TSP1.

Eine sicherheitsgerichtete Kommunikationsverbindung ist zwischen 2 Knoten, einem Master und einem Slave, definiert. Diese Kommunikationsverbindung wird durch die Connection-ID identifiziert.

Die sichere Adressierung wird durch die Informationen Connection-ID und Event1 bei TSP1 und durch die Informationen Connection-ID und Event2 bei TSP2 vorgenommen. Aus dem Event1 bei TSP1 und dem Event2 bei TSP2 lässt sich ableiten, ob eine Nachricht zu einer Verbindung vom Master zum Slave oder vom Slave zum Master versendet wurde. Der Master sendet immer INDICATIONs und der Slave immer RESPONSEs.

Die Connection-ID muss in einer Kommunikationsdomain ungeachtet der verwendeten TSP-Varianten eineindeutig sein. Eine Kommunikationsdomain ist dabei ein logisches Netzwerk, innerhalb dessen die sicherheitsgerichteten Nachrichten transportiert werden. Über dieses Netzwerk hinaus dürfen die Nachrichten nicht transportiert werden können.

Somit ermöglicht TSP1 innerhalb einer Kommunikationsdomain 2046 sicherheitsgerichtete Kommunikationsverbindungen. Bei TSP2 sind 65534 Kommunikationsverbindungen in einer Kommunikationsdomain möglich.

Dabei ist es auch möglich, dass zwischen 2 Knoten mehr als eine Kommunikationsverbindung besteht. Diese erhalten dann jeweils eine eigene Connection-ID.

Die Rolle, die ein Gerät des Automatisierungssystems bzgl. einer TSP-Verbindung einnimmt, kann Master oder Slave sein. Dies ist jede Kommunikationsverbindung wählbar. In Folge können Geräte z.B. als sicheres Gateway dienen oder als ein Slave im Multi-Master-Betrieb arbeiten, wenn die Applikation das Rangieren der Prozessdaten vornimmt (siehe Abbildung 5.5).

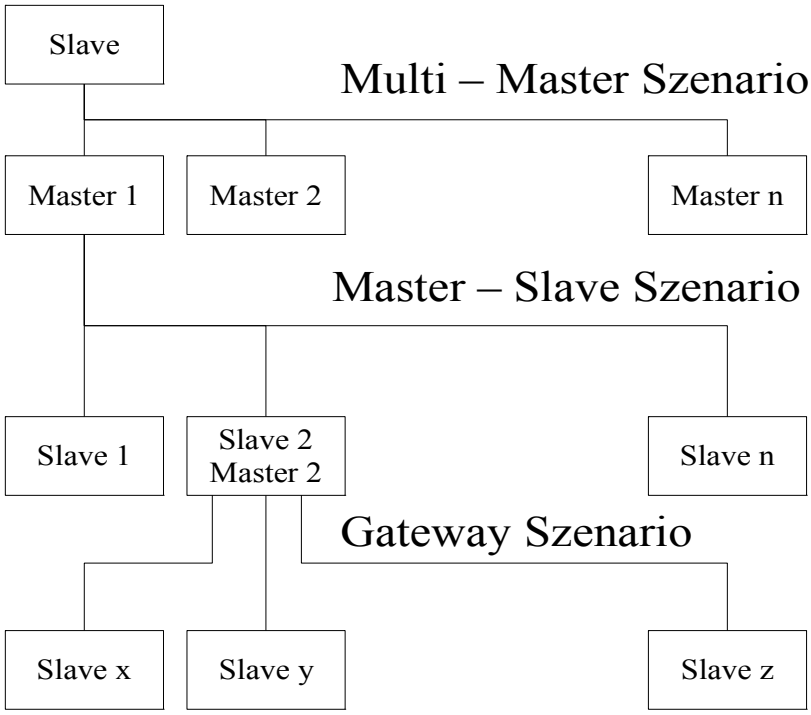


*Abbildung 5.5: TSP Anwendungsszenarien*

Bei einem Gateway ist ein Knoten Slave zu einem Master und gleichzeitig Master zu einer Menge von anderen Slave-Knoten.

Die Sequence-Number dient der Sicherstellung der monotonen Reihenfolge von Nachrichten. Mit jeder neuen Nachricht, die der Master generiert, inkrementiert der Master die Sequence-Number um 1. Der Slave antwortet auf eine korrekt empfangene Master-Nachricht mit der selben Sequence-Number, die der Master in seiner Nachricht benutzte. Die Sequence-Number ist ein umlaufender Zähler, der nach dem Erreichen von $(2^{32}-1)$ wieder 0 annimmt.

Somit haben Master und Slave jeweils immer eine Erwartungshaltung bzgl. der Sequence-Number. Die Verbindungsaufnahme startet mit einer vereinbaren Sequence-Number.

Wird diese Erwartungshaltung für eine Nachricht nicht erfüllt, so wird die Nachricht verworfen.

Ist der empfangene Teil der Sequence-Number (Seq-No-LSB) und der/die CRCs gleich zu der zuvor korrekt empfangenen Nachricht, so wird die empfangene Nachricht als Wiederholung der unterlagerten Transportschicht gewertet und ignoriert.

Die nachfolgend bei TSP Datenstrukturen verwendeten Datentypen sind in C/C++ folgendermaßen definiert:

- `typedef unsigned char ubyte;`
- `typedef unsigned short uword;`
- `typedef unsigned long udword;`

Die folgenden Symbole werden für die Abwicklung des Protokolls verwendet. Die konkrete Anwendung wird an der Stelle der Verwendung, in den Transitionen von Master und Slave, erläutert. Die Bedeutung der Symbole wird ebenfalls erst dort vollständig ersichtlich.

*Tabelle 5.2: TSP Symbole*

| *Wert* | *Symbol* | *Bedeutung* |
|---|---|---|
| 0x00 | **OPEN_IND_ABORT** | Allgemeine Abbruchsignalisierung der Verbindungsaufnahme |
| 0x01 | **OPEN_IND_FRAG_ACK** | Bestätigung eines **OpenIndData** Fragments |
| 0x02 | **OPEN_IND_UNDERFLOW** | Abbruchsignalisierung der Verbindungsaufnahme wegen zu wenig **OpenIndData** Daten. |
| 0x03 | **OPEN_IND_OVERFLOW** | Abbruchsignalisierung der Verbindungsaufnahme wegen zu vieler **OpenIndData** Daten. |
| 0x04 | **CONFIG_MISSMATCH** | Abbruchsignalisierung der Verbindungsaufnahme wegen unterschiedlicher Konfigurationen im Master und Slave. Weiterhin ist der Slave so konfiguriert, dass er keine Konfiguration vom Master erwartet. |
| 0x05 | **CONFIG_NOT_SUPPORTED** | Abbruchsignalisierung der Verbindungsaufnahme bei Konfiguration eines Slaves, der keine Konfiguration erwartet, aber vom Master eine bekam. |
| 0x06 | **CONFIG_DIFFER** | Abbruchsignalisierung der Verbindungsaufnahme wegen unterschiedlicher Konfigurationen im Master und Slave. Weiterhin ist der Slave so konfiguriert, dass er mit einer Konfiguration vom Master konfiguriert werden kann. |
| 0x07 | **PROTO_VERSION_NOT_SUPPORTED** | Protokollversion des Masters wird vom Slave nicht unterstützt. |
| 0x08 | **CONFIG_CANNOT_HANDLE** | Abbruchsignalisierung der Verbindungsaufnahme wegen mConfig-Konfigurationsdaten, die der Slave aus applikativen Gründen nicht annehmen kann. |
| 0x09 | **PROTO_VERSION_NOT_SUPPORTED** | Protokollversion des Masters wird vom Slave nicht unterstützt. |
| 0x0A | **EMPTY** | Vorbelegung |
| 0xAF | **ACCEPTED** | Verbindungsaufbau akzeptiert. |
| 0xFF | **OPEN_RESP_FRAG_ACK** | Bestätigung eines **OpenRespData** Fragments |
| 0x0000_0815 | **INITIAL_SEQUENCE_NO** | Initialwert für NextSeqNumber beim Verbindungsaufbau |
| 0x0000_5A47 | **INITIAL_MASTER_PRESET** | Initialwert für LastMasterPreset beim Verbindungsaufbau |
| 0xFFFF_A3B7 | **INITIAL_SLAVE_PRESET** | Initialwert für LastSlavePreset beim Verbindungsaufbau |

Die Verbindungsaufnahme verwendet ebenfalls den Aufbau der TSP1/TSP2 Nachrichten, nur dass hierbei innerhalb der Nutzdaten spezifische Informationen der Verbindung übertragen werden.

Der Master stößt die Verbindungsaufnahme durch das Versenden eines **`OpenIndData`**-Pakets zum Slave an. Der Slave beantwortet die Anfrage, mit einem **`OpenRespData`**-Paket, mit dem Ergebnis der Validierung. Passen die Daten **`OpenIndData`** der Open-Indication oder die Daten **`OpenRe-`**

**spData** der Open-Response nicht in den Nutzdatenbereich der TSP-Nachricht, so werden sie fragmentiert, d.h. mit mehreren TSP-Nachrichten, übertragen.

Die detaillierten Regeln der Verbindungsaufnahme werden in den Kapiteln 5.2 und folgende, für die Transitionen der Master-Zustandsmaschine und in den Kapiteln 5.1 und folgende, für die Zustandsmaschine des Slaves, dargelegt.

Die Nutzdatenstruktur der Open-Indication (C++-Notation, gepackt) enthält Informationen im Big-Endian Format.

```
struct OpenInd_DATA_t {
    ubyte    mOpenTMO;
    ubyte    mWDT[3];
    udword   mMasterPreset;
    udword   mSlaveConfigSignature;
    uword    mConnectionId;
    uword    mConfigSize;
    ubyte    mProtoVersion;
    udword   mOpenCrc;
    ubyte    mConfig[]; // optional
} OpenIndData;
```

Der Slave erkennt die TSP-Nachricht mit **OpenIndData** Daten an Event1/2=OPEN_INDICATION und seiner Erwartungshaltung und der folgenden positiven Prüfung der Daten.

- **mOpenTMO** wird verwendet für:

Timeout während des Verbindungsaufbaus. Die Einheit ist 2 Sekunden. Der Binärwert 0 entspricht 512 Sekunden. Damit können, passend zu `mWDT`, Timeouts zwischen 2 Sekunden und bis zu 8,5 Minuten eingestellt werden.

- **mMasterPreset** wird verwendet für:

Sicherheitskonzept für Anlauf und Wiederanlauf (Anwendung siehe auch Kapitel 5.3 Preset-Handling).

- **mConnectionId** wird verwendete für:

Identifikation der Verbindung; sichert ebenfalls, dass die **OpenIndData** Daten zur Verbindung gehören.

- **mWDT** wird verwendet für:

Die vom Slave verwendete Watch-Dog-Time in der Einheit 32 Mikrosekunden und unterstützt damit einen Wertebereich von bis zu rund 8,9 Minuten. Der Binärwert 0 hat die Bedeutung von $2^{24}$.

- **mProtoVersion** wird verwendet für:

Protokollversion die der Master benutzt. Zur Zeit ist nur der Wert 1 zulässig.

- **mSlaveConfigSignature** wird verwendet für:

Signatur über die Konfiguration des Slaves bezüglich dieser Verbindung. Damit wird beim Verbindungsaufbau geprüft, ob Master und Slave zueinander passend projektiert sind. Die Signatur ist vom Programmiertool geeignet zu berechnen.

- **mOpenCrc** wird verwendet für:

CRC mit Polynom CRC1 mit Startwert S=0xffff_ffff über die Bytes der Elemente mOpenTMO, mWDT[3], mMasterPreset, mSlaveConfigSignature, mConnectionId, mConfigSize und mProtoVersion. Damit wird die Konsistenz der fragmentierten Übertragung gesichert.

- **mConfigSize** wird verwendet für:

Länge der nachfolgenden Bytes in mConfig. Dabei kann **mConfig** maximal

```
65535 - sizeof( mOpenTMO, mMasterPreset, mSlaveConfigSignature,
mConnectionId, mWDT[3], mProtoVersion, mConfigSize, mOpenCrc)
```

Bytes groß sein. D.h. Open-Indication Daten incl. mConfig dürfen nicht größer als 65.535 Bytes sein.

- **mConfig** wird verwendet für:

Übertragung einer spezifischen Geräte (Slave) Parametrierung. Auch wenn die Übertragung hier gesichert erfolgt, muss doch das Gerät und das eingesetzte Programmiertool ein geeignetes Sicherheitskonzept verwenden, sofern die Daten in mConfig Sicherheitsrelevanz besitzen. Insbesondere wenn das Gerät über mConfig konfiguriert wird, sollte die Connection-ID in den mConfig Daten zwecks Identifikation enthalten sein.

Weitere Informationen zu den Werten der Elemente und deren Anwendungen sind den Transitions-Beschreibungen zu entnehmen.

Die Nutzdatenstruktur Open-Response (C++-Notation, gepackt) enthält Informationen im Big-Endian Format.

```
struct OpenResp_DATA_t {
    ubyte    mResult;
    ubyte    mProtoVersion;
    uword    mConnectionId;
    udword   mSlavePreset;
    udword   mSlaveConfigSignature;
    udword   mMasterPreset;
    udword   mOpenCrc;
} OpenRespData;
```

Der Master erkennt die TSP-Nachricht mit **OpenRespData** Daten an Event1/2=RESPONSE und seiner Erwartungshaltung und der folgenden positiven Prüfung der Daten.

- **mResult** wird verwendet für:

Prüfergebnis einer Open-Indication, sofern sie beantwortet wird. Dabei werden die Werte der Tabelle 5.2 verwendet. Die konkrete Anwendung wird in den Kapiteln über die Transitionen von Master und Slave erläutert.

- **mMasterPreset** wird verwendet für:

Sicherheitskonzept für Anlauf und Wiederanlauf.

- **mSlaveConfigSignature** wird verwendet für:

Quittierte Signatur der Open-Indication, sofern diese akzeptiert wurde (=ACCEPTED) und falls diese nicht akzeptiert wurde, entspricht der Wert der im Slave vorhandenen, d.h. der erwarteten Signatur bzw. 0, falls er keine Konfiguration besitzt aber eine vom Master erwartet.

- **mSlavePreset** wird verwendet für:

Sicherheitskonzept für den Anlauf und Wiederanlauf

- **`mProtoVersion`** wird verwendet für:

Damit signalisiert der Slave die maximal vom Slave unterstützte Protokollversion.

- **`mOpenCrc`** wird verwendet für:

CRC mit Polynom CRC1, mit Startwert S=`0xffff_ffff` über die Bytes der Elemente (Big-Endian Darstellung) in der Reihenfolge : mResult, mProtoVersion, mConnectionId, mSlavePreset, mSlave-ConfigSignature und mMasterPreset. Damit wird die Konsistenz der fragmentierten Übertragung gesichert.

Weitere Informationen zu den Werten der Elemente von **`OpenRespData`** und deren Anwendungen sind den Transitions-Beschreibungen zu entnehmen.

## 5.1 TSP Slave

Zu jeder TSP Kommunikationsverbindung müssen im Slave folgenden Informationen vorhanden sein, bevor das TSP Protokoll den Datenaustausch vornehmen kann:

- welches Nachrichtenformat, TSP1 oder TSP2, für die Verbindung zur Anwendung kommen soll,
- Nachrichtenlänge für SI_INPUT, sofern nicht durch Konfiguration mConfig bestimmt,
- Nachrichtenlänge für SI_OUTPUT, sofern nicht durch Konfiguration mConfig bestimmt,
- Erwartungshaltung bzgl. der Slave-Konfiguration mSlaveConfigSignature/mConfig,
- Connection-ID der Verbindung.

Die Informationen Nachrichtenformat, Connection-ID, Master/Slave-Rolle und Erwartungshaltung bzgl. Konfiguration müssen außerhalb des TSP im Slave sicherheitsgerichtet eingestellt und gespeichert werden.

Sofern die Nachrichtenlängen nicht über mConfig eingestellt werden, müssen diese auch außerhalb des TSP im Slave sicherheitsgerichtet eingestellt und in jedem Fall sicherheitsgerichtet gespeichert werden.

Nachfolgend werden die Slave Datenobjekte definiert:

- ALIVE_TIMER — ein Timer zur Alive-Überwachung der Verbindung, muss zur Messung der mWDT und mOpenTMO geeignet sein
- ActiveWDT — Startwert für ALIVE_TIMER während Prozessdatenaustauschs
- OpenTMO — Startwert für ALIVE_TIMER während Verbindungsaufnahme
- OpenIndData — siehe oben
- OpenRespData — siehe oben
- LastSlavePreset — Randomized Startwert für CRC Berechnung für Nachrichten an den Slave
- LastMasterPreset — Randomized Schlüssel des Masters als Startwert für Sequence-Number und Startwert für CRC Berechnung für Nachrichten an den Master
- MasterProtoVersion — für die Verbindung genutzte Protokollversion, z.Z. immer 1
- NextSeqNo — der Wert der Sequence-Number, die der Slave in der nächsten Nachricht vom Master erwartet bzw. für die RESPONSE benutzt
- SI_INPUT — sicherheitsgerichtete Daten vom Slave an den Master (optional)
- SI_OUTPUT — sicherheitsgerichtete Daten vom Master an den Slave (optional)

- SI_OK_BIT_IN — nimmt Wert des OkBit's aus TSP-Header an und kann von der Applikation lesend genutzt werden. Der Wert der Applikation des Masters ist nur im Zustand VALID_DATA verfügbar.
- SI_OK_BIT_OUT — wie SI_OK_BIT_IN, wird zum Versenden an den Master genutzt und muss von der Applikation belegt werden. Der Wert wird nur bei der Prozessdatenübertragung benutzt und steht dem Master nur im Master Zustand VALID_DATA zur Verfügung.
- Duplicate

```
struct {
        udword CRC1;  // CRC1 aus Nachricht
        udword CRC2;  // CRC2 aus Nachricht
        udword CRC3;  // CRC3 aus Nachricht
        bool   mSeqNoLsb;  // SeqNoLsb aus Nachricht
        bool   mValid;     // TRUE andere Elemente sind
                           // gültig, FALSE nicht gültig
} Duplicate;
```

Verwendung für die Erkennung von duplizierten Nachrichten. (siehe Hilfsfunktionen unten)

**Slave State-Maschine**

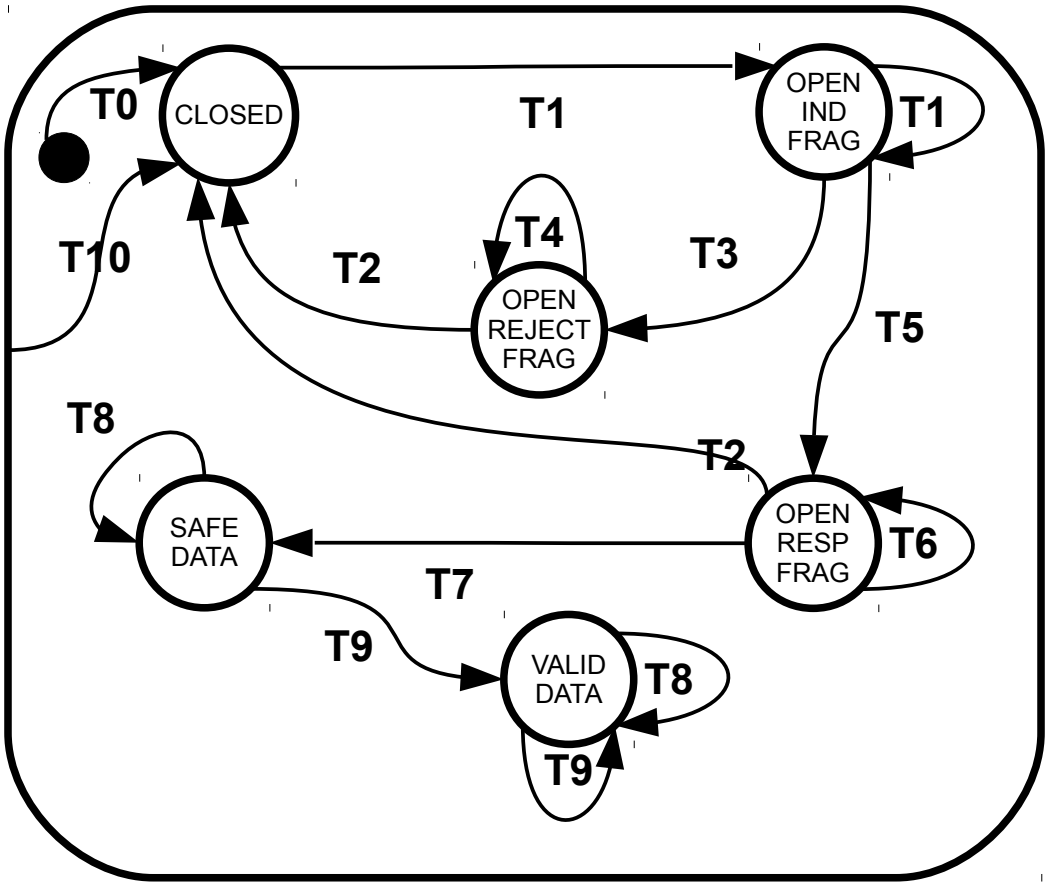


*Abbildung 5.6: TSP Slave Zustandsdiagramm*

Im Initialzustand ist das Protokoll nicht bereit, eine Kommunikationsverbindung aufzubauen und zu betreiben. Die der Kommunikationsverbindung zugeordneten sicheren SI_OUTPUT Daten haben ih-

ren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft nicht.

### Zustand: CLOSED

Das Protokoll ist bereit eine Kommunikationsverbindung aufzubauen, die der Kommunikationsverbindung zugeordneten sicheren SI_OUTPUT Daten haben ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft nicht.

### Zustand: OPEN_IND_FRAG

Die Verbindung wird aufgebaut und die dazu nötigen Daten werden vom Master empfangen und geprüft. Die der Kommunikationsverbindung zugeordneten sicheren SI_OUTPUT Daten haben ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft mit OpenTMO.

### Zustand: OPEN_REJECT_FRAG

Der Verbindungsaufbau wird abgebrochen und die dazu nötigen Daten werden an den Master geschickt. Die der Kommunikationsverbindung zugeordneten sicheren SI_OUTPUT Daten haben ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft mit OpenTMO.

### Zustand: OPEN_RESP_FRAG

Der Verbindungsaufbau wird akzeptiert und die dazu nötigen Daten werden an den Master geschickt. Die der Kommunikationsverbindung zugeordneten sicheren SI_OUTPUT Daten haben ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft mit OpenTMO.

### Zustand: SAFE_DATA

Das Protokoll ist geöffnet, es liegen jedoch noch keine SI_OUTPUT Daten des Masters vor. Daher haben die der Kommunikationsverbindung zugeordneten sicheren SI_OUTPUT Daten ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft mit ActiveWDT.

### Zustand: VALID_DATA

Das Protokoll ist geöffnet, es liegen gültige Daten vor und SI_OK_BIT_IN hat den Wert aus der letzten gültigen Prozessdatennachricht. Der Überwachungs-Timer ALIVE_TIMER läuft mit ActiveWDT.

### Hilfsfunktionen für Transitionen

Zur Vereinfachung der späteren Definition der Transitionen der Slave Statemaschine, folgen hier einige Hilfsfunktionen.

### Bestätige OpenIndData Fragment

**`AckOpenIndDataFragment`**`()`

Falls TSP1 Betrieb:

- Connection-ID=voreingestellte Connection-ID

- Event1= RESPONSE

Falls TSP2 Betrieb:

- Connection-ID-High/Low = voreingestellte Connection-ID
- reserved-Bits=0
- Event1=TSP2-Format
- Event2= RESPONSE

Gemeinsam:

- OkBit=0
- Seq-No-LSB=(NextSeqNo & 0x1)
- Fülle alle Nutzdatenbytes[209] mit OPEN_IND_FRAG_ACK
- CRC Berechnung mit Startwert LastSlavePreset und die Sequence-Number=NextSeqNo ausführen und in CRC1 respektive CRC2/3 eintragen.

Schlussendlich wird die Nachricht an den zur Connection-ID passenden Master versendet.

### Fülle OpenRespData

**`FillOpenRespData(result,masterPreset`**)

- `OpenRespData.mResult= result;`
- `OpenRespData.mMasterPreset= masterPreset;` // LastMasterPreset wird noch nicht gesetzt
- `OpenRespData.mSlavePreset= GeneratePreset();` // LastSlavePreset wird noch nicht gesetzt; siehe Kapitel 5.3
- `OpenRespData.mSlaveConfigSignature=` Wert der im Slave vorhandenen gültigen Signatur oder 0, falls der Slave auf eine Signatur/Konfiguration wartet.
- `OpenRespData.mConnectionId=` Wert der eigenen Connection-ID
- `OpenRespData.mProtoVersion=` Wert der maximal unterstützten Version
- `OpenRespData.mOpenCrc=` Wert des CRCs für **OpenRespData**

### Sende OpenRespData Fragment

**`SendOpenRespDataFragment`**(**`last`**)

Falls TSP1 Betrieb:

- Connection-ID=voreingestellte Connection-ID
- Event1= RESPONSE

Falls TSP2 Betrieb:

- Connection-ID-High/Low = voreingestellte Connection-ID
- reserved-Bits=0
- Event1=TSP2-Format
- Event2= RESPONSE

Gemeinsam:

- OkBit=`last`
- Seq-No-LSB=(NextSeqNo & 0x1)

[209] Die Länge der Nutzdaten ist eine Größe die fest im Gerät verankert ist oder durch seine Parametrierung bestimmt wird. Wird die Nutzdatenlänge der Prozessdaten (SI_INPUT/SI_OUTPUT) durch mConfig bestimmt, so ist eine Länge der Nutzdaten der Nachrichten des Verbindungsaufbaus zu vereinbaren.

- Fülle die Nutzdatenbytes mit dem nächsten Teil der **`OpenRespData`** Daten und falls ein Rest in den Nutzdaten[210] verbleibt, fülle diesen mit `0xff` (`last` muss dann `TRUE` sein)
- CRC Berechnung mit Startwert LastMasterPreset und die Sequence-Number=NextSeqNo ausführen und in CRC1 respektive CRC2/3 eintragen.

Schlussendlich wird die Nachricht an den zur Connection-ID passenden Master versendet.

### Sende Prozessdaten

**`SendData`**(**`okBit`**)

Falls TSP1 Betrieb:

- Connection-ID=voreingestellte Connection-ID
- Event1= RESPONSE

Falls TSP2 Betrieb:

- Connection-ID-High/Low = voreingestellte Connection-ID
- reserved-Bits=0
- Event1=TSP2-Format
- Event2= RESPONSE

Gemeinsam:

- OkBit=`okBit`
- Seq-No-LSB=(NextSeqNo & 0x1)
- Fülle die Nutzdatenbytes mit den Prozessdaten SI_INPUT. Falls keine Prozessdaten geschickt werden, wird das eine Nutzdatenbyte mit 0 gefüllt.[211]
- CRC Berechnung mit Startwert LastMasterPreset und die Sequence-Number=NextSeqNo ausführen und in CRC1 respektive CRC2/3 eintragen.

Schlussendlich wird die Nachricht an den zur Connection-ID passenden Master versendet.

### Prüfungen des TSP1 Headers

`bool` **`CheckTSP1Header`**(**`event`**)

1. Sollte von einer unterlagerten Transportschicht bekannt sein, wie lang die empfangene Nachricht ist, so ist die Prüfung negativ, wenn die empfangene Nachricht nicht die Länge der erwarteten Nutzdatenlänge plus 6 Bytes entspricht und bricht mit `return FALSE` ab. Andernfalls wird mit der Prüfung fortgefahren. *(Dieser Schritt ist optional)*
2. Falls Duplicate.mValid==`TRUE` und Duplicate.mCRC1==CRC1 und Duplicate.mNextSeqNo-Lsb==SeqNoLsb, dann endet der Ablauf hier mit `return FALSE`.
3. Falls Connection-ID nicht der voreingestellten Connection-ID entspricht, ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
4. Ist Event1!=event, so ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
5. Ist Seq-No-LSB ungleich (0x1&NextSeqNo), so ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.

---

[210] Die Länge der Nutzdaten ist eine Größe die fest im Gerät verankert ist oder durch seine Parametrierung bestimmt wird. Wird die Nutzdatenlänge der Prozessdaten (SI_INPUT/SI_OUTPUT) durch mConfig bestimmt, so ist eine Länge der Nutzdaten der Nachrichten des Verbindungsaufbaus zu vereinbaren.

[211] Für Ausgabe-Slaves ist es z.B. unter Umständen möglich, dass keine Prozessdaten vom Slave an den Master geschickt werden, jedoch zu Open-Verarbeitung mindestens 1 Nutzdatenbyte definiert werden muss.

6. Mit dem Startwert LastSlavePreset und der erwarteten NextSeqNo wird CRC1 berechnet. Stimmt dieser mit dem Wert für CRC1 in der Nachricht nicht überein, so ist die Prüfung negativ und bricht mit `return FALSE` ab.
7. Sichere die Informationen zur Duplikat-Erkennung. Duplicate.mCRC1 = CRC1; Duplicate.mNextSeqNoLsb=SeqNoLsb; Duplicate.mValid=`TRUE`;
8. Ende mit `return TRUE`.

### Prüfungen des TSP2 Headers

`bool` **`CheckTSP2Header`**`(`**`event`**`)`

1. Sollte von einer unterlagerten Transportschicht bekannt sein, wie lang die empfangene Nachricht ist, so ist die Prüfung negativ, wenn die empfangene Nachricht nicht die Länge der erwarteten Nutzdatenlänge plus 12 Bytes entspricht und bricht mit `return FALSE` ab. Andernfalls wird mit der Prüfung fortgefahren. *(Dieser Schritt ist optional)*
2. Falls Duplicate.mValid==`TRUE` und Duplicate.mCRC2==CRC2 und Duplicate.mCRC3==CRC3 und Duplicate.mNextSeqNoLsb==SeqNoLsb, dann endet der Ablauf hier mit `return FALSE`.
3. Falls Connection-ID (high/low) nicht der für die Protokollinstanz voreingestellten entspricht, ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
4. Falls die reserved Bits nicht 0 sind, ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
5. Ist Event1!=TSP2-Format oder Event2!=event, so ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
6. Ist Seq-No-LSB ungleich (0x1&NextSeqNo), so ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
7. Mit dem Startwert LastSlavePreset und der erwarteten NextSeqNo werden CRC2 und CRC3 berechnet. Stimmen diese mit den Werten für CRC2 und CRC3 in der Nachricht nicht überein, so ist die Prüfung negativ und bricht mit `return FALSE` ab
8. Sichere die Informationen zur Duplikat-Erkennung. Duplicate.mCRC2 = CRC2; Duplicate.mCRC3 = CRC3; Duplicate.mNextSeqNoLsb = SeqNoLsb; Duplicate.mValid = `TRUE`;
9. Ende mit `return TRUE`.

Die Transitionen der Slave-Statemaschine werden nach dem Muster „**Auslöser**“, Ereignis, dass die Transition auslöst und „**Aktivität**“, Liste der Maßnahmen, die darauf hin ausgeführt werden, definiert.

### Transition: T0 – Protokollerzeugung

**Auslöser:**

Die Erzeugung der Protokollinstanz, Freigabe der Protokollverarbeitung durch die Applikation des Slaves.

**Aktivität:**

1. Initialisierung von LastMasterPreset mit `INITIAL_MASTER_PRESET`
2. Initialisierung von NextSeqNo mit `INITIAL_SEQUENCE_NO`
3. Initialisierung von LastSlavePreset mit `INITIAL_SLAVE_PRESET`
4. Initialisierung von Duplicate.mValid mit `FALSE`

### Transition: T1 OpenInd-Verarbeitung

**Auslöser:**

Der Empfang einer Nachricht durch Protokollinstanz

**Aktivität:**

Es wird geprüft, ob eine gültige Open-Indication (Fragment) für diese Protokollinstanz empfangen wurde

Falls TSP1-Betrieb:

> Falls `CheckTSP1Header(OPEN_INDICATION) == FALSE`, so wird die Nachricht verworfen.
> Falls der ALIVE_TIMER läuft und das Ergebnis von `CheckTSP1Header()` war `FALSE`, so endet der Ablauf hier.
> Falls der ALIVE_TIMER nicht läuft und das Ergebnis von `CheckTSP1Header()` war `FALSE`, so wird T10 ausgelöst und der Ablauf endet hier.
> Andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Falls TSP2-Betrieb:

> Falls `CheckTSP2Header(OPEN_INDICATION) == FALSE`, so wird die Nachricht verworfen.
> Falls der ALIVE_TIMER läuft und das Ergebnis von `CheckTSP2Header()` war `FALSE`, so endet der Ablauf hier.
> Falls der ALIVE_TIMER nicht läuft und das Ergebnis von `CheckTSP2Header()` war `FALSE`, so wird T10 ausgelöst und der Ablauf endet hier.
> Andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Gemeinsam für TSP1 und TSP2:

1. Falls dies die erste Nachricht der Verbindungsaufnahme ist und NextSeqNo ist nicht gleich `INITIAL_SEQUENCE_NO`, so wird T10 ausgelöst und der Ablauf endet hier.
   Falls dies die erste Nachricht der Verbindungsaufnahme ist und NextSeqNo gleich `INITIAL_SEQUENCE_NO` ist, wird das erste Nutzdatenbyte in OpenTMO gespeichert und mit dem nächsten Schritt fortgefahren.
2. Der ALIVE_TIMER wird mit OpenTMO gestartet
   *Anmerkung: Dies geschieht schon hier, weil der ALIVE_TIMER möglichst zeitnah zum Empfang der Nachricht gestartet werden soll. Dies geschieht, obwohl die restlichen Daten der Nachricht noch logisch ungültig sein könnten. Die Nachricht selber ist jedoch nicht verfälscht.*
3. Die Nutzdaten werden an die Daten für das **`OpenIndData`** angehängt. Dabei wird natürlich auf den Überlauf der `OpenIndData` geachtet.
4. Ist das OkBit = 0 (Nutzdaten stellen ein Fragment dar) und die Daten für `OpenIndData` sind noch nicht vollständig, so wird `AckOpenIndDataFragment()` aufgerufen und NextSeqNo wird um 1 erhöht und der Ablauf endet hier.
5. Ist das OkBit = 0 (Nutzdaten stellen ein Fragment dar) und die Daten für `OpenIndData` sind schon vollständig[212], so wird `FillOpenRespData(OPEN_IND_OVERFLOW, 0)` aufgerufen, T3 ausgelöst und der Ablauf endet hier.

[212] Durch die feste Nutzdatenlänge der Nachricht, basierend auf den realen Prozessdaten, können im letzten Fragment am Ende ungenutzte Bytes enthalten sein, die jedoch 0xff sein müssen.

6. Ist das OkBit = 1 (Fragment-Ende) und die Daten für `OpenIndData` sind noch nicht vollständig, so wird `FillOpenRespData(OPEN_IND_UNDERFLOW, 0)` aufgerufen, T3 ausgelöst und der Ablauf endet hier
7. Ist das OkBit = 1 (Fragment-Ende) und die Daten für `OpenIndData` sind vollständig, so wird mit der Prüfung des `OpenIndData` fortgefahren.
8. Falls `OpenIndData.mConnectionID` nicht der für die Protokollinstanz voreingestellten Connection-ID entspricht, wird `OpenIndData` verworfen und `FillOpenRespData(OPEN_IND_ABORT, 0)` aufgerufen, T3 ausgelöst und der Ablauf endet hier.
9. Die Länge des `OpenIndData`'s wird aus sizeof(`OpenIndData`) und dem Wert in mConfigSize ermittelt. Ist die berechnetet Länge von `OpenIndData` incl. mConfig größer als die maximal zulässige Länge von 65.535 Bytes, wird das `OpenIndData` verworfen und `FillOpenRespData(OPEN_IND_ABORT, 0)` aufgerufen und T3 ausgelöst und der Ablauf endet hier[213].
10. Prüfe, ob die Daten in `OpenIndData` (ohne mConfig) zum `OpenIndData.mOpenCrc` passen. Ist die Prüfung erfolgreich, so wird fortgefahren, andernfalls wird `FillOpenRespData(OPEN_IND_ABORT, 0)` aufgerufen, T3 ausgelöst und der Ablauf endet hier.
11. Unterstützt der Slave die vom Master gewünschte `OpenIndData.mProtoVersion` nicht, so wird das `OpenIndData` verworfen, `FillOpenRespData(PROTO_VERSION_NOT_SUPPORTED, OpenIndData.mMasterPreset)` aufgerufen und T3 ausgelöst. Andernfalls merkt sich der Slave in MasterProtoVersion=`OpenIndData.mProtoVersion` und fährt dann mit der Prüfung fort.
    *Anmerkung: Ab hier ist `OpenIndData.mMasterPreset` gültig und wird in OpenRespData übernommen. Zur Zeit wird nur die `mProtoVersion==1` unterstützt.*
12. Ist der Slave ohne Konfiguration zu betreiben, und die `OpenIndData.mConfigSize` ist ungleich 0, so wird das `OpenIndData` verworfen, `FillOpenRespData(CONFIG_NOT_SUPPORTED, OpenIndData.mMasterPreset)` aufgerufen und T3 ausgelöst. Andernfalls wird mit der Prüfung fortgefahren.
13. Wird der Slave mit einer Konfiguration betrieben, die nicht vom Master kommt oder wird er ohne Konfiguration betrieben und stimmt die `OpenIndData.mSlaveConfigSignature`[214] nicht mit der Signatur der Slave-Konfiguration überein, wird das `OpenIndData` verworfen, `FillOpenRespData(CONFIG_MISSMATCH, OpenIndData.mMasterPreset)` aufgerufen und T3 ausgelöst. Andernfalls wird mit der Prüfung fortgefahren.
14. Kann der Slave mit einer Konfiguration vom Master betrieben werden, und ist die auf Grund von `OpenIndData.mConfig[]` berechnete Signatur ungleich `OpenIndData.mSlaveConfigSignature`, so wird das `OpenIndData` verworfen, `FillOpenRespData(CONFIG_ABORT, OpenIndData.mMasterPreset)` aufgerufen, T3 ausgelöst und der Ablauf endet hier.
15. Kann der Slave mit einer Konfiguration vom Master betrieben werden, und ist `OpenIndData.mConfigSize==0`, aber `OpenIndData.mSlaveConfigSignature` ungleich der vorhandenen Config-Signatur im Slave, so wird das `OpenIndData` verworfen , `FillOpenRespData(CONFIG_DIFFER, OpenIndData.mMasterPreset)` aufgerufen, T3 ausgelöst und der Ablauf endet hier.
16. Kann der Slave mit einer Konfiguration vom Master betrieben werden, und ist die auf Grund von `OpenIndData.mConfig[]` berechnete Signatur gleich `OpenIndData.mSlaveConfig-`

[213] Hier sollte eigentlich schon OPEN_IND_OVERFLOW ausgelöst worden sein.

[214] Falls bei der Signaturberechung der Konfiguration 0 herauskommt, so muss der Wert 1 verwendet werden.

Signature, aber der Inhalt von mConfig[] ist für die Applikation des Slaves nicht gültig, so wird das `OpenIndData` verworfen, `FillOpenRespData(CONFIG_ABORT, OpenIndData.mMasterPreset)` aufgerufen, T3 ausgelöst und der Ablauf endet hier.

17. Andernfalls ist die Prüfung damit beendet, es wird `FillOpenRespData(ACCEPTED, OpenIndData.mMasterPreset)` aufgerufen und T5 ausgelöst.

### Transition: T2 – Open Abort

**Auslöser:**

Negatives Prüfergebnis einer Nachricht durch Protokollinstanz, Abbruch.

**Aktivität:**

1. Initialisierung von LastMasterPreset mit `INITIAL_MASTER_PRESET`
2. Initialisierung von NextSeqNo mit `INITIAL_SEQUENCE_NO`
3. Initialisierung von LastSlavePreset mit `INITIAL_SLAVE_PRESET`
4. Evtl. laufenden ALIVE_TIMER stoppen
5. Initialisierung von Duplicate.mValid mit `FALSE`
6. Sichere Daten für SI_OUTPUT einstellen und SI_OK_BIT_IN=0, sofern nicht schon erfolgt.

### Transition: T3 – OpenRespReject-Fragment-Verarbeitung

**Auslöser:**

Eigentrigger

**Aktivität:**

1. Rufe `SendOpenRespDataFragment(last)` auf mit last=FALSE, falls bei diesem Aufruf noch nicht alle Bytes der OpenRespData Struktur übertragen werden können, und sonst mit TRUE.
2. Falls last==`TRUE`, löse T2 aus, sonst wird NextSeqNo um 1 erhöht.

### Transition: T4 – OpenRespReject-Ack-Verarbeitung

**Auslöser:**

Eine Nachricht wurde empfangen.

**Aktivität:**

Es wird erwartet, dass die Nachricht ein Ack eines `OpenRespData` Fragments ist.

Falls TSP1-Betrieb:

Falls `CheckTSP1Header(OPEN_INDICATION) == FALSE`, so wird die Nachricht verworfen und der Ablauf endet hier. Andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Falls TSP2-Betrieb:

Falls `CheckTSP2Header(OPEN_INDICATION) == FALSE`, so wird die Nachricht verworfen und der Ablauf endet hier. Andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Gemeinsam für TSP1 und TSP2:

1. Der ALIVE_TIMER wird mit OpenTMO gestartet.
2. Prüfe, ob die Nutzdaten alle OPEN_RESP_FRAG_ACK sind. Falls nein, löse T2 aus, sonst fahre fort.

3. Rufe `SendOpenRespDataFragment(last)` auf mit last=`FALSE`, falls bei diesem Aufruf noch nicht alle Bytes der `OpenRespData` Struktur übertragen werden können, sonst mit `TRUE`.
4. Falls last==`TRUE`, löse T2 aus, sonst wird NextSeqNo um 1 erhöht.

### Transition: T5 – OpenRespAccept-Fragment-Verarbeitung

**Auslöser:**

Eigentrigger

**Aktivität:**

1. Rufe `SendOpenRespDataFragment(last)` auf mit last=`FALSE`, falls bei diesem Aufruf noch nicht alle Bytes der **`OpenRespData`** Struktur übertragen werden können, und sonst mit `TRUE`.
2. Falls last==`TRUE`, löse T7[215] aus und ende hier, sonst weiter mit NextSeqNo um 1 erhöhen.

### Transition: T6 – OpenRespAccept-Ack-Verarbeitung

**Auslöser:**

Eine Nachricht wurde empfangen.

**Aktivität:**

Es wird erwartet, dass die Nachricht eine Bestätigung eines `OpenRespData` Fragments ist.

Falls TSP1-Betrieb:

Falls `CheckTSP1Header(OPEN_INDICATION) == FALSE`, so wird die Nachricht verworfen und der Ablauf endet hier, andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Falls TSP2-Betrieb:

Falls `CheckTSP2Header(OPEN_INDICATION) == FALSE`, so wird die Nachricht verworfen und der Ablauf endet hier, andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Gemeinsam für TSP1 und TSP2:

1. Der ALIVE_TIMER wird mit OpenTMO gestartet.
2. Prüfe, ob die Nutzdaten alle OPEN_RESP_FRAG_ACK sind. Falls nein, löse T2 aus, sonst fahre fort.
3. Rufe `SendOpenRespDataFragment(last)` auf mit last=`FALSE`, falls bei diesem Aufruf noch nicht alle Bytes der `OpenRespData` Struktur übertragen werden können, sonst mit `TRUE`.
4. Falls last==`TRUE`, löse T7[216] aus, sonst wird NextSeqNo um 1 erhöht.

### Transition: T7 – Open-Verarbeitung-Beendet

**Auslöser:**

Eigentrigger.

**Aktivitäten:**

1. ActiveWDT= **`OpenIndData`**.mWDT
2. Starte ALIVE_TIMER mit ActiveWDT

---

[215] Die Transition T7 ist zeitlich unmittelbar nach der Verarbeitung des Ablaufs zur Transition T5 auszuführen.

[216] Die Transition T7 ist zeitlich unmittelbar nach der Verarbeitung des Ablaufs zur Transition T6 auszuführen.

3. LastMasterPreset= **`OpenIndData.`**mMasterPreset
4. LastSlavePreset= **`OpenRespData.`**mSlavePreset
5. NextSeqNo= **`OpenIndData.`**mMasterPreset.

### Transition: T8 – Empfangene Prozessdaten

**Auslöser:**

Empfang einer Nachricht.

**Aktivität:**

Von der empfangenen Nachricht wird erwartet, dass es sich um eine Datennachricht handelt. Dies wird nun geprüft:

Falls TSP1-Betrieb:

> Falls `CheckTSP1Header(OUTPUT_DATA_INDICATION) == FALSE`, so wird die Nachricht verworfen und die Aktivität endet hier, andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Falls TSP2-Betrieb:

> Falls `CheckTSP2Header(OUTPUT_DATA_INDICATION) == FALSE`, so wird die Nachricht verworfen und die Aktivität endet hier, andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Gemeinsam für TSP1 und TSP2:

1. Der ALIVE_TIMER wird mit ActiveWDT neu gestartet.
2. Das OkBit aus dem Header wird in SI_OK_BIT_IN gespeichert.
3. Die Nutzdaten für SI_OUTPUT werden, sofern vorhanden, aus der Nachricht übernommen.
4. Es wird T9 ausgelöst.[217]

### Transition: T9 – Sende Prozessdaten

**Auslöser:**

Positive Prüfung einer Datennachricht.

**Aktivität:**

1. Rufe `SendData(`SI_OK_BIT_OUT`)` auf.
2. Die NextSeqNo wird auf NextSeqNo+1 gesetzt.

Der ALIVE_TIMER läuft schon mit dem Empfang der letzten gültigen Nachricht und darf hier nicht neu gestartet werden.

### Transition: T10

**Auslöser:**

Der ALIVE_TIMER ist abgelaufen.

**Aktivität:**

1. Initialisierung von LastMasterPreset mit `INITIAL_MASTER_PRESET`
2. Initialisierung von NextSeqNo mit `INITIAL_SEQUENCE_NO`

[217] Sollte dies nicht zur Verarbeitung innerhalb des Slaves passen, so kann die Transition T9 bis zum passenden Zeitpunkt verschoben werden. Z.B. ist es sinnvoll, wenn Eingabe-Slaves vor der Transition T9 ihre Eingangsdaten für SI_INPUT ermitteln.

3. Initialisierung von LastSlavePreset mit `INITIAL_SLAVE_PRESET`
4. Initialisierung von Duplicate.mValid mit `FALSE`
5. Für die sicherheitsgerichteten SI_OUTPUT Daten werden die Initialwerte und SI_OK_BIT_IN auf 0 eingestellt.

## 5.2 TSP Master

Zu jeder TSP Kommunikationsverbindung müssen im Master folgenden Informationen vorhanden sein, bevor das TSP Protokoll den Datenaustausch vornehmen kann:

- Welches Nachrichtenformat, TSP1 oder TSP2, für die Verbindung zur Anwendung kommen soll.
- Connection-ID der Verbindung
- Konfigurierte Überwachungszeit der Verbindung (ActiveWDT)
- Konfigurierte Überwachungszeit der Verbindung während Verbindungsaufnahme (OpenT-MO)
- Nachrichtenlänge für SI_INPUT
- Nachrichtenlänge für SI_OUTPUT
- Erwartungshaltung des Slaves bzgl. der Slave-Konfiguration mSlaveConfigSignatur/mConfig und gegebenenfalls auch die mConfig Daten, falls USE_SLAVE_CONFIG==TRUE.

Die Informationen müssen außerhalb des TSP im Master sicherheitsgerichtet eingestellt und gespeichert werden.

Nachfolgend werden die Master Datenobjekte definiert:

- ALIVE_TIMER — ein Timer zur Alive-Überwachung der Verbindung
  Er muss zur Messung der mWDT und mOpenTMO geeignet sein. Er überwacht die Abstände der empfangenen RESPONSEs des Slaves bzw. bei der Verbindungsaufnahme, auch zum Teil die Antwortzeiten des Slaves, sowie den Wiederanlauf.
- ActiveWDT — Startwert für ALIVE_TIMER während Prozessdatenaustausch
- OpenTMO — Startwert für ALIVE_TIMER während Verbindungsaufnahme
- OpenIndData — siehe oben
- OpenRespData — siehe oben
- LastSlavePreset — Startwert für CRC Berechnung empfangener Slave-Nachrichten
- LastMasterPreset — Schlüssel des Masters als Startwert für Sequence-Number und als Startwert für die CRC Berechnung von Master-Nachrichten
- NextSeqNo — Wert der Sequence-Number, die der Slave in der nächsten Nachrichten vom Master erwartet oder aus der Indication an den Slave, zu der beim Master noch keine Response verarbeitet wurde
- CONFIG_TO_SEND — `TRUE`-> Slave-Konfiguration soll beim nächsten Open-Ind. mit gesendet werden. `FALSE` -> es wird nur die mSlaveConfigSignature gesendet
- USE_SLAVE_CONFIG — `TRUE`-> der Slave soll durch den Master konfiguriert werden, `FALSE` -> nicht
- SLAVE_RESPONSE — Speicher für `mResult` aus **`OpenRespData`**.
- SI_INPUT — sicherheitsgerichtete Daten vom Slave an den Master (optional)
- SI_OUTPUT — sicherheitsgerichtete Daten vom Master an den Slave (optional)

- SI_OK_BIT_IN — nimmt Wert des OkBits aus TSP-Header an und kann von Applikation genutzt werden. Der Wert der Applikation des Slaves ist nur im Master-Zustand VALID_DATA verfügbar.
- SI_OK_BIT_OUT — wie SI_OK_BIT_IN, wird zum Versenden an den Slave genutzt und muss von der Applikation belegt werden. Der Wert wird nur bei der Prozessdatenübertragung benutzt und steht dem Slave nur im Slave Zustand VALID_DATA zur Verfügung.
- Duplicate — wie beim TSP-Slave

**Master State-Maschine**

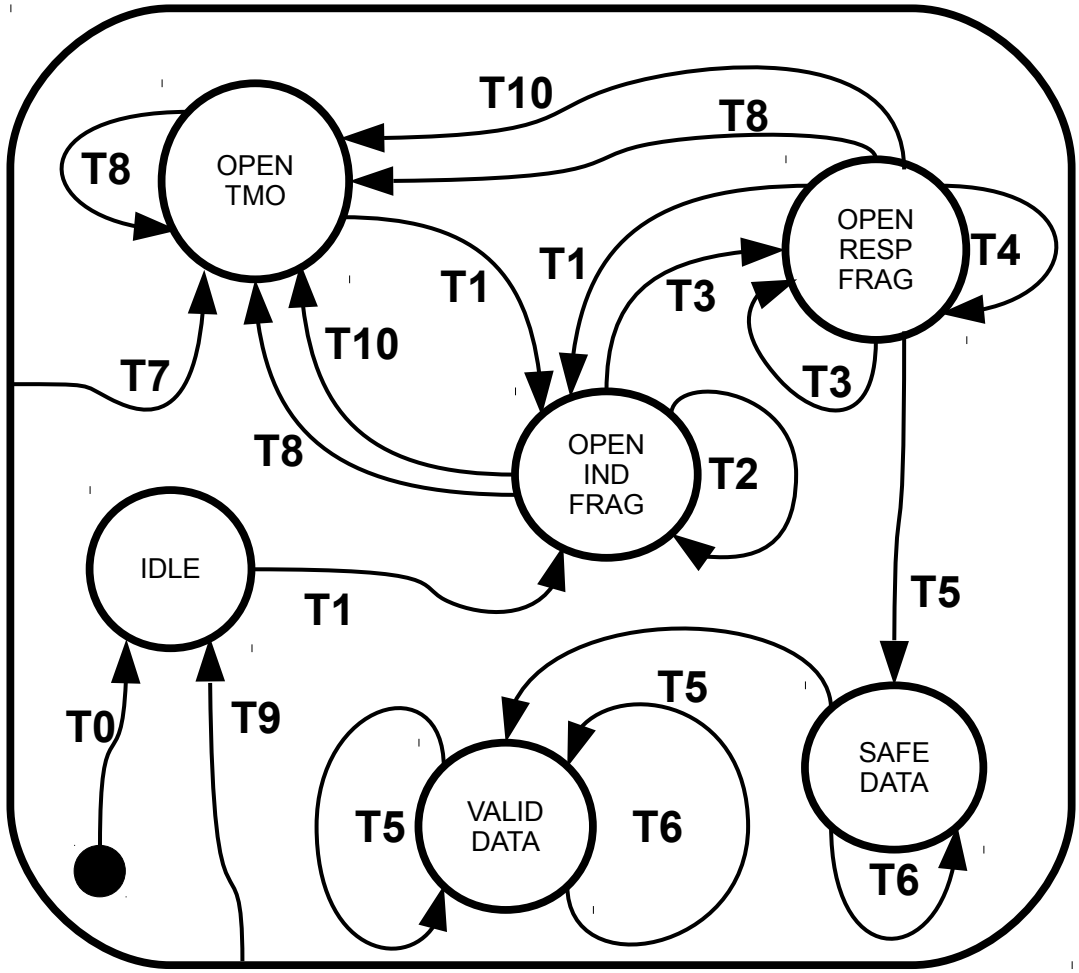


*Abbildung 5.7: TSP Master Zustandsdiagramm*

Im Initialzustand ist das Protokoll nicht bereit eine Kommunikationsverbindung aufzubauen und zu betreiben. Die der Kommunikationsverbindung zugeordneten sicheren SI_INPUT Daten haben ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft nicht.

**Zustand: IDLE**

Das Protokoll ist nicht bereit eine Kommunikationsverbindung aufzubauen, die der Kommunikationsverbindung zugeordneten sicheren SI_INPUT Daten haben ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft nicht.

### Zustand: OPEN_IND_FRAG

Das Protokoll ist beim Verbindungsaufbau und versendet **`OpenIndData`** Fragmente. Die der Kommunikationsverbindung zugeordneten sicheren SI_INPUT Daten haben ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft mit OpenTMO.

### Zustand: OPEN_RESP_FRAG

Das Protokoll ist beim Verbindungsaufbau und empfängt **`OpenRespData`** Fragmente. Die der Kommunikationsverbindung zugeordneten sicheren SI_INPUT Daten haben ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft mit OpenTMO.

### Zustand: SAFE_DATA

Das Protokoll ist geöffnet, es liegen jedoch noch keine SI_INPUT Daten des Slaves vor. Daher haben die der Kommunikationsverbindung zugeordneten sicheren SI_INPUT Daten ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0. Der Überwachungs-Timer ALIVE_TIMER läuft mit ActiveWDT.

### Zustand: VALID_DATA

Das Protokoll ist geöffnet, es liegen gültige SI_INPUT Daten des Slaves vor und SI_OK_BIT_IN hat den Wert aus dem OkBit der letzten gültigen Nachricht. Der Überwachungs-Timer ALIVE_TIMER läuft mit ActiveWDT.

### Zustand: OPEN_TMO

Das Protokoll hatte den Verbindungsaufbau abgebrochen und wartet auf den Anstoß für einen neuen Verbindungsaufbau. Die der Kommunikationsverbindung zugeordneten sicheren SI_INPUT Daten haben ihren sicherheitsgerichteten Initialwert und SI_OK_BIT_IN ist 0.

### Hilfsfunktionen für Transitionen

Zur Vereinfachung der späteren Definition der Transitionen der Master Statemaschine, folgen hier einige Hilfsfunktionen.

### Bestätige OpenRespData Fragment

**`AckOpenRespDataFragment()`**

Falls TSP1 Betrieb:

- Connection-ID=voreingestellte Connection-ID
- Event1= OPEN_INDICATION

Falls TSP2 Betrieb:

- Connection-ID-High/Low = voreingestellte Connection-ID
- reserved-Bits=0
- Event1=TSP2-Format
- Event2= OPEN_INDICATION

Gemeinsam:

- OkBit=0
- Seq-No-LSB=(NextSeqNo & 0x1)

- Fülle alle Nutzdatenbytes[218] mit OPEN_RESP_FRAG_ACK.
- CRC Berechnung mit Startwert LastMasterPreset und die Sequence-Number=NextSeqNo ausführen und in CRC1 respektive CRC2/3 eintragen.

Schlussendlich wird die Nachricht an den zur Connection-ID passenden Slave versendet.

### Fülle OpenIndData

**`FillOpenIndData()`**

- `OpenIndData.mOpenTMO=` OpenTMO
- `OpenIndData.mMasterPreset=`**`GeneratePreset();`** // mMasterPreset wird noch nicht belegt
- `OpenIndData.mSlaveConfigSignature=` Wert der Signatur, falls vorhanden oder 0
- `OpenIndData.mConnectionId=` Wert der eigenen Connection-ID
- `OpenIndData.mWDT= ActiveWDT`
- `OpenIndData.mProtoVersion=` Wert der zu nutzenden Version, z.Z. immer 1

Falls USE_SLAVE_CONFIG und CONFIG_TO_SEND

- `OpenIndData.mConfigSize=` die Länge der parametrierten Konfiguration für mConfig
- `OpenIndData.mConfig`[] = falls mConfigSize > 0, die parametrierten Konfigurationsdaten
- CONFIG_TO_SEND= `FALSE`;

Falls !USE_SLAVE_CONFIG oder !CONFIG_TO_SEND

- `OpenIndData.mConfigSize=` 0;
- `OpenIndData.mConfig`[] = entfällt, Länge 0

Gemeinsam:

- `OpenIndData.mOpenCrc=` Wert des CRCs für **OpenIndData**

### Sende OpenIndData Fragment

**`SendOpenIndDataFragment(last)`**

Falls TSP1 Betrieb:

- Connection-ID=voreingestellte Connection-ID
- Event1= OPEN_INDICATION

Falls TSP2 Betrieb:

- Connection-ID-High/Low = voreingestellte Connection-ID
- reserved-Bits=0
- Event1=TSP2-Format
- Event2= OPEN_INDICATION

Gemeinsam:

- OkBit=last
- Seq-No-LSB=(NextSeqNo & 0x1)
- Fülle die Nutzdatenbytes[218] mit dem nächsten Teil der `OpenIndData` Daten und falls ein Rest in den Nutzdaten verbleibt, fülle diesen mit 0xFF (last muss dann `TRUE` sein)

[218] Die Länge der Nutzdaten ist eine Größe die fest im Gerät verankert ist oder durch seine Parametrierung bestimmt wird. Wird die Nutzdatenlänge der Prozessdaten (SI_INPUT/SI_OUTPUT) durch mConfig bestimmt, so ist eine Länge der Nutzdaten der Nachrichten des Verbindungsaufbaus zu vereinbaren.

- CRC Berechnung mit dem Startwert LastSlavePreset und der Sequence-Number=NextSeqNo ausführen und in CRC1 respektive CRC2/CRC3 eintragen.

Schlussendlich wird die Nachricht an den zur Connection-ID passenden Slave versendet.

### Sende Prozessdaten

**`SendData`**`(`**`okBit`**`)`

Falls TSP1 Betrieb:

- Connection-ID=voreingestellte Connection-ID
- Event1= OUTPUT_DATA_INDICATION

Falls TSP2 Betrieb:

- Connection-ID-High/Low = voreingestellte Connection-ID
- reserved-Bits=0
- Event1=TSP2-Format
- Event2= OUTPUT_DATA_INDICATION

Gemeinsam:

- OkBit=`okBit`
- Seq-No-LSB=(NextSeqNo & 0x1)
- Fülle die Nutzdatenbytes mit den Prozessdaten aus SI_OUTPUT. Falls keine Prozessdaten geschickt werden, wird das Nutzdatenbyte mit 0 gefüllt.
- CRC Berechnung mit dem Startwert LastSlavePreset und der Sequence-Number=NextSeqNo ausführen und in CRC1 respektive CRC2/3 eintragen.

Schlussendlich wird die Nachricht an den zur Connection-ID passenden Slave versendet.

### Prüfung des TSP1 Headers

`bool` **`CheckTSP1Header`**`(`**`event`**`)`

1. Sollte von einer unterlagerten Transportschicht bekannt sein, wie lang die empfangene Nachricht ist, so ist die Prüfung negativ, wenn die empfangene Nachricht nicht die Länge der erwarteten Nutzdatenlänge plus 6 Bytes entspricht und bricht mit `return FALSE` ab. Andernfalls wird mit der Prüfung fortgefahren. *(Dieser Schritt ist optional)*
2. Falls Duplicate.mValid==`TRUE` und Duplicate.mCRC1==CRC1 und Duplicate.mNextSeqNoLsb==SeqNoLsb, dann endet der Ablauf hier mit `return FALSE`.
3. Falls Connection-ID nicht der voreingestellten Connection-ID entspricht, ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
4. Ist Event1!=event, so ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
5. Ist Seq-No-LSB ungleich (0x1&NextSeqNo), so ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
6. Mit dem Startwert LastMasterPreset und der erwarteten NextSeqNo wird CRC1 berechnet. Stimmt dieser mit dem Wert für CRC1 in der Nachricht nicht überein, so ist die Prüfung negativ und bricht mit `return FALSE` ab
7. Sichere die Informationen zur Duplikat-Erkennung. Duplicate.mCRC1 = CRC1; Duplicate.mNextSeqNoLsb=SeqNoLsb; Duplicate.mValid=`TRUE`;
8. Ende mit `return TRUE`.

### Prüfung des TSP2 Headers

`bool` **`CheckTSP2Header`**`(`**`event`**`)`

1. Sollte von einer unterlagerten Transportschicht bekannt sein, wie lang die empfangene Nachricht ist, so ist die Prüfung negativ, wenn die empfangene Nachricht nicht die Länge der erwarteten Nutzdatenlänge plus 12 Bytes entspricht und bricht mit `return FALSE` ab. Andernfalls wird mit der Prüfung fortgefahren. *(Dieser Schritt ist optional)*
2. Falls Duplicate.mValid==`TRUE`, Duplicate.mCRC2==CRC2, Duplicate.mCRC3==CRC3 und Duplicate.mNextSeqNoLsb==SeqNoLsb, dann endet der Ablauf hier mit `return FALSE`. Es wurde ein Duplikat erkannt.
3. Falls Connection-ID (high/low) nicht der für die Protokollinstanz voreingestellten entspricht, ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
4. Falls die reserved Bits nicht 0 sind, ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
5. Ist Event1!=TSP2-Format oder Event2!=event, so ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
6. Ist Seq-No-LSB ungleich (0x1&NextSeqNo), so ist die Prüfung negativ und bricht mit `return FALSE` ab, andernfalls wird mit der Prüfung fortgefahren.
7. Mit dem Startwert LastMasterPreset und der erwarteten NextSeqNo werden CRC2 und CRC3 berechnet. Stimmen diese mit den Werten für CRC2 oder CRC3 in der Nachricht nicht überein, so ist die Prüfung negativ und bricht mit `return FALSE` ab.
8. Sichere die Informationen zur Duplikat-Erkennung. Duplicate.mCRC2 = CRC2; Duplicate.mCRC3 = CRC3; Duplicate.mNextSeqNoLsb = SeqNoLsb; Duplicate.mValid= `TRUE`;
9. Ende mit `return TRUE`.

Die Transitionen der Slave-Statemaschine werden nach dem Muster „**Auslöser**“, Ereignis, dass die Transition auslöst und „**Aktivität**“, Liste der Maßnahmen, die darauf hin ausgeführt werden, definiert.

### Transition: T0 – Protokollerzeugung

**Auslöser:**

Die Erzeugung der Protokollinstanz

**Aktivität:**

1. Initialisierung von USE_SLAVE_CONFIG, ActiveWDT und OpenTMO mit den Werten der Parametrierung.
2. Initialisierung von SI_INPUT mit sicheren Initialwerten
3. Initialisierung von SI_OK_BIT_IN mit 0
4. Initialisierung von SLAVE_RESPONSE mit `Empty`
5. Initialisierung von Duplicate.mValid mit `FALSE`
6. CONFIG_TO_SEND=`FALSE`; // zuerst ohne Slave-Konfiguration Verbindung öffnen

### Transition: T1 – Protokollaktivierung

**Auslöser:**

Aktivierung der Protokollinstanz durch die Applikation oder durch den ALIVE_TIMER, wenn er mit OPEN_TMO abläuft, mit dem Ziel eine Verbindung aufzubauen.

**Aktivität:**

1. Setze SLAVE_RESPONSE auf **`EMPTY`** als Vorbelegung
2. Initialisierung von LastMasterPreset mit `INITIAL_MASTER_PRESET`
3. Initialisierung von NextSeqNo mit `INITIAL_SEQUENCE_NO`
4. Initialisierung von LastSlavePreset mit `INITIAL_SLAVE_PRESET`
5. Initialisierung von Duplicate.mValid mit `FALSE`
6. Rufe `FillOpenIndData()` zum Füllen der **`OpenIndData`** Struktur auf
7. Rufe `SendOpenIndDataFragment(last)` auf mit last=`FALSE`, falls bei diesem Aufruf noch nicht alle Bytes der **OpenIndData** Struktur übertragen werden können, und sonst mit `TRUE`
8. Der ALIVE_TIMER wird mit OpenTMO gestartet
9. Falls last==`TRUE`, triggere mit T3 weiter

### Transition: T2 – OpenInd-Ack-Verarbeitung

**Auslöser:**

Empfang einer Nachricht vom Slave.

**Aktivität:**

Falls TSP1-Betrieb:

Falls `CheckTSP1Header(RESPONSE) == FALSE`, so wird die Nachricht verworfen und T10 ausgelöst, andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Falls TSP2-Betrieb:

Falls `CheckTSP2Header(RESPONSE) == FALSE`, so wird die Nachricht verworfen und T10 ausgelöst, andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Gemeinsam für TSP1 und TSP2:

1. Starte ALIVE_TIMER mit OpenTMO
2. NextSeqNo= NextSeqNo + 1
3. Speichere das erste Nutzdatenbyte in SLAVE_RESPONSE.
4. Falls das erste Nutzdatenbyte `OPEN_IND_FRAG_ACK` ist, rufe `SendOpenIndDataFragment(last)` mit last=`FALSE` auf, falls bei diesem Aufruf noch nicht alle Bytes der OpenIndData Struktur übertragen werden können, sonst mit `TRUE`. Falls last=`TRUE`, triggere mit T3 weiter und der Ablauf endet dann hier. Falls last=`FALSE`, fahre mit den nächsten Aktionen fort.
5. Falls das erste Nutzdatenbyte ungleich `OPEN_IND_FRAG_ACK` ist und OkBit==`TRUE` ist; breche das weitere Senden von `OpenIndData`-Fragmenten ab; übernehme Nutzdaten unter Beachtung der Länge nach `OpenRespData;` triggere mit T10 weiter. Der Ablauf endet dann hier.
6. Falls das erste Nutzdatenbyte ungleich `OPEN_IND_FRAG_ACK` ist und OkBit==`FALSE` ist, breche das weitere Senden von `OpenIndData`-Fragmenten ab, übernehme Nutzdaten unter

Beachtung der Länge nach `OpenRespData`, triggere mit T3 weiter. Der Ablauf endet dann hier.

### Transition: T3 – OpenResponse Prüfen

**Auslöser:**

Eigentrigger.

**Aktivität:**

1. Falls `OpenRespData` noch nicht vollständig empfangen wurde, endet der Ablauf hier. Ansonsten geht der Ablauf mit der Prüfung von `OpenRespData` weiter.
2. Prüfe, ob die Daten in `OpenRespData` zum `OpenRespData.mOpenCrc` passen. Ist die Prüfung erfolgreich, so wird fortgefahren, andernfalls wird T10 ausgelöst.
3. Ist `OpenRespData.mProtoVersion` < als die vom Master gewünschte, so wird die Nachricht verworfen und T10 ausgelöst, andernfalls fährt man fort.
4. Falls der `OpenRespData.mMasterPreset` != `OpenIndData.mMasterPreset` ist, wird die Nachricht verworfen und T10 ausgelöst, andernfalls wird mit der Prüfung fortgefahren.
5. Ist `OpenRespData.mResult` == `CONFIG_DIFFER` und es wurden in der letzten Open-Indication keine mConfig Daten verschickt und es sind mConfig Daten vorhanden, so wird nun `CONFIG_TO_SEND=TRUE` gesetzt, T1 getriggert und der Ablauf endet hier.
6. Ist `OpenRespData.mResult` != `ACCEPTED`, wird die Nachricht verworfen und T10 ausgelöst. Andernfalls wird mit der Prüfung fortgefahren.
7. Ist `OpenRespData.mSlaveConfigSignature` nicht gleich dem vom Master geschickten Wert, wird die Nachricht verworfen und T10 ausgelöst, andernfalls wird fortgefahren.
8. LastSlavePreset= `OpenRespData.mSlavePreset`
9. `LastMasterPreset=` `OpenIndData.mMasterPreset`
10. `NextSeqNo=LastMasterPreset`.
11. Starte ALIVE_TIMER mit ActiveWDT.
12. `CONFIG_TO_SEND=FALSE`; (beim nächsten Mal wieder ohne Konfigurationsdaten probieren)
13. Löse T5 aus

### Transition: T4 – OpenResponse-Verarbeitung

**Auslöser:**

Empfang einer Nachricht vom Slave.

**Aktivität:**

Falls TSP1-Betrieb:

Falls `CheckTSP1Header(RESPONSE) == FALSE`, so wird die Nachricht verworfen und T10 ausgelöst, andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Falls TSP2-Betrieb:

Falls `CheckTSP2Header(RESPONSE) == FALSE`, so wird die Nachricht verworfen und T10 ausgelöst, andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Gemeinsam für TSP1 und TSP2:

1. Starte ALIVE_TIMER mit `OPEN_TMO`

2. NextSeqNo wird um 1 erhöht.
3. Die nun verifizierten Nutzdaten werden an die Daten für das `OpenRespData` angehängt, wobei auf die maximale zulässige Länge zu achten ist.
4. Ist das OkBit == 0 (=Datenfragment) und die Daten für `OpenRespData` sind noch nicht vollständig, so wird `AckOpenRespDataFragment()` aufgerufen und dann endet der Ablauf hier.
5. Ist das OkBit == 0 (=Datenfragment) und die Daten für `OpenRespData` sind schon vollständig[219], so wird T10 ausgelöst und der Ablauf endet hier.
6. Ist das OkBit == 1 (Fragment-Ende) und die Daten für `OpenRespData` sind noch nicht vollständig, so wird T10 ausgelöst und der Ablauf endet hier.
7. Ist das OkBit == 1 (Fragment-Ende) und die Daten für `OpenIndData` sind vollständig, so wird mit der Prüfung des `OpenIndData` fortgefahren, indem T3 getriggert wird.

### Transition: T5 – Sende Prozessdaten

**Auslöser:**

Die Transition kommt zur Anwendung, wenn die entsprechende Phase des Masters zur Anwendung kommen soll (siehe Kapitel 5.4).

**Aktivität:**

1. Rufe `SendData(SI_OK_BIT_OUT)` auf.

Der ALIVE_TIMER läuft schon mit dem Empfang der letzten gültigen Nachricht und darf hier nicht neu gestartet werden.

### Transition: T6 – Empfange Prozessdaten

**Auslöser:**

Empfang einer Nachricht vom Slave.

**Aktivität:**

Falls TSP1-Betrieb:

Falls `CheckTSP1Header(RESPONSE) == FALSE`, so wird die Nachricht verworfen und der Ablauf endet hier. Andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Falls TSP2-Betrieb:

Falls `CheckTSP2Header(RESPONSE) == FALSE`, so wird die Nachricht verworfen und der Ablauf endet hier. Andernfalls wird mit dem gemeinsamen Teil fortgefahren.

Gemeinsam für TSP1 und TSP2:

1. Starte ALIVE_TIMER mit `ActiveWDT`
2. `NextSeqNo` wird um 1 erhöht
3. Das OkBit aus dem Header wird in SI_OK_BIT_IN gespeichert
4. Die Nutzdaten für SI_INPUT werden, sofern vorhanden, aus der Nachricht übernommen
5. Es wird T5 ausgelöst.[220]

---

[219] Durch die feste Nutzdatenlänge der Nachricht, basierend auf den realen Prozessdaten, können im letzten Fragment am Ende ungenutzte Bytes enthalten sein, die jedoch 0xff sein müssen.

[220] Sollte dies nicht zur Verarbeitung innerhalb des Masters passen, so kann die Transition T5 bis zum passenden Zeitpunkt verschoben werden. Z.B. ist es sinnvoll, wenn die Applikationsverarbeitung des Masters vor der Transition T5 ihre Arbeiten durchführt.

### Transition: T7 – Alive Timeout

**Auslöser:**

Der ALIVE_TIMER ist abgelaufen.

**Aktivität:**

1. Initialisierung von Duplicate.mValid mit `FALSE`
2. Für die sicherheitsgerichteten SI_INPUT Daten werden die Initialwerte eingestellt
3. SI_OK_BIT_IN wird auf 0 gestellt
4. Der ALIVE_TIMER wird mit OpenTMO gestartet

### Transition: T8 – OPEN Timeout

**Auslöser:**

Der ALIVE_TIMER, der mit OpenTMO lief, ist abgelaufen.

**Aktivität:**

1. Initialisierung von Duplicate.mValid mit `FALSE`
2. Es wird T1 getriggert

### Transition: T9 – Deaktivierung

**Auslöser:**

Deaktivierung der Protokollinstanz durch die Applikation.

**Aktivität:**

1. Für die sicherheitsgerichteten SI_INPUT Daten werden die Initialwerte eingestellt
2. SI_OK_BIT_IN wird auf 0 gestellt
3. Initialisierung von Duplicate.mValid mit `FALSE`
4. CONFIG_TO_SEND=`FALSE`; // beim nächsten Mal wieder ohne Konfigurationsdaten probieren
5. Alle evtl. laufenden Timer werden beendet

### Transition: T10 Open Abort

**Auslöser:**

Der Verbindungsaufbau wurde wegen Fehlern abgebrochen.

**Aktivität:**

1. Initialisierung von Duplicate.mValid mit `FALSE`
2. Starte ALIVE_TIMER mit OpenTMO, damit nach dem Ablauf ein erneuter Verbindungsaufbau erfolgt
3. Hier ist die Möglichkeit eine Diagnose im Master zu erzeugen, die aus SLAVE_RESPONSE die Ursache anzeigt, warum die Verbindungsaufnahme gescheitert ist. Gegebenenfalls sind auch schon **OpenRespData** Daten vorhanden, woraus die weitere Diagnose erfolgen kann.

## 5.3 Preset-Handling

Das Handling der verwendeten Preset-Werte für die CRC Berechnungen ist entscheidend für die Fehlerannahmen im Zusammenhang mit dem automatischen Wiederanlauf des TSP.

Das Handling kann in zwei Varianten umgesetzt werden.

### Persistenter Preset

Der TSP Knoten speichert die von ihm generierten und verwendeten Preset-Werte persistent, so dass sie nach einem Neustart wieder zur Verfügung stehen. Im Master ist dies der LastMasterPreset und im Slave ist dies der LastSlavePreset. Die persistenten Werte müssen verfälschungssicher abgelegt werden.

Sind in einem Knoten viele TSP-Verbindungen aktiv, so kann für alle zusammen ein LastxxxPreset, sowohl für Master-, als auch für Slave-Verbindungen, zur Anwendung kommen. Dabei wird beim Generieren eines neuen Presets für verschiedene TSP-Verbindungen dieser globale Preset-Wert benutzt.

Sind persistente Presets nicht verfügbar, z.B. wenn noch nie ein TSP aktiv war oder der persistente Speicher seinen Inhalt verloren hat, so werden die Initialwerte durch den Randomized Preset erzeugt.

### Randomized Preset

Steht im Knoten kein geeigneter Speicher zur Verfügung, so bildet der Knoten bei seinem Hochlauf den Preset-Wert mit einem Zufallsgenerator. Im Master ist dies der LastMasterPreset und im Slave ist dies der LastSlavePreset.

Als Eingang zur Unterstützung einer möglichst guten Streuung müssen Quellen verwendet werden, die bei erneutem Start einen anderen Preset-Wert generierten.

Dazu eignen sich z.B. die Empfangszeitpunkte von Netzwerk-Nachrichten, bezüglich einer internen Zeitbasis (Clock-Register) oder eine vorhandene Real-Time-Clock kann zusammen mit einer internen Zeitbasis eine passende Streuung generieren.

Die Technik zur Generierung des initialen Presets mit einem Zufallswert kann auch beim Einsatz einer globalen Preset-Variable für alle TSP-Verbindungen eines Knotens zum Einsatz kommen.

### Generierung des Presets

Der jeweils verwendete Preset-Wert wird aus dem im Knoten verfügbaren letzten persistenten Preset-Wert (`persistantPreset`) durch Inkrement von 1 gebildet. Die Werte 0, **`INITIAL_MASTER_PRESET`** und **`INITIAL_SLAVE_PRESET`** werden ausgespart, damit die initialen Werte für die Verbindungsaufnahme nicht zur Anwendung kommen.

```
extern udword persistantPreset;

udword
GeneratePreset(void)
{
    persistantPreset= persistantPreset + 1;
    if ((persistantPreset == 0) ||
        (persistantPreset == INITIAL_MASTER_PRESET) ||
        (persistantPreset == INITIAL_SLAVE_PRESET) )
    {
      persistantPreset= persistantPreset + 1;
    }
    return persistantPreset;
}
```

## 5.4 Phasenkonzept

Das Protokoll TSP kann synchron oder asynchron zu den Input- und Output-Phasen einer Applikation bzw. einer Gerätefunktion angewendet werden. Dies gilt sowohl für den Master, wie auch für den Slave. Welches Konzept zu Anwendung kommt, darf für Slave und Master unterschiedlich sein.

### Asynchrones Phasenkonzept

Das Protokoll TSP muss für ein asynchrones Phasenkonzept so implementiert werden, dass in der Input-Phase, d.h. vor einer Applikation, bzw. einer Gerätefunktion, Nachrichten empfangen und in der Output-Phase, d.h. nach einer Applikation, bzw. Gerätefunktion, Nachrichten versendet werden. Dies gilt sowohl für den Master, wie auch für den Slave.

Das asynchrone Modell von TSP bedeutet weiterhin, dass die Gerätefunktion (Slave) aus Sicht des Masters nicht synchron zu seinem Zyklus ausgeführt werden.

Aus Sicht des Slaves bedeutet ein asynchrones Modell, dass er auf die OUTPUT_DATA_INDICATIONs des Masters mit einer Daten RESPONSE, mit nach dem Empfangszeitpunkt ermittelten Daten, reagiert. Ist dies in einem Slave nicht möglich, so können auch die vorhandenen (älteren) Daten in der Daten RESPONSE versendet werden. Dann jedoch muss das Alter der Daten, sowie ein evtl. vorhandener zeitlicher Verzug bei der Ausgabe, bei den Reaktionszeitbetrachtungen hinzu addiert werden.

### Synchrones Phasenkonzept

Beim synchronen Phasenkonzept ist es erforderlich, dass der Slave die Anfragen aus der Input-Phase des Masters von denen der Output-Phase des Masters unterscheiden kann. Dazu wird bei diesem Modell das OkBit nicht für die applikative Steuerung genutzt, sondern das OkBit=0 steht zusammen mit dem Event OUTPUT_DATA_INDICATION für die Anfrage in der Input-Phase und das OkBit=1 für die Anfrage in der Output-Phase des Masters.

Nachfolgend wird INPUT_DATA_INDICATION für die Anfrage in der Input-Phase und OUTPUT_DATA_INDICATION für die Anfrage in der Output-Phase verwendet.

Weiterhin wird TSP um einen Response-Timer im Master erweitert. Dieser Response-Timer überwacht die Antwortzeit der Slaves, ist bei diesem Phasenkonzept kleiner als die Zykluszeit des Masters und kleiner als die WDT.

### Synchrones Slave Konzept

Ziel des synchronen Konzepts beim Slave ist es, die Daten in der Response-Nachricht erst nach dem Empfang der Indication-Nachricht zu ermitteln und damit für das Datenalter keinen zusätzlichen Aufschlag zu benötigen.

Dazu beantwortet der Slave eine INPUT_DATA_INDICATION mit den Daten, die er nach dem Empfang der Master-Nachricht gemessen/ermittelt hat.

Dazu beantwortet der Slave eine OUTPUT_DATA_INDICATION mit den Daten, die zur Ausgabefunktion ermittelt werden.

Aus Sicht des Masters wurden die von ihm empfangenen Daten in seiner Input, respektive Output-Phase gebildet und können beim synchronen Master Konzept zeitlich so behandelt werden, als würde die Applikation lokal Eingänge lesen und Ausgänge schreiben.

### Synchrones Master Konzept

Ziel des synchronen Konzepts beim Master ist es, dass die Daten in der INPUT-Response-Nachricht noch in der Input-Phase des Masters empfangen und anschließend von der Applikation verarbeitet werden. Dazu sendet der Master in der Input-Phase eine INPUT_DATA_INDICATION und wartet auf die Slave Antwort (oder ResponseTMO).

Weiteres Ziel ist es, dass in der Output-Phase des Masters die Ergebnisse der Applikation mit OUTPUT_DATA_INDICATION an die Slaves geschickt werden und diese den Empfang noch in der selben Phase beantworten (oder ResponseTMO).

Welches der Konzepte bzw. welche Kombination angewendet werden sollten, damit eine möglichst kurze Reaktionszeit und eine möglichst kurze Worst-Case-Reaktionszeit erreicht werden kann, hängt von verschiedenen Kriterien ab.

Ist die Master-Zykluszeit verglichen mit der Response-Zeit des Slaves und der zu überbrückenden Netzwerkstrecke verhältnismäßig groß, so kann durch das synchrone Master Konzept die Reaktionszeit minimiert werden. Ab wann dies genau zu einer besseren Reaktionszeit führt, hängt von der Implementierung des asynchronen Konzepts ab.

Ist die Slave-Zykluszeit verglichen mit der Master-Zykluszeit sehr klein, so kann bei Anwendungen des asynchronen Slave-Konzepts das zusätzliche Datenalter u.U. in Kauf genommen werden. Als Gewinn bekommt man schnellere, synchrone Master-Zyklen.

Ist es insbesondere bei Slaves für Input-Daten möglich, die Input-Daten (schnell) nach dem Empfang der INPUT_DATA_INDICATION zu ermitteln, so kann eben dieses gerade dargelegte Datenalter vermieden werden. Dies fällt bei kurzen Master-Zykluszeiten entsprechend mehr ins Gewicht.

Die aus dem jeweiligen Phasenkonzept resultierenden Reaktionszeiten sind in Kapitel 6.4 dargestellt.

## 5.5 Transporttechniken

Das TSP Protokoll enthält keinerlei Mechanismen zur verfügbaren Beherrschung von Nachrichtenverfälschungen oder Verlusten. Es wird erwartet, dass dies die unterlagerte Transporttechnik leistet.

Zu diesem Zweck ist es der Transporttechnik ausdrücklich erlaubt, dass sie von TSP versendete Nachrichten mehrfach versendet[221].

Die Sicherheitstechniken von TSP sind ausreichend wirksam, dass sie ein Schließen der Verbindung und eine damit verbundene Sicherheitsabschaltung beim Auftreten von Nachrichtenverfälschungen oder Einfügungen von Nachrichten *nicht* erfordern, wie dies bei anderen Protokollen erforderlich ist.

Die Transporttechnik ist dafür verantwortlich, dass die Nachrichten an den richtigen Adressaten ausgeliefert werden. Eine eventuell notwendige Umsetzung der Connection-ID auf die Adresstechniken der Transporttechnik ist nicht Gegenstand von TSP.

Bezüglich Bandbreite und Verzögerung stellt TSP an die Transporttechnik keine besonderen Anforderungen, außer dass diese Eigenschaften zu den parametrierten Überwachungszeiten von TSP passen müssen.

Der Transporttechnik ist es erlaubt, TSP-Nachrichten in Fragmenten zu übertragen. TSP selber unterstützt für Prozessdaten jedoch kein Disassembly und Assembly von Fragmenten, dies ist der Transporttechnik überlassen.

Für die ordnungsgemäße Funktion von TSP sollte die Transporttechnik die TSP-Nachrichten an einen Empfänger in der Reihenfolge übertragen, wie sie vom Sender versendet wurden. Die Reihenfolge von Nachrichten verschiedener TSP-Verbindungen obliegt der Transporttechnik.

Sofern die Transporttechnik der TSP-Implementierung die empfangene Nachrichtenlänge liefern kann, kann eine verbesserte Diagnose in TSP durchgeführt werden. Diese Eigenschaft ist jedoch optional.

[221] TSP selber ist es nicht erlaubt eine Nachricht zu wiederholen und darin andere Daten zu transportieren.

# 6 Sicherheitstechnische Analyse des TSP Protokolls

Nachfolgend wird die sicherheitstechnische Analyse des TSP Protokolls betrachtet.

Das TSP Protokoll ist für den Einsatz der sicherheitsgerichteten Datenübertragung innerhalb von Automatisierungssystemen vorgesehenen.

Es wird dargelegt, mit welchen Maßnahmen TSP die Anforderungen von sicherheitsgerichteten Kommunikationsprotokollen erreicht.

## 6.1 Nachrichtenverfälschung

Zur Erkennung der Nachrichtenverfälschung werden bei TSP die CRCs 1 bis 3 eingesetzt.

Als 32 Bit CRC1 wird das im Anwendungsbereich von bis zu 1008 Bits propere Polynom 0x1_F192_2815[222] (gespiegelt 0xA814_498F) verwendet. Das Polynom weist den Faktor (x+1) auf und erkennt daher alle Paritätsfehler. Das Polynom von CRC1 ist unabhängig vom CAN-CRC-Polynom und unabhängig vom dem bei Ethernet verwendetem IEEE-802 Polynom (vgl. CRC-32/8 bei Castagnoli[223]). Die Unstetigkeitsstellen bei 2046 Bit Codeworten liegen jenseits der hier gewählten maximalen Nachrichtenlänge.

Berechnet wird der CRC ausgehend von einem Startwert S über die nicht übertragenen höherwertigen 32 Bits der Sequence-Number. Da der Startwert S immer ungleich 0 ist und der Nachrichten-Header auf Grund der Connection-ID und des Event1s für Prozessdatennachrichten immer ungleich 0 ist, kommt die Schwäche von CRC Prüfsummen bei einer Sequenz, die mit 0 Bytes beginnt, nicht zum tragen.

Ebenfalls nicht relevant ist die Schwäche zyklischer binärer Codes, dass bei einer Rotation eines gültigen Codeworts wieder ein gültiges Codewort entsteht[224], da ein nicht übertragener Teil, die Sequence-Number mit in den CRC eingeht. Risikosenkend wirken sich zusätzlich die Erwartungswerte an fester Stelle, den ersten 16 Bits aus.

Als 32 Bit CRC2 wird das propere Polynom 0x1_3258_3499 (gespiegelt: 0x992C_1A4C) verwendet. Das Polynom weist den Faktor (x+1) auf und erkennt daher alle Paritätsfehler[224]. Das Polynom von CRC2 ist sowohl vom CAN-CRC-Polynom, als auch von dem bei Ethernet verwendetem IEEE-802 Polynom unabhängig[225].

Als 32 Bit CRC3 wird das propere Polynom 0x1_F6AC_FB13 (gespiegelt: 0xC8DF_356F) verwendet. Das Polynom von CRC3 ist unabhängig vom CRC2, unabhängig vom CAN-CRC-Polynom und unabhängig vom dem bei Ethernet verwendetem IEEE-802 Polynom[226].

Berechnet werden die CRCs ausgehend von einem Startwert S über die nicht übertragenen höherwertigen 32 Bits der Sequence-Number. Da der Startwert S immer ungleich 0 ist und der Nachrichten-Header auf Grund von Connection-ID, Event1 und Event2 für Prozessdatennachrichten immer ungleich 0 ist, kommt die Schwäche von CRC Prüfsummen bei einer Sequenz, die mit 0 Bytes beginnt, nicht zum tragen.

Die Fähigkeit von TSP1, rotierte Codeworte zu erkennen, ist in der Art auch bei TSP2 vorhanden. Hier ist der Erwartungswert für 28 der ersten 32 Bits relevant. Weiterhin unterbindet die Art der Verwendung von CRC2 und CRC3 im Nachrichtenformat, dass eine Rotation der gesamten Nachricht

---

222 normale Darstellung mit expliziter, führender 1

223 [Cast93]

224 [Pete81]

225 [Koop02]

226 vgl. CRC-32/6 bei [Cast93]

eine Rotation der Codeworte bezüglich der einzelnen CRCs darstellt. Partiell rotiert werden könnte das zu CRC2 gehörende Codewort[227].

## 6.2 Fehlermodell

Setzt man als Modell für die Übertragungseinrichtung den binär-symmetrischen Übertragungskanal voraus, so ergibt sich bei gegebener Hamming-Distanz die Restfehlerwahrscheinlichkeit pro Stunde unerkannt empfangener verfälschter Nachrichten mit Gleichung (2.2). Dabei wird für propere Polynome *R()* mit der Gleichung (2.6) angewendet.

Zusätzlich zur Gleichung (2.6) werden für die verwendeten CRCs deren Eigenschaften, wie Erkennung von Burst-Fehlern bis zur Länge *k* bzw., wenn ihr Polynom den Term *(x+1)* enthält, die Erkennung aller ungeraden Bitfehler (Parity-Funktion), eingerechnet.
Die Restfehlerwahrscheinlichkeit einer TSP1 Nachricht wird nachfolgend für die Bitfehlerraten $p=10^{-3}$ und $p=10^{-2}$ berechnet.

Für das Nachrichtenformat TSP1 können wegen der eineindeutigen sicheren Erwartungshaltung von Connection-ID, OkBit, Seq-No-Low und Event1 von der Bitfehlerbetrachtung ausgeschlossen werden. Bitfehler in diesen Werten werden im TSP1 immer erkannt bzw. sicher behandelt .

Die Hamming-Distanz des gewählten Polynoms CRC1 ist für unterschiedliche Bitlängen wie folgt in die Berechnungen eingegangen. Für die Nachrichtenlängen bis 6 Nutzdatenbytes wurden die Hamming-Distanzen im Rahmen dieser Arbeit berechnet, für größere Längen die Ergebnisse von Castagnoli[228] herangezogen.

*Tabelle 6.1: TSP1 Hamming-Distanz des CRC1*

| ***Bitlängen*** | ***Nutzdatenbytes*** | ***Hamming-Distanz CRC1 0x1_F192_2815*** |
|---|---|---|
| $N+k=48$ | 2 | 12 |
| $N+k=64$ | 4 | 10 |
| $N+k=80$ | 6 | 10 |
| $N+k=96$ | 8 | 10 |
| $N+k=104 \ldots 992$ | 9 .. 120 | 8 |

Zur Verifikation der approximativ berechneten Restfehlerraten des CRC1 von TSP1 wurden für die Nachrichtenlängen bis zu 80 Bits die Gewichte, sowie die Hamming-Distanz des CRC1 berechnet.

Die realen Gewichte für Nachrichten (Anzahl Fehler) mit 2, 4 und 6 Bytes Nutzdaten, sowie die Gewichte

$$\frac{1}{2^k} \bullet \binom{N+k}{n} \qquad (6.1)$$

aus der Approximation werden in den nachfolgenden Diagrammen dargestellt. Als Orientierung sind die Binominialkoeffizienten (*B.K.*) und die Binominialkoeffizienten mal $2^{-k}$ mit angegebenen.

---

[227] partielle Rotation, z.B. denkbar bei fragmentierter Übertragung und Synchronisationsfehler nur eines Fragments

[228] [Cast93]

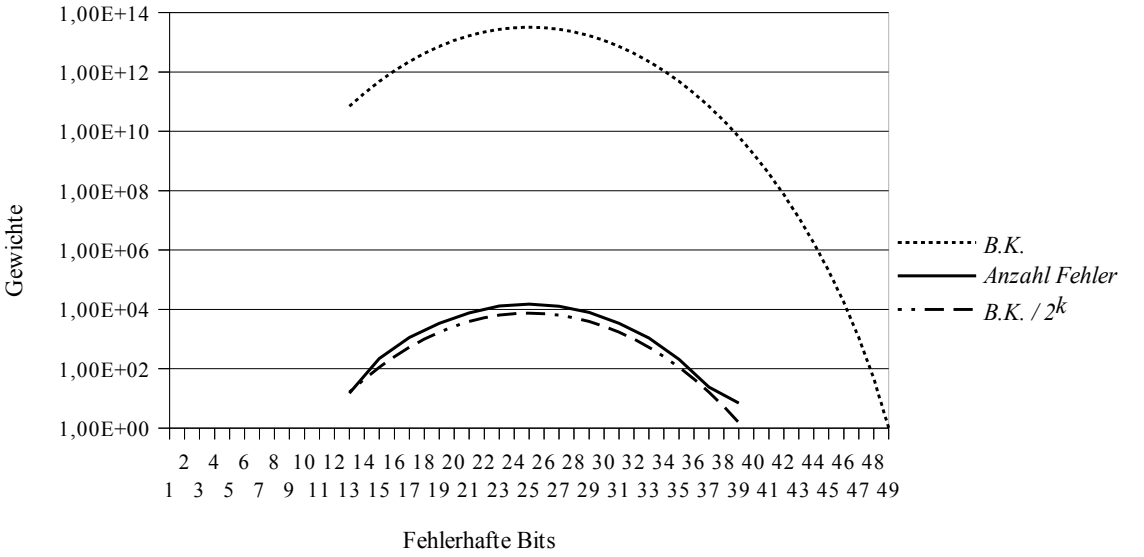


*Abbildung 6.1: TSP1 Gewichte für 2 Bytes Nutzdaten*

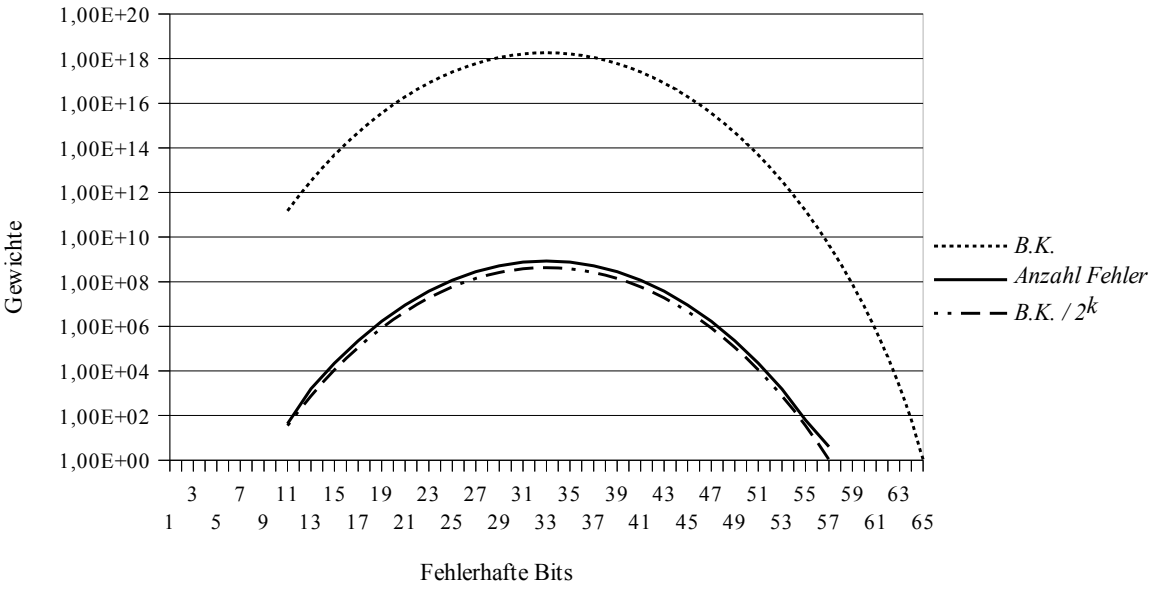


*Abbildung 6.2: TSP1 Gewichte für 4 Bytes Nutzdaten*

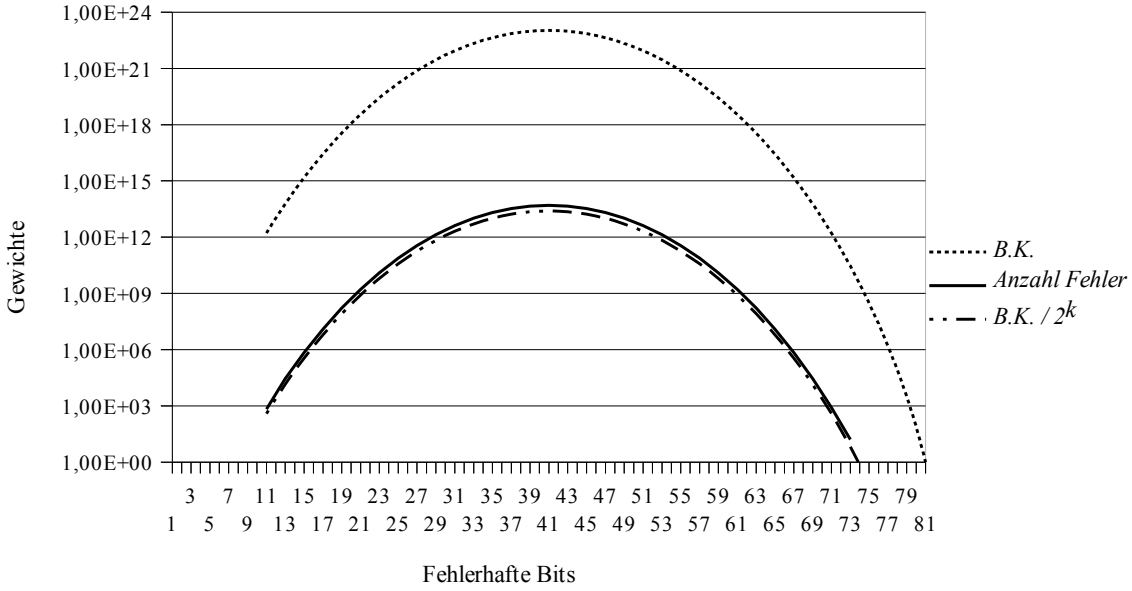


*Abbildung 6.3: TSP1 Gewichte für 6 Bytes Nutzdaten*

Beim hier verwendeten CRC1 zeigt die Approximation bei 2, 4 und 6 Bytes Nutzdaten über den größten Bereich fehlerhafter Bits zu niedrige Gewichte. Da der erste Koeffizient von $B.K./2^k$ für 2 Bytes Nutzdaten größer ist, als die konkrete Anzahl Fehler, ist die reale Restfehlerwahrscheinlichkeit bei einer Bitfehlerrate $p$ von $10^{-2}$ geringer als die approximativ berechnete. Bei 4 und 6 Bytes Nutzdaten verhält es sich umgekehrt und somit ist die reale Restfehlerwahrscheinlichkeit hier höher als die approximativ berechnete.
Aus den ermittelten realen Gewichten folgt die konkrete Restfehlerrate von TSP1. Diese werden den approximativen Rechnungen auf Basis der Gleichung (2.6)

$$R_{CRC1}(p, N, k) \approx \frac{1}{2^k} \bullet \sum_{n=d}^{N+k} \binom{N+k}{n} \bullet p^n \bullet (1-p)^{(N+k-n)} \qquad (6.2)$$

jedoch nur für gerade n, da der CRC den Term *(x+1)* enthält, gegenüber gestellt.

*Tabelle 6.2: TSP1 Vergleich von Approximation und konkreten Gewichten bei $p = 10^{-2}$*

| ***Bitlängen*** | ***TSP1*** $R_{Gewichte}()$ | ***Approximation*** $R_{CRC1}()$ |
|---|---|---|
| 48 Bits | $1{,}0462 \bullet 10^{-23}$ | $1{,}13045 \bullet 10^{-23}$ |
| 64 Bits | $2{,}50852 \bullet 10^{-19}$ | $2{,}05418 \bullet 10^{-19}$ |
| 80 Bits | $3{,}31971 \bullet 10^{-18}$ | $1{,}90408 \bullet 10^{-18}$ |

*Tabelle 6.3: TSP1 Vergleich von Approximation und konkreten Gewichten bei $p = 10^{-3}$*

| ***Bitlängen*** | ***TSP1*** $R_{Gewichte}()$ | ***Approximation*** $R_{CRC1}()$ |
|---|---|---|
| 48 Bits | $1{,}33696 \cdot 10^{-35}$ | $1{,}56472 \cdot 10^{-35}$ |
| 64 Bits | $4{,}07400 \cdot 10^{-29}$ | $3{,}34135 \cdot 10^{-29}$ |
| 80 Bits | $6{,}22843 \cdot 10^{-28}$ | $3{,}57437 \cdot 10^{-28}$ |

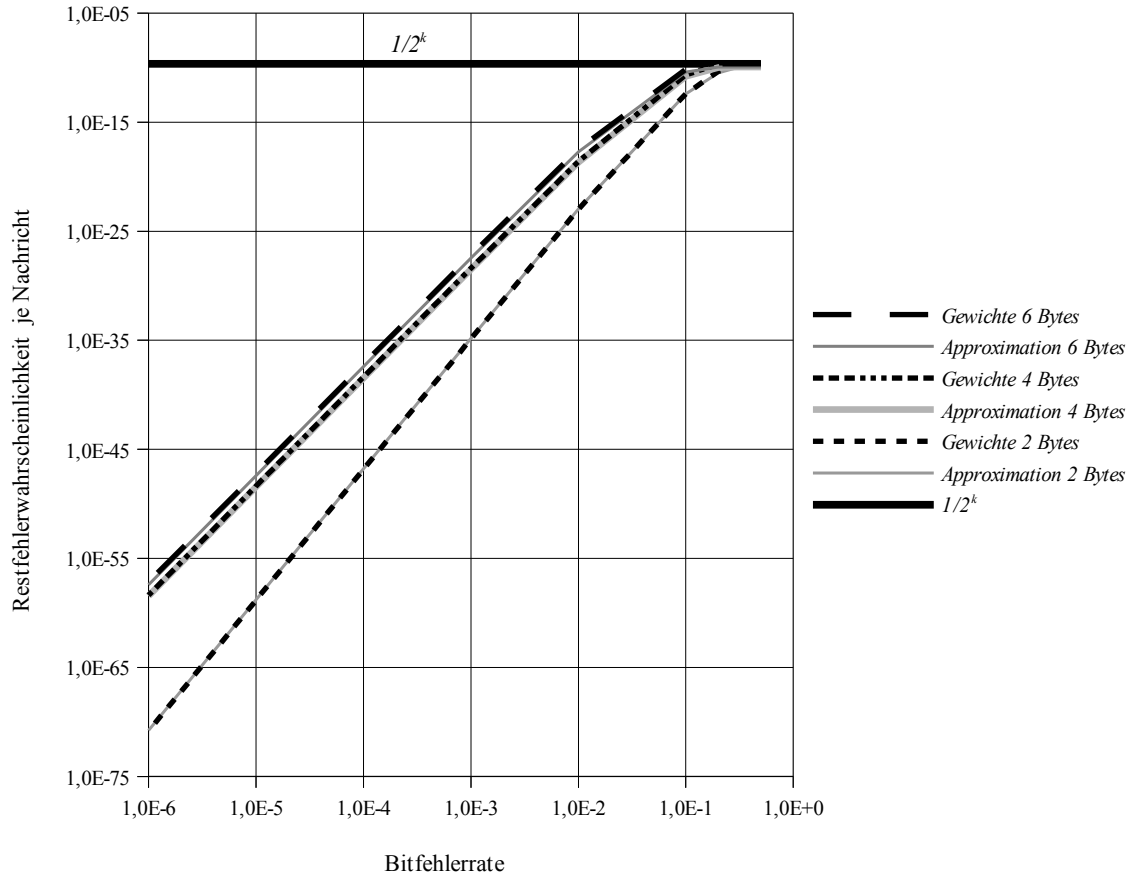


*Abbildung 6.4: Vergleich der TSP1 Restfehler Berechnungsmethoden*

Da die Unterschiede der Restfehlerwahrscheinlichkeiten für die Nutzdatenlängen 2, 4 und 6 Bytes gering sind, sie bewegen sich in der gleichen Größenordnung und mit der sehr konservativen Annahme einer Bitfehlerrate von $10^{-2}$, kann in diesem Fall auch die Approximation Gleichung (2.6) für CRC1 zur Anwendung kommen.

## Anwendungsgrenzen von TSP1

Nachfolgende Tabelle zeigt, in Abhängigkeit der anzunehmenden Bitfehlerrate, wie viele Nutzdaten-Bytes beispielhaft mit wie vielen Nachrichten/Sekunde und Verbindungen für eine Restfehlerwahrscheinlichkeit Lambda von ≤ 1% von SIL3 mit TSP1 Nachrichten möglich sind.

*Tabelle 6.4: Restfehlerraten für TSP1 Nachrichten verschiedener Nachrichtenlängen*

| ***Bitfehler-rate*** | ***Verbindungen einer Sicherheitsfunktion*** | ***Nachrichten je Sekunde je Verbindung*** | ***Nutzdaten-Bytes maximal (N+k/d)*** | ***R() nach Gewichten oder $R_{CRCI}()$*** |
|---|---|---|---|---|
| $10^{-2}$ | 8190 | 3.241.739 (3.000.283)[229] | 2 (48/12) | $1{,}0462 \cdot 10^{-23}$ ($1{,}13045 \cdot 10^{-23}$) (229) |
| $10^{-2}$ | 1000 | 1007 (1.352)[229] | 4 (64/10) | $2{,}50852 \cdot 10^{-19}$ ($2{,}05418 \cdot 10^{-19}$) (229) |
| $10^{-2}$ | 100 | 836 (1458)[229] | 6 (80/10) | $3{,}31971 \cdot 10^{-18}$ ($1{,}90408 \cdot 10^{-18}$) (229) |
| $10^{-2}$ | 20 | 1248 | 8 (96/10) | $1{,}11283 \cdot 10^{-17}$ |
| $10^{-3}$ | 8190 | $2{,}17 \cdot 10^{+18}$ | 2 (48/12) | $1{,}56472 \cdot 10^{-35}$ |
| $10^{-3}$ | 8190 | $1{,}01 \cdot 10^{+12}$ | 4 (64/10) | $3{,}34135 \cdot 10^{-29}$ |
| $10^{-3}$ | 8190 | $9{,}48 \cdot 10^{+10}$ | 6 (80/10) | $3{,}57437 \cdot 10^{-28}$ |
| $10^{-3}$ | 8190 | $1{,}40 \cdot 10^{+10}$ | 8 (96/10) | $2{,}40992 \cdot 10^{-27}$ |
| $10^{-3}$ | 8190 | 44.355 | 14 (144/8) | $7{,}64651 \cdot 10^{-22}$ |
| $10^{-3}$ | 500 | 623 | 40 (352/8) | $8{,}9159 \cdot 10^{-19}$ |
| $10^{-3}$ | 50 | 225 | 64 (544/8) | $2{,}46799 \cdot 10^{-17}$ |
| $10^{-3}$ | 5 | 466 | 80 (672/8) | $1{,}19104 \cdot 10^{-16}$ |
| $10^{-3}$ | 5 | 27 | 120 (992/8) | $1{,}98815 \cdot 10^{-15}$ (230) |

*Anmerkungen:*

- *20 • 1.248 für eine 8 Byte Nutzdaten-Nachrichten entspricht bei Ethernet (minimales Paket 64 Bytes) einer Übertragungsrate von 12,779 MBit/s für eine Sicherheitsfunktion.*
- *Bei 2 Bytes Nutzdaten und 3 Mio. Nachrichten für 8.190 Verbindungen sind dies wegen der größeren Nachrichtenmenge 13,59 TBit/s für eine Sicherheitsfunktion.*
- *Wird ein Transportsystem ohne Nachrichten-Overhead eingesetzt, so ergibt sich bei 2 Bytes Nutzdaten eine mögliche Übertragungsrate von 1,67 TBit/s je Sicherheitsfunktion.*
- *Solange das Produkt aus (Nachrichten je Sekunde) und (Verbindungen einer Sicherheitsfunktion) nicht überschritten wird, sind auch andere Verhältnisse von Nachrichten und Verbindungen zulässig.*
- *Vergleicht man TSP1 mit den kurzen Nachrichten der aktuellen CIP-Safety[231] oder PROFIsafe[232] Protokolle, so erreicht TSP1 bei einer Bitfehlerrate von $10^{-2}$ eine um ca. 6 Größenordnungen geringere Restfehlerwahrscheinlichkeit für eine unerkannt verfälschte Nachricht.*

[229] Berechnet auf Basis der Approximation

[230] ob der geringen Nachrichtenanzahl nur beschränkt einsatzfähig

[231] [CIP5-2.2]

[232] [PROFIsafeV2]

Für das TSP2 Format können wegen der eineindeutigen Erwartungshaltung von Connection-ID-Low/High, OkBit, Reserve-Bits, Seq-No-Low, Event1 und Event2 von der Bitfehlerbetrachtung ausgeschlossen werden. Bitfehler in diesen Werten werden immer erkannt bzw. sicher behandelt.

Die Hamming-Distanz des gewählten Polynoms CRC2 ist für unterschiedliche Bitlängen wie folgt in die Berechnungen eingegangen. Für die Nachrichtenlängen bis 6 Nutzdatenbytes wurden die Hamming-Distanzen im Rahmen dieser Arbeit berechnet, für größere Längen die Ergebnisse von Koopman[233] herangezogen.

*Tabelle 6.5: TSP2 Hamming-Distanz des CRC2*

| ***Bitlängen*** | ***Nutzdatenbytes*** | ***Hamming-Distanz CRC2 0x1_992C_1A4C*** |
|---|---|---|
| $N+k$= 48 | 2 | 10 |
| $N+k$= 64 | 4 | 8 |
| $N+k$= 80 | 6 | 8 |
| $N+k$= 96 | 8 | 8 |
| $N+k$= 132... 160 | 9 ... 16 | 8 |
| $N+k$= 168... 32768 | 17 ... 4092 | 6 |

Für CRC3 werden die Hamming-Distanzen der folgenden Tabelle benutzt. Für die Nachrichtenlängen bis 6 Nutzdatenbytes wurden die Hamming-Distanzen im Rahmen dieser Arbeit berechnet, für größere Längen die Ergebnisse von Castagnoli[234] herangezogen.

*Tabelle 6.6: TSP2 Hamming-Distanz des CRC3*

| ***Bitlängen*** | ***Nutzdatenbytes*** | ***Hamming-Distanz CRC3 0x1_F6AC_FB13*** |
|---|---|---|
| $N+k$= 48 | 2 | 10 |
| $N+k$= 64 | 4 | 10 |
| $N+k$= 80 | 6 | 9 |
| $N+k$= 96 | 8 | 8 |
| $N+k$= 104 ... 304 | 9 ... 34 | 8 |
| $N+k$= 312 ... 32768 | 35 ... 4092 | 6 |

Die angegebenen Bitlängen beziehen sich auf die Nutzdatenbytes und den jeweiligen CRC. Die gesamte, für die Bitfehlerrate zu betrachtende Bitlänge der TSP2 Nachricht ist um 32 Bits größer, da sie beide CRCs umfasst.

Da die Polynome voneinander unabhängig sind, insbesondere keine gemeinsamen Faktoren besitzen, werden für die approximative Berechnung der Restfehlerwahrscheinlichkeiten die Hamming-Distan-

[233] [Koop02]

[234] [Cast93]

zen addiert. Entsprechend der Anwendung der beiden 32-Bit CRCs wird $k$=64 gesetzt. Mit der selben Motivation werden die Restfehlerwahrscheinlichkeiten der CRCs als unabhängige Wahrscheinlichkeiten betrachtet und entsprechend multipliziert.
Nachfolgende Tabelle zeigt die Restfehlerwahrscheinlichkeit einer Nachricht nach den approximativen Rechenverfahren:

1. Summe der Hamming-Distanzen (siehe Gleichung (2.8))
2. Produkt der Einzelwahrscheinlichkeiten (siehe Gleichung (2.9))

*Tabelle 6.7: Restfehlerraten für TSP2 Nachrichten verschiedener Nachrichtenlängen*

| ***Bitfehler-rate*** | ***Nutzdatenbytes maximal (N+k/d)*** | ***R(p,N,k) für TSP2 mit R(d=d2+d3)*** | ***R(p,N,k) für TSP2 mit RCRC2()•RCRC3()*** |
|---|---|---|---|
| $10^{-2}$ | 2 (80/20) | $1{,}04945 \cdot 10^{-41}$ | $1{,}08279 \cdot 10^{-40}$ |
| $10^{-2}$ | 4 (96/17) | $3{,}39114 \cdot 10^{-37}$ | $1{,}21002 \cdot 10^{-35}$ |
| $10^{-2}$ | 6 (112/17) | $5{,}99171 \cdot 10^{-36}$ | $1{,}08395 \cdot 10^{-31}$ |
| $10^{-2}$ | 8 (128/16) | $1{,}64857 \cdot 10^{-32}$ | $1{,}65392 \cdot 10^{-30}$ |
| $10^{-2}$ | 14 (176/16) | $2{,}19675 \cdot 10^{-30}$ | $5{,}19817 \cdot 10^{-28}$ |
| $10^{-2}$ | 40 (384/12) | $2{,}5169 \cdot 10^{-23}$ | $1{,}03237 \cdot 10^{-22}$ |
| $10^{-2}$ | 64 (576/12) | $5{,}59503 \cdot 10^{-22}$ | $4{,}05586 \cdot 10^{-21}$ |
| $10^{-2}$ | 120 (1024/12) | $1{,}04421 \cdot 10^{-20}$ | $1{,}23899 \cdot 10^{-20}$ |
| $10^{-2}$ | 238 (1968/12) | $2{,}66475 \cdot 10^{-20}$ | $1{,}35513 \cdot 10^{-20}$ |

Zur Verifikation der Restfehlerraten der CRCs von TSP2 wurden für die Bitfehlerrate $10^{-2}$ und für die Nachrichtenlängen bis zu 112 Bits die konkreten Gewichte, sowie die Hamming-Distanzen der CRCs CRC2 und CRC3, sowie für TSP2 insgesamt berechnet.
Die realen Gewichte für Nachrichten (Anzahl Fehler) mit 2, 4 und 6 Bytes Nutzdaten, sowie die Gewichte

$$\frac{1}{2^k} \bullet \binom{N+k}{n} \qquad (6.3)$$

aus der Approximation werden in den nachfolgenden Diagrammen dargestellt.

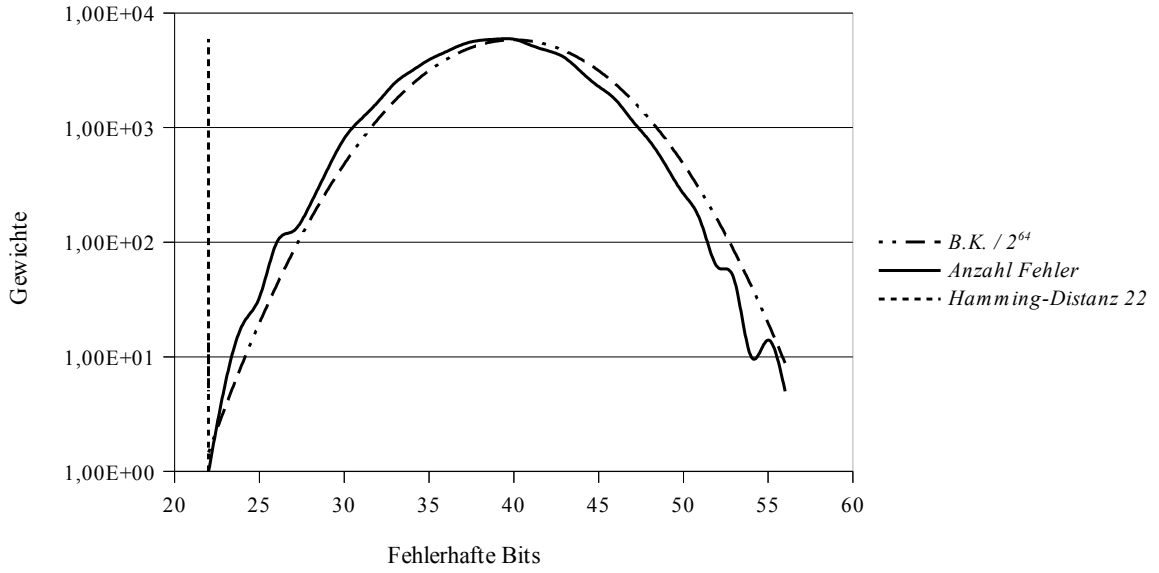


*Abbildung 6.5: TSP2 Gewichte für 2 Bytes Nutzdaten*

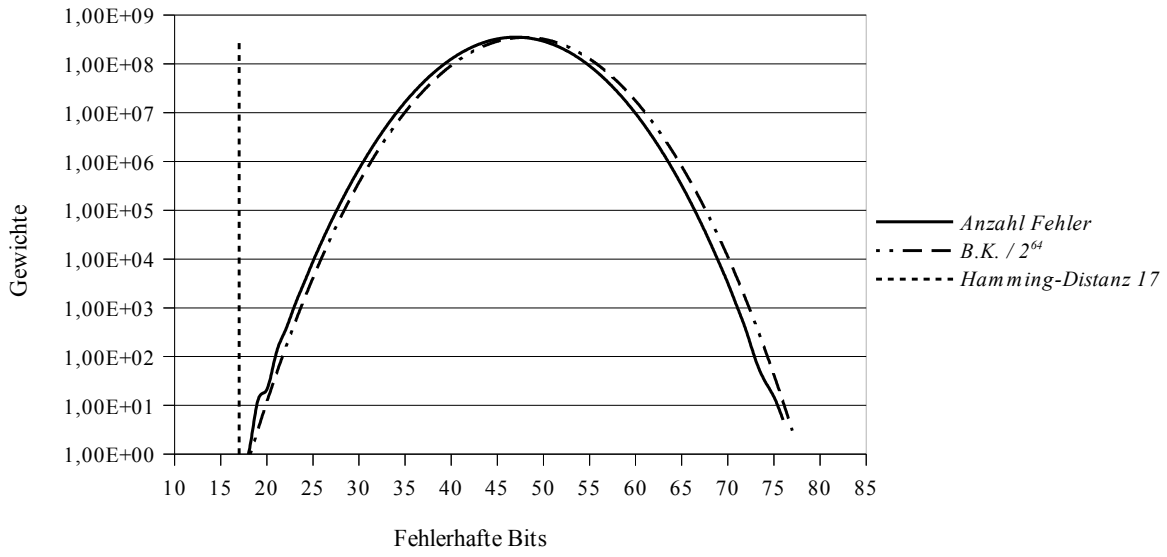


*Abbildung 6.6: TSP2 Gewichte für 4 Bytes Nutzdaten*

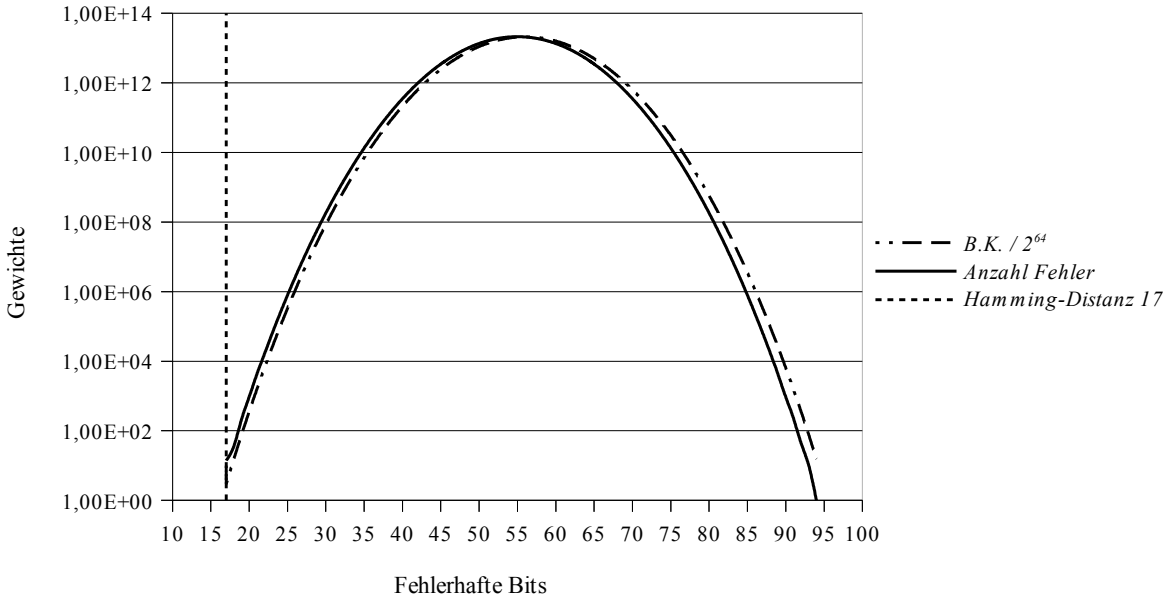


*Abbildung 6.7: TSP2 Gewichte für 6 Bytes Nutzdaten*

Aus diesen konkreten Gewichten folgt die konkrete Restfehlerrate der kombinierten CRCs. Im Rahmen der Berechnungen wurden die Gewichte der einzelnen CRCs (CRC2 und CRC3) ermittelt und daraus die konkrete Restfehlerrate von CRC2 und CRC3 berechnet.

Diese Berechnungen werden den approximativen Rechnungen mit $R_{CRC2}() \bullet R_{CRC3}()$ und $R(d=d_{CRC2}+d_{CRC3})$ gegenüber gestellt.

*Tabelle 6.8: TSP2 Restfehlerwahrscheinlichkeiten auf Basis konkreter Gewichte*

| ***effektive Länge*** | ***TSP2*** | ***CRC2 (Approx. $R_{CRC2}()$)*** | ***CRC3 (Approx. $R_{CRC3}()$)*** |
|---|---|---|---|
| 80 Bits | $1{,}32552 \bullet 10^{-44}$ [d=22] | $2{,}21845 \bullet 10^{-20}$ ($1{,}04057 \bullet 10^{-20}$) [d=10] | $4{,}46787 \bullet 10^{-20}$ ($1{,}04057 \bullet 10^{-20}$) [d=10] |
| 96 Bits | $4{,}57174 \bullet 10^{-35}$ [d=17] | $3{,}99583 \bullet 10^{-16}$ ($5{,}89053 \bullet 10^{-17}$) [d=8] | $2{,}79061 \bullet 10^{-19}$ ($5{,}89053 \bullet 10^{-17}$) [d=10] |
| 112 Bits | $5{,}53718 \bullet 10^{-34}$ [d=17] | $1{,}95143 \bullet 10^{-15}$ ($3{,}29234 \bullet 10^{-16}$) [d=8] | $1{,}91610 \bullet 10^{-17}$ ($3{,}29234 \bullet 10^{-16}$) [d=9] |

*Tabelle 6.9: TSP2 Vergleich der Restfehlerwahrscheinlichkeiten*

| ***effektive Länge*** | ***TSP2*** | ***CRC2 • CRC3 (Approx. $R_{CRC2}() \bullet R_{CRC3}()$)*** | ***Approximation R(d=d1+d2)*** |
|---|---|---|---|
| 80 Bits | $1{,}32552 \bullet 10^{-44}$ [d=22] | $9{,}91177 \bullet 10^{-40}$ ($1{,}08279 \bullet 10^{-40}$) | $1{,}04945 \bullet 10^{-41}$ [d=20] |
| 96 Bits | $4{,}57174 \bullet 10^{-35}$ [d=17] | $1{,}11508 \bullet 10^{-34}$ ($1{,}21002 \bullet 10^{-35}$) | $3{,}39114 \bullet 10^{-37}$ [d=18] |
| 112 Bits | $5{,}53718 \bullet 10^{-34}$ [d=17] | $3{,}73914 \bullet 10^{-32}$ ($1{,}08395 \bullet 10^{-31}$) | $5{,}99171 \bullet 10^{-36}$ [d=17] |

Das Produkt CRC2 • CRC3 in Tabelle 6.9 ist das Ergebnis der durch die Simulation ermittelten Restfehlerwahrscheinlichkeiten der einzelnen CRCs. Die Werte in Klammern darunter ergeben sich aus dem Produkt der Approximationsrechnungen der einzelnen CRCs gemäß Gleichungen (2.6) und (2.9).

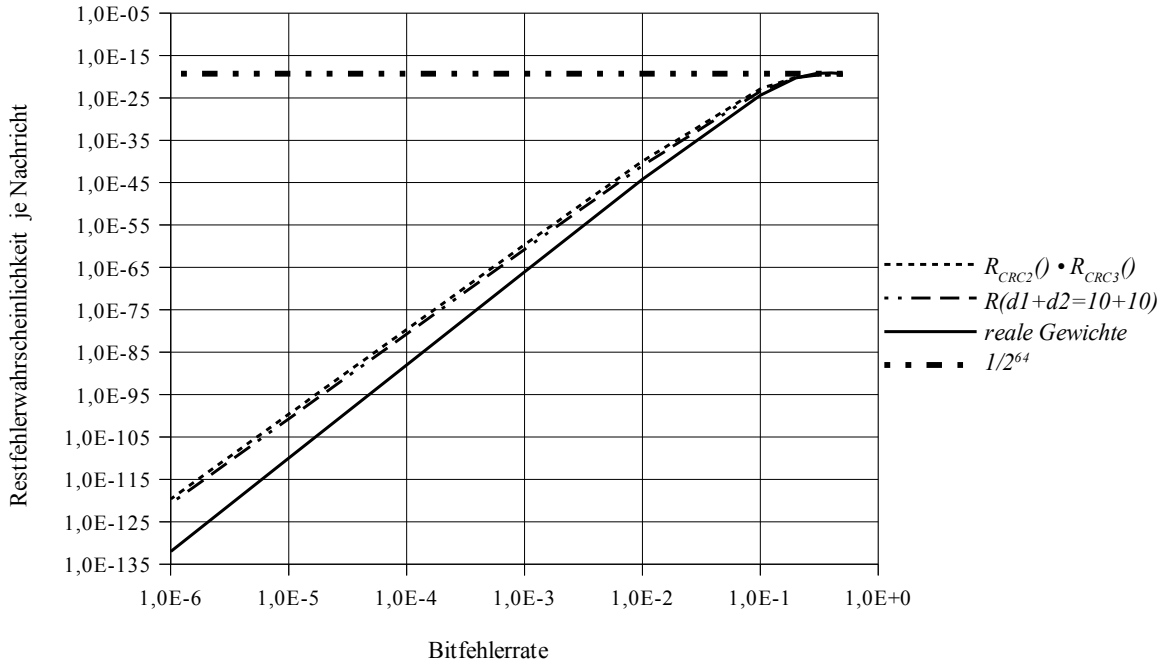


*Abbildung 6.8: Vergleich der TSP2 Restfehlerberechnungen für 2 Nutzdatenbytes*

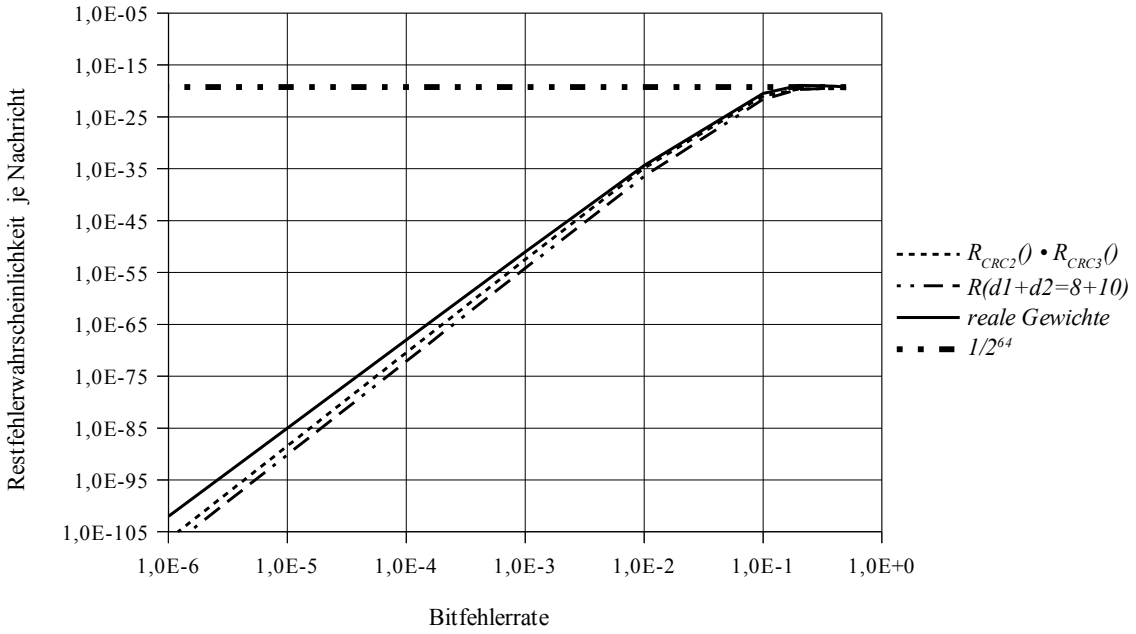


*Abbildung 6.9: Vergleich der TSP2 Restfehlerberechnungen für 4 Nutzdatenbytes*

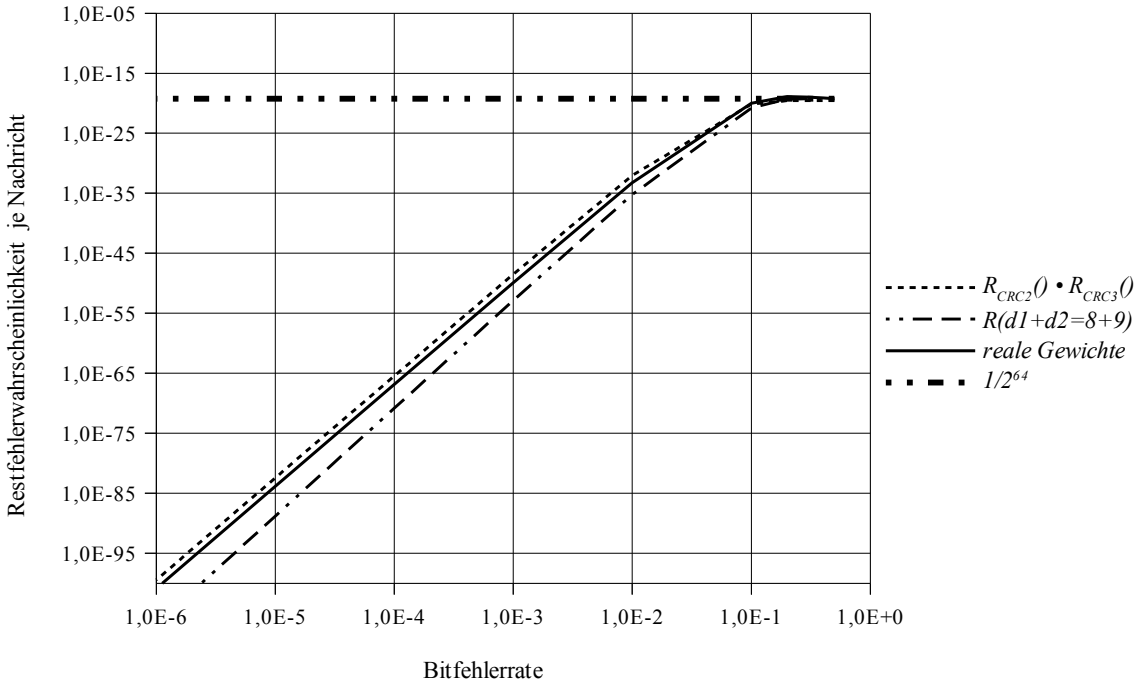


*Abbildung 6.10: Vergleich der TSP2 Restfehlerberechnungen für 6 Nutzdatenbytes*

Für die hier betrachteten Nachrichtenlängen und TSP2 CRCs zeigt sich, dass die Hamming-Distanz $d$ von TSP2 um die Summe $d_{CRC2}+d_{CRC3}$ mit einer Abweichung von 0 bis -1 schwankt.

Das approximative Rechenverfahren mit $R(d=d_{CRC2}+d_{CRC3})$ schwankt ebenso mit der Schwankung der Hamming-Distanzen und liefert um zwei bis drei Größenordnungen zu „optimistische“ oder zu „pessimistische“ Werte.

Das Rechenverfahren mit den realen Gewichten für *CRC2 • CRC3*, wie auch das approximative Verfahren mit $R_{CRC2}() \bullet R_{CRC3}()$, ergibt meist zu „pessimistische“ Werte. Diese sind bei einer Bitfehlerrate von $10^{-2}$ um bis zu 3 Größenordnungen zu hoch.

Die CRC-Kombination in TSP2 zeigt anscheinend größere Abweichungen von den $B.K./2^{64}$ je kürzer die Nachricht ist. Dies passt in das Bild, dass der Faktor $1/2^{64}$ der Approximation vorwiegend für große Nachrichtenlängen gegen den realen Wert der Gewichte tendiert.

Die Berechnung der Gewichte zeigte auch, dass bei einer Bitfehlerrate von $10^{-2}$ nur die ersten 1 bis 4 Gewichte für das Ergebnis relevant sind. Jedoch lagen gerade diese bei CRC2 und CRC3, je nach Nachrichtenlänge, über oder unter der Approximation.

An dieser Stelle sei ausdrücklich darauf hingewiesen, dass die hier ermittelten Ergebnisse nur beispielhaften Charakter haben und nur für TSP1 und TSP2, sowie ausschließlich für die betrachteten Nachrichtenlängen, Gültigkeit haben.

### Anwendungsgrenzen TSP2

Für die Berechnung der Nachrichten je Sekunde und Verbindung wurde eine Größenordnung Reserve vom approximativen Berechnungsergebnis $R_{CRC2}() \bullet R_{CRC3}()$ berücksichtigt, um der unsicheren mathematischen Grundlage Rechnung zu tragen. Dies erfolgte trotz der zu hohen Restfehlerwahrscheinlichkeiten bei den betrachteten kurzen Nachrichten. Die Verwendung des Ansatzes *d=d2+d3* wird zu Gunsten einer höheren Sicherheitsreserve nicht verwendet.
Bei den bekannten Gewichten wurde die Nachrichtenrate ohne Reserve berechnet.

*Tabelle 6.10: Restfehlerraten für TSP2 Nachrichten*

| ***Bitfehlerrate*** | ***Verbindungen einer Sicherheitsfunktion*** | ***Nachrichten je Sekunde je Verbindung*** | ***Nutzdatenbytes maximal (N+k)*** | ***R() nach Gewichten oder $R_{CRC2}() \bullet R_{CRC3}()$*** |
|---|---|---|---|---|
| $10^{-2}$ | 65534 | $3{,}19774 \bullet 10^{+26}$ | 2 (80) | $1{,}32552 \bullet 10^{-44}$ |
| $10^{-2}$ | 65534 | $9{,}27148 \bullet 10^{+16}$ | 4 (96) | $4{,}57174 \bullet 10^{-35}$ |
| $10^{-2}$ | 65534 | $7{,}65495 \bullet 10^{+15}$ | 6 (112) | $5{,}53718 \bullet 10^{-34}$ |
| $10^{-2}$ | 65534 | 256.280.235.825 | 8 (128) | $1{,}65392 \bullet 10^{-30}$ |
| $10^{-2}$ | 65534 | 815.417.779 | 14 (176) | $5{,}19817 \bullet 10^{-28}$ |
| $10^{-2}$ | 10.000 | 26.906 | 40 (384) | $1{,}03237 \bullet 10^{-22}$ |
| $10^{-2}$ | 2.048 | 3.344 | 64 (576) | $4{,}05586 \bullet 10^{-21}$ |
| $10^{-2}$ | 2.048 | 2.189 | 120 (1024) | $1{,}23899 \bullet 10^{-20}$ |
| $10^{-2}$ | 1.024 | 2.001 | 238 (1968) | $1{,}35513 \bullet 10^{-20}$ |

Selbst bei 238 Bytes Nutzdaten ist eine Nachrichtenrate von 4 GBit/s je Sicherheitsfunktion möglich; bei 2 Bytes sind es 2,35•1024 GBit/s. Diese Grenzen sind jedoch derzeit, mit im Industriellen Umfeld eingesetzter Technik, nicht zu erreichen.

Nutzdatenlängen für die Bitfehlerrate $10^{-3}$ sind hier nicht mehr angegeben, da diese ohnehin bessere Werte aufweisen.

Der Binär-Symmetrische-Kanal setzt voraus, dass die Verfälschung mehrerer Bits einer Nachricht voneinander unabhängig sind. Dieses Modell lässt sich auf serielle Übertragungssysteme gut anwenden[235], jedoch besteht ein Übertragungssystem auch aus nicht sicherheitsgerichteten Komponenten mit parallelen Strukturen, wie beispielsweise FIFOs und anderen Speicherbausteinen oder gar komplexe Ethernet-Controller und Prozessoren.

Bei diesen sind zum Beispiel auch Fehler statischer Ausfälle (Stuck-At)[236] und Übersprechen zwischen Datenleitungen (cross-talk)[236] zu berücksichtigen. Übertragen auf einen seriellen Datenstrom bedeutet z.B. ein Stuck-At, dass jedes n-te Bit einen festen Wert hat oder bei einem Übersprechen, dass z.B. jedes n-te und (n+x)-te Bit immer den gleichen Wert haben. Damit sind verfälschte Bits nicht mehr unabhängig voneinander und der Sicherungsmechanismus muss nachweisen, wie er effizient er solche und andere Fehlermuster aufdeckt.

Da die Art der eingesetzten Komponenten beim Black-Channel Ansatz nicht betrachtet werden sollen, bietet sich folgende Restfehlerabschätzung an. Es wird angenommen, dass jedes empfangene Nachrichtenmuster gleich wahrscheinlich ist, d.h. und insbesondere unabhängig von dem der versendeten Nachricht. Die Restfehlerwahrscheinlichkeit einer Nachricht berechnet sich damit aus $A_n^{N+k}$, der Menge von Codeworten der Länge $N+k$, die bei $n$-Bit-Verfälschungen durch den CRC Sicherungsmechanismus nicht erkannt werden und $2^{N+k}$ die Menge aller möglichen Codeworte.

Da die Anzahl der Codeworte mit unerkannten Fehlern $2^N$ ist, erhält man als Restfehlerwahrscheinlichkeit $R$ der CRC gesicherten Nachricht:

$$R \approx \frac{\sum_{n=1}^{N+k} A_n^{N+k}}{2^{N+k}} = \frac{1}{2^k} \qquad (6.4)$$

Dies entspricht der Restfehlerwahrscheinlichkeit gemäß Gleichung (2.6), mit einer angenommenen Bitfehlerrate von $p$=0,5. Für TSP1 erhält man $R$=2,32•10$^{-10}$, die für den praktischen Einsatz mit dem Anspruch an 1% von SIL3 ungeeignet ist.

Für TSP2 berechnet sich, auch für die untersuchten Gewichte der Nachrichten, $R=2^{-k}=2^{-64}=5{,}42\bullet10^{-20}$. Unabhängig von der Nachrichtenlänge können 5,12•10$^6$ Nachrichten je Sekunde je Sicherheitsfunktion mit dem Anspruch an 1% von SIL3 genutzt werden.

Zur Erkennung von Nachrichtenwiederholungen verwendet TSP die Sequence-Number. Der Empfängerknoten hat eine Erwartungshaltung bzgl. der Sequence-Number.

Eine falsche Sequence-Number äußert sich immer durch einen CRC Fehler und gegebenenfalls zusätzlich durch den falschen niederwertigen übertragenen Anteil der Sequence-Number.

Bei der Sequence-Number handelt es sich um einen streng monoton steigenden, umlaufenden 32 Bit Zähler.

Beim Einsatz von TSP ist zu beachten, dass die eingesetzten Übertragungseinrichtungen gegebenenfalls eine Nachricht mehrfach übertragen. Das heißt, eine TSP Nachricht wird z.B. gemäß dem Zyklus des Standardbussystems solange übertragen, bis eine neue TSP Nachricht zur Übertragung bereit gestellt wird.

Daher stellt es für TSP keinen Sonderfall dar, dass Nachrichten wiederholt werden. Erkannt wird die Wiederholung an der sich wiederholenden Sequence-Number und dem gleichen CRC, da die Daten bei wiederholten Paketen nicht geändert werden.

Die als Duplikat erkannten Nachrichten werden vom TSP Empfänger verworfen und stellen keinen Fehler dar.

---

235 [EN50159]

236 [IEC61508-2] Tabelle A.1, A.3, A.7, A.8

Die Anlauf-Sequenz von TSP sieht eine spezielle Signalisierung vor (siehe `TSP_OpenInd`, `TSP_OpenResp`).

Nachfolgend wird betrachtet, was bei der Wiederholung einer solchen Sequenz oder einer Wiederholung einzelner Nachrichten geschieht.

**Slave**

- Wird eine Open-Response vom Slave empfangen, so wird sie verworfen.
- Wird eine Open-Indication Nachricht während einer bestehenden Verbindung empfangen, so wird sie vom Slave verworfen.
- Besteht auf dem Slave keine Verbindung und es wird eine Wiederholung einer Open-Indication empfangen, so ist diese für den Slave im Allgemeinen korrekt und wird mit einer (a) Open-Response beantwortet.
- Empfängt der Slave nun eine weitere wiederholte Datensendung aus einer alten Sequenz von Open-Indication und folgenden Datensendungen, so wird die Datensendung daran erkannt, dass ihr CRC nicht den bei (a) generierten mSlavePreset für die CRC-Berechnung verwendet hat, und wird verworfen.

**Master**

- Wird eine Open-Indication vom Master empfangen, so wird sie verworfen.
- Wird eine Open-Response Nachricht während einer bestehenden Verbindung empfangen, so wird sie vom Master verworfen.
- Besteht auf dem Master keine Verbindung, es wurde keine Open-Indication an den Slave verschickt und es wird eine Wiederholung einer Open-Response empfangen, so ist diese für den Master ungültig und wird verworfen.
- Besteht auf dem Master keine Verbindung, es wurde ein Open-Indication an den Slave verschickt und es wird eine Wiederholung einer Open-Response empfangen, so kann der Master am darin enthaltenen mMasterPreset erkennen, ob die Open-Indication zur der von ihm zuletzt verschickten Open-Indication passt.
- Eine darauf folgende, wiederholte Datensendung ist dann wegen der aus dem mMasterPreset zu bildenden Sequence-Number und dem unpassenden mMasterPreset für den CRC Berechnung ungültig und wird vom Master verworfen.

## 6.3 Nachrichtenverlust und Einfügungen von Nachrichten

Der Verlust einer Nachricht wird durch die Erwartungshaltung der strengen Monotonie der Sequence-Number oder an der Zeiterwartung für Nachrichten erkannt.

Betrachtet wird hierbei für TSP das Einfügen von Nachrichten während einer bestehenden Verbindung. Dabei sind die Fälle „Eingefügte Nachricht an den selben Empfänger“ und „Eingefügte Nachricht an andere Empfäner“ zu betrachten.

Der erste Fall tritt ein, wenn eine Nachricht, die zuvor einmal an den Knoten gerichtet wurde, nun vom Knoten empfangen wird.

a) Falls die Sequence-Number unverändert zu der zuvor vom Knoten empfangenen Nachricht ist, wird die Nachricht als Duplikat verworfen.

b) Falls die Sequence-Number (LSB) verändert zu der zuvor vom Knoten empfangenen Nachricht ist, so wird der CRC1 oder CRC2+3 über die erwartete Sequence-Number und die Nachricht berechnet. Eine Nachricht wird nicht als eingefügt erkannt, wenn die Sequence-Number zufällig passt oder wenn der CRC1 oder beide CRC2+3 zufällig passen. Die Wahrscheinlichkeit, dass sie zufällig passen ist

$2^{-32} = 2{,}328 \cdot 10^{-10}$. Die Wahrscheinlichkeit kann auch bei CRC2+3 nicht besser sein, da der Wertebereich der Sequence-Number dominant ist.

Setzt man die erwartete Fehlerrate[237] für das Wiederholen einer Nachricht mit $10^{-6}\ h^{-1}$ an, so ergibt sich eine Restfehlerwahrscheinlichkeit pro Stunde von

$$10^{-6} h^{-1} \bullet 2^{-32} = 2{,}328 \bullet 10^{-16} h^{-1} \leq 1 \bullet 10^{-9} h^{-1} = 1\% \ von \ SIL3 \qquad (6.5)$$

Damit erreicht die Maßnahme eine eingefügte Sequence-Number über den CRC1 oder über CRC2+CRC3 zu erkennen den Level SIL3. Dies gilt auch dann noch, wenn die angenommene Fehlerrate der Netzwerkkomponente auf 4 $h^{-1}$ ansteigt.

Das Einfügen von Nachrichten, die für andere Knoten bzw. Verbindungen erstellt wurden, wird im Kapitel 6.5 betrachtet.

Eine falsche Nachrichtenreihenfolge wird durch die Erwartungshaltung des Sequence-Number Low-Bits und der strengen Monotonie der 32-Bit Sequence-Number erkannt. Die Restfehlerwahrscheinlichkeit für die Behandlung falscher Nachrichtenreihenfolgen entspricht der Behandlung von eingefügten Nachrichten.

Die Fragmentierung von Nachrichten, d.h. das Zusammensetzen einer Nachricht aus Teilen, z.B. aus anderen Nachrichten, bzw. die Reihenfolgevertauschung von Teilen einer Nachricht, wird von TSP durch den CRC über die gesamte Nachricht erkannt.

TSP hat eine zeitliche Erwartungshaltung über die Abstände der korrekt empfangenen Nachrichten. Ist der zeitliche Abstand größer als diese Erwartung, so wird die Sicherheitsfunktion eingeleitet.

Drift bezüglich der für den Nachrichtentransport benötigten Zeit wird durch das Verfahren des Handshakes zwischen Master und Slave beherrscht. Da der Slave nur auf Anfragen des Masters antwortet, können aus Sicht des Masters keine unerkannten Drifts auftreten. In Folge produziert der Master auch keine Nachrichten, die zu einem Drift in Richtung Slave führen können, ohne dass dieser dies durch seine zeitliche Erwartungshaltung erkennt.

## 6.4 Reaktionszeiten

Bei den Reaktionszeiten wird hier die sicherheitstechnische Worst-Case Reaktionszeit ($t_{wc}$) betrachtet.

$\boldsymbol{t_{wc2}}$ ist dabei definiert als die maximale Zeit, die benötigt wird, um auf einen Signalwechsel am Eingang eines Knotens A, am Ausgang des anderen Knotens B zu reagieren und zwar auch dann, wenn Fehler auftreten. Fehler ist hier bezüglich der Kommunikation ausdrücklich im Plural verwendet.

Die Worst-Case Reaktionszeit mit einem Fehler wird mit $\boldsymbol{t_{wc1}}$ bezeichnet.

Betrachtet werden nachfolgend die Reaktionszeiten für eine Sicherheitsfunktion bestehend aus den Szenarien:

a) Input-Slave / Output-Master
b) Input-Slave / Master / Output-Slave

Die konkrete Reaktionszeitzeit hängt neben dem TSP auch von dem in der konkreten Implementation gewählten synchronen bzw. asynchronen Modellen und anderen Implementierungseigenschaften der Knoten (Master/Slave) ab.

Die Zykluszeit der jeweiligen Knoten wird durch *ZZ* abgekürzt und als Worst-Case Wert mit *WCZZ* angegeben.

Es wird angenommen, dass die *WCZZ* auch die maximale Zeit ist, die benötigt wird, um nach dem Ausfall eines Knotens, dessen Ausgänge in den sicheren Zustand zu versetzen.

---

[237] Annahme der Fehlerrate für einen Ethernet-Switch-Baustein mit 100 FIT (siehe auch [PROFIsafeV2]) Damit ein praxistauglicher Ansatz gewährleistet ist, werden 10 Ethernet-Switch-Bausteine angenommen.

### 6.4.1 Reaktionszeiten beim asynchronen Phasenkonzept

Beispielhaft wird hier von einem Slave-Modell und einem Master-Modell ausgegangen, das der obigen Beschreibung des asynchronen Phasenkonzepts aus Kapitel entspricht. Dabei dürfen der Input- und der Output-Slave auch identische Knoten sein und über eine oder zwei TSP Verbindungen zum Master agieren.

#### Input-Slave

Für den Input-Slave wird weiter unterstellt, dass er die TSP-Input-Verarbeitung (Empfang des OUTPUT_DATA_INDICATION) vor der Eingabewertermittlung (z.B. eines digitalen Eingangs) ausführt und in seiner Output-Phase die RESPONSE mit den zuvor ermittelten Eingabewerten versendet.

Hier nicht betrachtet werden die Verzögerungen, die sich durch die Beschaffenheit des Eingangs des Input-Slaves ergeben.

#### Master

Für den Master bedeutet dies, dass er in seiner Input-Phase die RESPONSE empfängt, bzw. eine empfangene erst in der Input-Phase bearbeitet (fehlerfreier Fall). Die darauf folgende Applikationsverarbeitung erhält diese empfangenen Daten und generiert darauf hin ein Ergebnis.

Fall a) Dieses Ergebnis wird vom Master in seiner Output-Phase an einem seiner Ausgänge signalisiert.

Fall b) Dieses Ergebnis wird vom Master in seiner Output-Phase mittels OUTPUT_DATA_INDICATION an einen Slave verschickt.

#### Output-Slave

Für den Output-Slave bedeutet dies, dass er in seiner Input-Phase die DATA_OUTPUT_INDICATION des Masters empfängt/verarbeitet und in seiner Output-Phase den Ausgangswert des Masters an seinem (physikalischen) Ausgang verfügbar macht und dann schlussendlich die RESPONSE als Antwort an den Master verschickt.

Hier nicht betrachtet werden die Verzögerungen, die sich durch die Beschaffenheit des Ausgangs des Output-Slaves ergeben.

#### Worst-Case-Reaktionszeiten asynchrones Phasenkonzept

Der zeitliche Abstand der beim Master empfangenen Slave Sendungen ist bis zu *WDTinput* groß. Damit kommt nur alle *WDTinput* ein neuer Wert und damit wird gemäß Abtasttheorem die Reaktionszeit für diesen Anteil der Gesamtreaktionsteil mit 2 • *WDTinput* berechnet. Weiterhin wird betrachtet, dass der Master zum spätest möglichen Zeitpunkt auf das Ablaufen der Zeitüberwachung für *WDTinput* reagiert (Fall a-1).

Fall a-1):

$$t_{wc1} = 2 \bullet WDTinput + 2 \bullet WCZZmaster \qquad (6.6)$$

Wird weiter angenommen, dass der Ausfall des Masters seine Ausgänge spätestens innerhalb von *WCZZmaster* absteuert und unmittelbar vor dem Ende von Fall a-1) ausfällt (Fall a-2).

Fall a-2):

$$t_{wc2} = 2 \bullet WDTinput + 2 \bullet WCZZmaster + WCZZmaster \qquad (6.7)$$

Dies beschreibt den ungünstigsten Fall, bei dem die Verbindung zum Input-Slave ausfällt, die Reaktion des Masters darauf 2 • *WCZZmaster* dauert, die Abschaltnachricht vom Master dann nicht unmittelbar an den Output-Slave geschickt werden kann, weil noch eine Antwort von diesem aussteht und schließlich der Output-Slave noch 2 Zyklen für die Reaktion auf die Nachricht benötigt (Fall b-1).

Fall b-1):

$$t_{wc1} = 2 \bullet WDTinput + 2 \bullet WCZZmaster + 2 \bullet WDToutput + 2 \bullet WCZZoutput \quad (6.8)$$

Wird weiter angenommen, dass der Output-Slave zum spätest möglichen Zeitpunkt ausfällt, dh. bevor er auf das Ausbleiben einer Master Nachricht reagieren würde, so ergibt sich Fall b-2).

Fall b-2):

$$t_{wc2} = 2 \bullet WDTinput + 2 \bullet WCZZmaster + 2 \bullet WDToutput + 2 \bullet WCZZoutput + WCZZoutput \quad (6.9)$$

### 6.4.2 Reaktionszeiten beim synchronen Phasenkonzept

Beispielhaft wird hier von einem synchronen Slave-Modell und einem synchronen Master-Modell ausgegangen. Beide, Master und Slave, implementieren ein zyklisches Ausführungsmodell ihrer Applikation, bestehend aus Input-Verarbeitung, Applikationsverarbeitung und Output-Verarbeitung.

#### Input-Slave

Für den Input-Slave wird weiter unterstellt, dass er die TSP-Input-Verarbeitung (Empfang des DATA_INPUT_INDICATION) vor der Eingabewertermittlung (z.B. eines digitalen Eingangs) ausführt und in seiner Output-Phase die DATA_RESPONSE mit den zuvor ermittelten Eingabewerten versendet.

Hier nicht betrachtet werden die Verzögerungen, die sich durch die Beschaffenheit des Eingangs des Input-Slaves oder anderweitiger Verarbeitungen ergeben.

#### Master

Für den Master bedeutet dies, dass er in seiner Input-Phase die DATA_INPUT_INDICATION an den Slave verschickt und eine dazu gehörige DATA_RESPONSE empfängt (fehlerfreier Fall). Die darauf folgende Applikationsverarbeitung bekommt diese empfangenen Daten und generiert darauf hin ein Ergebnis.

Fall a) Dieses Ergebnis wird vom Master in seiner Output-Phase an einem seiner lokalen Ausgänge signalisiert.

Fall b) Dieses Ergebnis wird vom Master in seiner Output-Phase mittels DATA_OUTPUT_INDICATION an einen Slave verschickt. Der Master wartet nun noch in der Output-Phase auf die dazu gehörige Antwort des Slaves.

Es wird davon ausgegangen, dass die Applikation eine Reaktion auf das Eingangsdatum direkt erzeugt und keine applikative Verzögerung eintritt.

#### Output-Slave

Für den Output-Slave bedeutet dies, dass er in seiner Input-Phase die DATA_OUTPUT_INDICATION des Masters empfängt/verarbeitet und in seiner Output-Phase den Ausgangswert des Masters an seinem physikalischen Ausgang verfügbar macht und dann in seiner Output-Phase schlussendlich die DATA_INDICATION als Antwort an den Master verschickt.

Hier nicht betrachtet werden die Verzögerungen, die sich durch die Beschaffenheit des Ausgangs des Output-Slaves ergeben.

### 6.4.3 Worst-Case-Reaktionszeiten synchrones Phasenkonzept

Für Szenario a-1) berechnet sich die Worst-Case Reaktionszeit mit:

$$t_{wc1} = 2 \bullet WCZZmaster = 2 \bullet WDTinput \quad (6.10)$$

und Szenario b-1) mit:

$$t_{wc1} = 2 \bullet WCZZmaster + 2 \bullet WCZZoutput = 2 \bullet WDToutput + 2 \bullet WCZZoutput \quad (6.11)$$

*Anmerkungen:*

- Die Response-Time-input und die Response-Time-output sind Bestandteil von WCZZmaster.
- Beim Master wird die *WDTinput*/*output* auf die Response-Time-input/output gesetzt.
- Der Output-Slave überwacht den Master mit der Zeit *WDToutput*, die hier gleich *WCZZmaster* gesetzt wird.
- Der Input-Slave überwacht den Master mit der Zeit *WDTinput*, die hier ebenfalls gleich *WCZZmaster* gesetzt wird. Diese Überwachung ist eigentlich sicherheitstechnisch optional.
- Durch die Response-Time-Überwachung des Masters, wird insbesondere für den Output-Slave überwacht, dass das Nachrichten-Delay vom Master Richtung Output-Slave kleiner als die Response-Time ist. Insbesondere ist dieses Nachrichten-Delay in *WCZZmaster* enthalten.
- Ein „Drift“ des Nachrichten-Delays kann damit ausgeschlossen werden, da der Master einen solchen erkennen und die Verbindung schließen würde.
- Aus Sicht des Output-Slaves betrachtet man für das Eintreten der Drift-Situation den Fall (i); dass gegenüber der zuletzt empfangenen Nachricht des Masters, die nächste Nachricht durch den Drift so verzögert wird, dass der zeitliche Abstand größer als *WDToutput* ist und damit der Output-Slave die Verbindung schließt. D.h. *WDToutput (+ 2•WCZZoutput)* nachdem die letzte zeitgerechte Nachricht beim Output-Slave empfangen wurde, ist die Sicherheitsreaktion vollendet.
- Für den Fall (ii) betrachtet man, dass der aktuelle Zyklus des Masters kleiner als *WCZZmaster* und kürzer als der letzte war, wodurch aus Sicht des Output-Slaves, trotz Drift, die Nachricht mit einem zeitlichen Abstand kleiner *WDToutput* zur letzten zeitgerechten Nachricht eintrifft. Diese wird vom Slave als gültig betrachtet. Der Slave gibt mit dieser Nachricht ein Datum aus, dass nach dem Empfang seiner letzten empfangenen Master-Nachricht gebildet wurde. Von daher handelt es sich um eine völlig zeitgerechte Reaktion. Nur der Master kann hierbei feststellen, dass der Drift aufgetreten ist (nicht jedoch, ob dieser beim Hin- und/oder Rückweg auftrat). Stellt nun der Master die Verbindung ein, so reagiert der Slave nach *WDToutput (+ 2•WCZZoutput)*, nachdem er die gerade völlig zeitgerechte Reaktion ausgeführt hat, dann mit der Sicherheitsreaktion.

Betrachtet man auch beim synchronem Modell den Ausfall des Masters, beziehungsweise des Output-Slaves zum ungünstigsten Zeitpunkt, so ergibt sich:

für Szenario a-2) eine Worst-Case2 Reaktionszeit von:

$$t_{wc2} = 2 \bullet WCZZmaster + WCCZZmaster \qquad (6.12)$$

und für Szenario b-2) eine Worst-Case2 Reaktionszeit von:

$$t_{wc2} = 2 \bullet WCZZmaster + 2 \bullet WCZZoutput + WCCZZoutput$$
$$= 2 \bullet WDToutput + 3 \bullet WCZZoutput \qquad (6.13)$$

### 6.4.4 Anmerkungen zur Worst-Case-Reaktionszeit

Im Gegensatz zu anderen Sicherheitsprotokollen wurde hier bei den Szenarien a-2) und b-2) keine 1 Fehler, sondern eine Mehr-Fehler-Betrachtung angestellt. Damit ist man bei Anwendungen nicht gezwungen, Common-Cause Fehler auszuschließen.

Will der Anwender von TSP dies dennoch tun, so obliegt es seiner eigenen Verantwortung. Bei den Reaktionszeiten wurde angenommen, dass keine Verzögerung innerhalb der Eingangswertermittlung oder bei der (physikalischen) Ausgabe der Werte auftritt. Diese beiden Zeiten wären gegebenenfalls einmal zu $t_{wc1/2}$ zu addieren.

Die Reaktionszeiten von Eingangsknoten über Verarbeitungsknoten bis zum Ausgangsknoten hängen maßgeblich von der Art der Implementierung ab. Daher ist es notwendig eine detaillierte Analyse durchzuführen, wenn man von den in den Zeitberechnungen vorgestellten Modellen abweicht.

Die implementierende Organisation sollte auf jeden Fall für das von ihr gewählte Modell und für ihre spezifischen Aspekte der Implementation die *WCRT* berechnen und in den sicherheitsgerichteten Funktionen einer Anlage berücksichtigen.

## 6.5 Falsche Adressierung der Verbindung (Authentizität)

Die Authentizität einer Nachricht wird durch die Connection-ID und den Event1/2 hergestellt. Die Connection-ID bestimmt eineindeutig die Verbindung zwischen zwei Knoten in einer Kommunikationsdomain. Anhand des Events des Typs Indication kann der Slave erkennen, dass die Nachricht vom für diese Connection passenden Master kommt und für ihn, den Slave, bestimmt ist.

In der umgekehrten Richtung erkennt der Master am Event des Typs Response, dass die Nachricht für diese Connection vom Slave kommt.

Der Einsatz von Routern und Gateways ist zugelassen, solange dafür Sorge getragen wird, dass die Vergabe der Connection-IDs in dem Netz, in dem sie transportiert werden können, eineindeutig ist.

Bei der Eineindeutigkeit ist darauf zu achten, dass diese für die Summe von TSP1 und TSP2 Verbindungen gelten muss.

## 6.6 Mischung von Standard und Safety-Nachrichten (Masquerading)

Im betrachteten Einsatzgebiet von TSP ist eine Mischung von Safety- und Standardnachrichten zugelassen. Somit ist ein Schutz zur Unterscheidung der beiden Klassen von Nachrichten erforderlich. Dazu verwendet TSP verschiedene Techniken an.

1. Bei der CRC Rechnung wird für Sicherheitsverbindungen ein spezieller Preset und die nicht übertragene Sequence-Number für den CRC-Algorithmus verwendet. Zudem werden CRCs verwendet, die in Standardnetzwerken nicht gebräuchlich sind.
2. Da der Preset, die Sequence-Number und die CRCs in den nicht sicherheitsgerichteten Komponenten nicht bekannt sind, ist nicht anzunehmen, dass diese eine korrekte Prüfsumme für eine sicherheitsgerichtete Nachricht erzeugen können. Gleichbedeutend mit der Verwendung eines anderen Presets, Sequence-Number bzw. CRCs, ist die zufällige Berechnung derselben Prüfsumme, wie dies durch den für den sicheren Verbindungstyp richtigen CRC erfolgen würde.

Damit eine TSP Nachricht von einer Standardkomponente kommend akzeptiert wird, müssen zusätzlich die Erwartungshaltung bezüglich Connection-ID, Event1/2 und Seq-No-LSB passend sein. Damit müssen in Summe 6 Bytes bei TSP1 und 12 Bytes bei TSP2 Nachrichten zufällig passen, damit trotz dieser Maßnahmen ein Masquerading nicht erkannt wird.

Für TSP1 ergibt sich eine Restfehlerwahrscheinlichkeit von

$$TSP1: \frac{1}{2^{(48)}} = 3{,}55 \bullet 10^{-15} \qquad (6.14)$$

Für TSP2 ergibt sich eine Restfehlerwahrscheinlichkeit

$$TSP2: \frac{1}{2^{(92)}} = 2{,}02 \bullet 10^{-28} \qquad (6.15)$$

Neben den Header-Feldern von TSP müsste auch die erwartete Nachrichtenlänge übereinstimmen. TSP1 erfüllt somit bis zu einer Nachrichtenrate von 78 Nachrichten je Sekunde mit korrekter Länge den Anspruch von 1% SIL3. TSP2 genügt bis zu einer Nachrichtenrate von $1{,}37 \bullet 10^{15}$ Nachrichten je Sekunde mit korrekter Länge dem Anspruch von 1% SIL3.

## 6.7 Kommunikationsfehler in offenen Kommunikationseinrichtungen

Beim Einsatzgebiet von TSP muss es sich um eine geschlossene Übertragungseinrichtung handeln, sofern diese ordnungsgemäß aufgebaut wurde. Da dies in den Bereich der vorhersehbaren Fehlbedienung beim Aufbau oder Umbau fällt, werden die Fehlermöglichkeiten offener Kommunikationseinrichtungen nachfolgend betrachtet.

Betrachtet werden hier Bedienungsmöglichkeiten die dazu führen könnten, dass die TSP Mechanismen ausgehebelt werden. Die Bedieneingriffe bei TSP beschränken sich im Wesentlichen auf den Aufbau der Kommunikationseinrichtungen, sowie deren Parametrierung.

Um diesbezügliche Schutzmaßnahmen zu erreichen, wird es dem Anwender von TSP empfohlen geeignete Zugangsmaßnahmen und Zugriffsschutz für die Knoten und deren Netzwerk zu realisieren. Dies ist jedoch außerhalb der Protokoll- technischen Betrachtungsebene von TSP.

Offene Übertragungssysteme haben, neben der Gefährdung des Zugriffs, auch die Eigenschaft nicht definieren zu können, welche anderen Protokolle benutzt werden.

Das Nachrichtenformat von TSP stellt keine spezifischen Anforderungen an ein Übertragungssystem. Daher muss damit gerechnet werden, dass das Übertragungssystem ein für Standardkomponenten ebenfalls genutztes Transportprotokoll und eine gemeinsame Übertragungstechnik nutzt.

Zur primären Unterscheidung nutzt TSP spezifische CRC32 Polynome und einen spezifischen Preset, die soweit bekannt, nicht für Standard-Kommunikationszwecke angewendet werden. Zudem wird der CRC über Daten berechnet, die nicht übertragen werden (Sequence-Number und Preset). Dies sollte mit den für das Masquerading beschriebenen Fähigkeiten von TSP für unbeabsichtigte Ereignisse ausreichend gegeben sein (s.a. Kapitel 6.6)

So wie im vorangegangenen Kapitel die versehentliche Mischung von Standard- und Safety-Nachrichten ausgeschlossen wurde, so ist sie doch bei absichtlicher Beeinflussung durchaus zu betrachten.

Da TSP nicht für offene Übertragungseinrichtungen ausgelegt wurde, gibt es keine spezifischen Mechanismen, die zum Zweck des Schutzes vor absichtlicher Unterminierung von Sicherheitsmechanismen des Protokolls, in den Funktionsumfang des Protokolls aufgenommen wurden.

Jedoch ist es trotzdem nicht einfach, z.B. eine Nachricht in das Übertragungssystem einzuschleusen, die vom Empfänger als korrekt akzeptiert wird. Voraussetzungen dazu sind die Kenntnis der Funktionsweise von TSP, der physikalischer Zugang in das Übertragungssystem, z.B. mit einem Mirroring-fähigen Switch und einen Kommunikationsknoten, der in der Lage ist den Nachrichtenverkehr in Echtzeit zu analysieren.

Insbesondere die Echtzeitfähigkeit des angreifenden Kommunikationsknotens stellt hohe technische, jedoch nicht unmögliche Anforderungen an diesen, da der Knoten eine Sequence-Number generieren muss, die dem aktuellen Erwartungsstand des Empfängers entspricht und gleichzeitig den Preset „erraten“ muss. Trifft der Kommunikationsknoten nicht das Zeitfenster, in der eine Sequence-Number gültig ist, wird die Nachricht vom Empfänger verworfen.

Eine Unterminierung der Sicherheitsmechanismen von TSP ist am wahrscheinlichsten durch den Einsatz einer TSP-Strategie umsetzenden Software. Damit dies Aussicht auf Erfolg hat, reicht es nicht aus die Funktionsweise von TSP zu kennen, es muss auch der eingesetzte Gerätetyp bekannt sein bzw., bei frei konfigurierbaren Systemen, ihre Konfiguration.

Dieses Szenario im eigentlichen Sinne keine Anforderung an TSP, sondern an den Zugriffsschutz des Systems als Ganzes. Hierbei ist einerseits der physikalische Zugriffsschutz der sicherheitsgerichteten Übertragungseinrichtungen, aber auch der über externe Übertragungseinrichtungen zu betrachten.

Der Zugriffsschutz des Systems als Ganzes spielt auch bei der absichtlichen fehlerhaften Konfiguration eine entscheidende Rolle.

Eher als die Unterminierung der Sicherheitsmechanismen, ist die Sabotage der Funktion als Ganzes. Dafür in Frage kommen Techniken für den Denial-of-Service Angriff auf ein oder mehrere Geräte im System durch die Generierung von Überlast mittels vieler Nachrichten pro Zeiteinheit.

## 6.8 Erweitere Fehlerbetrachtung für sichere Systeme

Neben den Fehlern, die in Kommunikationseinrichtungen auftreten können, werden nun auch die Gefahren, die sich für TSP durch die Umgebung und den Einsatz ergeben, betrachtet.

### 6.8.1 Fehlerhafte Konfiguration

Bei TSP ist der Anwender dafür verantwortlich eineindeutige Connection-IDs und dazu passende Master/Slave Rolle zu parametrieren.

Falls der Anwender „nur“ die Master-Slave Rolle eines Knotens für eine Verbindung verwechselt, kommt die Verbindung nicht zu Stande und bleibt im sicheren Zustand.

Falls der Anwender eine Connection-ID doppelt vergibt, so muss im Allgemeinen auch noch die Adresse des unterlagerten Transportsystems doppelt vergeben werden, um eine falsche Adressierung zu erreichen. Wenn ein Knoten mehr als einmal die gleiche Connection-ID parametriert bekommt, sollte er dies ablehnen und im sicheren Zustand bleiben.

Eine Gefährdung tritt im Allgemeinen dann ein, wenn die Connection-ID beim falschen Gerät eingestellt wird, z.B. indem zwei oder mehr Geräte gleichen Typs vertauscht wurden. Dies muss der Anwender bei der Inbetriebnahme daher sorgfältig prüfen. Diese Maßnahme liegt außerhalb des Protokoll- technischen Bereichs von TSP.

Bei einer Verzögerung/Unterbrechung einer TSP-Kommunikation kann eine Gefährdung eintreten, wenn die parametrierte mWDT für den zu steuernden/überwachenden Prozess unzulässig groß ist und daher im Fehlerfall eine Sicherheitsreaktion nicht zeitgerecht erfolgt. Dies ist vom Anwender zu prüfen. Hierbei ist es jedoch nicht ausreichend Messungen der Reaktionszeit vorzunehmen, da auf diese Weise im Allgemeinen nicht die Worst-Case-Zeiten ermittelt werden können.

Die Betrachtung der inhaltlich korrekten mConfig-Daten liegt außerhalb des Bereichs von TSP. Wohl von TSP unterstützt wird die Prüfung der „passenden“ Konfiguration zwischen Master und Slave. Eine solche kann von TSP erkannt werden und führt dazu, dass der sichere Zustand nicht verlassen wird.

Die Möglichkeiten der fehlerhaften Konfiguration sind durch organisatorische Maßnahmen durch den Anwender zu beherrschen.

### 6.8.2 Absehbarer Missbrauch und Fehlbedienung

Ein Missbrauch der Verwendung von TSP ist insbesondere durch eine absichtliche fehlerhafte Konfiguration vorstellbar.

Möglichkeiten das Protokoll TSP im vorgesehenen Einsatzgebiet zu umgehen, hat der Anwender nicht, da ihm im Allgemeinen keine Möglichkeit geboten wird, auf diesen Teil der Funktion des Systems oder des Geräts Einfluss zu nehmen.

Ausgenommen hiervon sind solche Systeme/Geräte, die die Kommunikationsfunktion auch in einer nicht sicheren Variante zur Verfügung stellen. Hier hat der Anwender die Möglichkeit die Sicherheit auszuschalten. Dies sollte durch entsprechende organisatorische Maßnahmen durch den Anwender beherrscht werden und liegt außerhalb des Einflusses von TSP.

Absehbare Fehlbedienung ist vorstellbar durch:

1. Verwendung nicht eineindeutiger Adressen bzw. durch Hinzufügen solcher Geräte in ein Übertragungssystem.
2. Zusammenschalten von zwei oder mehr Systemen/Teilsystemen mit identischen Adressen, wodurch ein Gesamtsystem mit nicht eineindeutigen Adressen gebildet wird.

Diese Fälle werden von TSP nicht absichtlich erkannt. Einen gewissen Schutz gibt es, dass eine bestehende TSP-Verbindung a) keine Verbindungsaufnahme ermöglicht und b) eine Nachricht aus einer anderen Verbindung (mit gleicher Connection-ID) auch noch die gleiche Transportsystemadresse haben muss und dann auch noch zufällig den passenden CRC zur unbekannten Sequence-Number haben muss.

Dennoch obliegt es auch hier letztlich dem Anwender, derartige Szenarien zu verhindern.

## 6.9 Verbleibende Anforderungen an den Anwender

Zusammenfassend verbleiben für den Anwender, neben allgemeinen Anforderungen, die aus dem Einsatz von sicherheitsgerichteten Anwendungen herrühren, dafür zu sorgen, dass:

1. der physikalische Zugriffsschutz auf die sicherheitsgerichtete Übertragungseinrichtung gewährleistet ist,
2. der externe (Netzwerk) Zugriffsschutz die Mittel des Systems, nur berechtigten Personen Zugriff zu gewähren, einsetzt,
3. das externe Netzwerk ausreichend geschützt ist, sodass von außen keine Angriffe und oder Fehlbedienungen auf dieses erfolgen können,
4. die Überwachungszeiten WDT passend zur sicherheitsgerichteten Anwendung parametriert sind.

Die Entwicklung und Verifikation von TSP ist für die Qualifizierung nach IEC 61508 / SIL3 im Rahmen einer IEC 61508 / SIL3 konformen Produktentwicklung vorzunehmen und in einem Knoten einzusetzen, der aus HW- und SW-technischer Sicht ebenso entwickelt wurde. Das erfolgreiche Konzept-Approval gemäß IEC 61608 / SIL3 erfolgte im August 2010 durch den TÜV Rheinland[238].

[238] [TÜV10]

# 7 Betrachtung und Bewertung der Ergebnisse

In bisherigen sicherheitsgerichteten Protokollen und den relevanten Normen fanden die Gefährdungspotentiale

- Masquerading und Adressierung in offenen Übertragungssystemen
- Absichtliche Unterminierung der Sicherheitsmechanismen
- Fehlerhafte Konfiguration

kaum Beachtung und die Gefährdungen durch

- Absehbaren Missbrauch
- Absehbare Fehlbedienung
- Unberechtigter Zugriff auf sichere Kommunikationseinrichtungen

wurde nur in Randgebieten diskutiert. Hier zeigt die vorliegende Arbeit die Bedeutung dieser für Einsatz von sicherheitsgerichteten Kommunikationsprotokollen im industriellen Umfeld auf und folgert daraus die entsprechenden Maßnahmen, die in der Verantwortung des Anwenders liegen, bzw. zum Teil auch durch Protokollmechanismen beherrscht werden können.

Die Arbeit stellt erstmals einen umfassenden Gefährdungskatalog auf und bewertet nach diesem einheitlichen Katalog die am weitest verbreiteten sicherheitsgerichteten Kommunikationsprotokolle nach einem einheitlichen Maßstab. Weiter zeigt die vorgelegte Arbeit auf, dass auch ein existierendes Zertifikat gemäß IEC 61508/SIL3 nicht in jedem Fall ausreichend ist, um die Eignung eines Protokolls für den Einsatz gemäß dem aktuellen Stand der Normen zu bestätigen.

Hervor zu heben ist insbesondere die quantitative Bewertung jeder einzelnen Maßnahme der Protokolle. Bislang wurde diese nur für die Beherrschung von verfälschten Nachrichten durchgeführt. Die Arbeit führt diese quantitative Bewertung der eingesetzten Maßnahmen erstmalig systematisch durch und zeigt dabei, dass diese Bewertung dringend erforderlich ist, da die betrachteten öffentlichen Protokolle nicht die für SIL3 notwendige Güte für alle ihre Maßnahmen aufweisen.

Insbesondere in den ersten Versionen der jeweiligen Protokolle treten bei quantitativer Analyse oft Probleme zu Tage. So verwendete zum Beispiel PROFIsafe V1 anfangs einen CRC16 und einen Monotonie-Zähler, der nicht CRC geschützt war[239]. CIP-Safety Edition 1.2 ging ebenfalls mit CRC16/CRC8 an den Start[240]. Beide Varianten konnten den später aufgestellten Anforderung für die Nutzung des Black-Channel Modells nicht genügen und mussten nachgebessert werden.

Bezüglich der Beherrschung einer Bitfehlerrate von $10^{-2}$ tun sich diese Protokolle auch in der aktuellen Version noch sehr schwer, beziehungsweise erreichen dies nicht mit einer Qualität, die 1% von SIL3 erfordert[241].

Erst bei recht massiven, Ressourcen intensiven Maßnahmen kommen sicherheitsgerichtete Protokolle mit den Anforderungen für die Bitfehlerrate $10^{-2}$ zurecht.

Eine weitere recht verbreitete Schwäche der Protokolle der 1. Generation ist die Beherrschung von eingefügten Nachrichten mit einer für SIL3 geeigneten Restfehlerrate. So verwendeten PROFIsafe V1, FF-SIF und CIP-Safety Edition 1.2 alle einen 16 Bit Zähler (*Consecutive Number / Time-Stamps / MCN*)[242]. Diese Zähler sind erst dann geeignet, wenn eine sehr geringe Auftrittswahrscheinlichkeit für den Fehler angenommen werden kann.

---

[239] [PROFIsafeV1]

[240] [CIP5]

[241] gefordert in [IEC 61784-3] und [IEC 61508-2], nicht erreicht von [CIP5-2.2], [PROFIsafeV2]

[242] [PROFIsafeV1], [CIP5], [FF-SIF]

Dies führte in den verbesserten Versionen der Protokolle zur Verwendung von 24- bzw. 32-Bit Zählern[243, 244].

## 7.1 Worst-Case Reaktionszeit

Einer der Schwerpunkte dieser Arbeit ist die Definition von Verarbeitungsmodellen für die Berechnung der maximalen Reaktionszeit und der Worst-Case Reaktionszeit. Dazu wurde ein Modell aus 5 Komponenten erstellt, dass geeignet ist, die Reaktionszeit über ein Kommunikationssystem für die im industriellen Umfeld gebräuchlichen Anwendungen zu untersuchen. Diese 5 Komponenten, das Eingangsmodul, die Sicherheitssteuerung, die Kommunikation zwischen beiden, sowie das Ausgangsmodul und die Kommunikation zwischen diesem und der Sicherheitssteuerung.

Anhand dieses Modells wurde die maximale Reaktionszeit definiert. Dies ist die Zeit, die von der Änderung eines physikalischen Eingangssignals der Eingangskomponente, bis zur zugehörigen Reaktion des physikalischen Ausgangssignals der Ausgangskomponente, über die Sicherheitssteuerung hinweg, benötigt. Die maximale Reaktionszeit betrachtet dabei den Fall, dass im gesamten System aus diesen 5 Komponenten kein Fehler vorliegt.

Da für den sicherheitsgerichteten Einsatz die maximale Reaktionszeit mit Fehlern die relevante Betrachtungseinheit darstellt, wurde anhand des Modells die Worst-Case 1 Reaktionszeit definiert. Diese definiert die Zeit die eine Reaktion maximal benötigt, wenn im Kommunikationssystem ein Fehler vorliegt.

Da nicht ausgeschlossen werden kann, dass sich ein Fehler nur an einer Stelle auswirkt, beziehungsweise die Wahrscheinlichkeit für das Eintreten von unterschiedlichen Fehlern in der zu betrachtenden Zeitspanne ausreichend hoch sein kann, definiert diese Arbeit im Gegensatz zur Protokolldefinitionen wie PROFIsafe und CIP-Safety[243] anhand des obigen 5-Komponentenmodells die Worst-Case 2 Reaktionszeit. Diese berücksichtigt den Fall, dass mehrere Fehler eintreten können. Dies ist deckt dann insbesondere den Fall von Fehlern gemeinsamer Ursache ab, die in einem realen Kommunikationssystem leicht übersehen werden. Für die dermaßen definierte Worst-Case 2 Reaktionszeit stellt diese Arbeit den Bezug zur „safety function response time“ (SFRT) der IEC 61784-3[245] her, die dort präziser und unmissverständlicher definiert sein könnte.

Weiter zeigt die Arbeit, dass die Worst-Case Reaktionszeit nur durch die Betrachtung der Mechanismen der Kommunikationsprotokolle nicht ermittelt werden kann und stellt daher ein Modellprotokoll auf. Viele der gängigen sicherheitsgerichteten Kommunikationsprotokolle lassen sich bezüglich der Worst-Case Reaktionszeit auf dieses Modellprotokoll abbilden. Weiter werden im Rahmen des Modellprotokolls synchrone und asynchrone Verarbeitungsmodelle samt möglicher Implementierungsvarianten untersucht. Für die weit verbreitete Kombination der synchroner Kopplung und Verarbeitung in den Phasen Input, Logik und Output, werden die sich daraus ergebenden Reaktionszeiten analysiert.

Dies Analyse geschah, neben dem fehlerfreien Fall, auch für Einfach- und Mehrfachfehler. Für Mehrfachfehler wurde, entgegen den Annahmen z.B. bei PROFIsafe, ausgeführt, dass durch eine gemeinsame Ursache[246], mehr als eine logische Komponente einer Sicherheitsfunktion von einem Fehler betroffen sein kann.

Zur Bestimmung der Worst-Case Reaktionszeit wurden verschiedene Fehlerszenarien analysiert. Dies umfasste das Ablaufen der Überwachungszeiten des Modellprotokolls der beiden Kommunikations-

[243] [PROFIsafeV2], [CIP5-2.2]

[244] Für FF-SIF liegt noch keine offizielle Spezifikation dazu vor.

[245] [IEC 61784-3]

[246] sogenannte Common Cause Fehler

verbindungen; die zwischen Eingangskomponente und Sicherheitssteuerung und zwischen Sicherheitssteuerung und Ausgangskomponente. Weiter wurde der Ausfall der Ein-, Ausgangskomponente und Sicherheitssteuerung zum jeweils für die Worst-Case Reaktionszeit ungünstigsten Zeitpunkt analysiert. Ferner wurde der Fall, Unterbrechung des Nachrichtenflusses zwischen den Komponenten, jeweils je Richtung getrennt betrachtet.

Dabei wurde ein bis dahin nicht bekannter Faktor $t_u$ ermittelt, der die Ungenauigkeit der Überwachungszeit beschreibt und in die Berechnung der Worst-Case Reaktionszeit eingeht.

Als Ergebnis hat sich ergeben, dass die Worst-Case Reaktionszeit einer Kommunikationsverbindung, im Gegensatz zu den Berechnungen der Protokolle PROFIsafe und CIP-Safety, mindesten die doppelte Überwachungszeit ist.

Für die Kopplung von 2 Kommunikationsverbindungen wird aufgezeigt, dass es auf der Ausgabeseite der Sicherheitssteuerung zu einer Blockade von einer Überwachungszeit kommen kann und somit auch dort mit einer Worst-Case Reaktionszeit von mindestens der doppelten Überwachungszeit zu rechnen ist. Daraus folgert die Arbeit, dass für die gesamte Sicherheitsfunktion, betrachtet man nur die Kommunikationsverbindungen, eine Worst-Case Reaktionszeit von mindestens der 4-fachen Überwachungszeit an zusetzen ist. Dies gilt insbesondere auch dann, wenn in den betrachteten 5 sicherheitsgerichteten Komponenten kein Fehler vorliegt.

Damit konnte gezeigt werden, dass die bisherigen Berechnungen von verbreiteten Sicherheitsprotokollen, wie z.B. PROFIsafe[248], nicht in allen Fällen korrekt sind.

Die Worst-Case Reaktionszeiten der Protokolle sind auf Grund der verschiedenen Konzepte sehr differenziert zu betrachten. Erschwerend kommt hier noch hinzu, dass die im konkreten System erreichte minimale Worst-Case Reaktionszeit stark von der Implementierung des jeweiligen Herstellers abhängt. Ebenso problematisch ist die unterschiedliche Modellbildung der jeweiligen Protokoll-Organisationen. Es werden Fehlerausschlüsse gemacht[247], wie die Annahme, dass nur ein Fehler auftritt und dieser Fehler sich nicht auf die Verbindung zwischen Input-Knoten und Sicherheitssteuerung und auf die Verbindung zwischen Sicherheitssteuerung und Output-Knoten gleichzeitig auswirken kann[248].

Der Anwender eines Systems mit sicherheitsgerichteter Kommunikation kann zu allem Überfluss die Worst-Case Reaktionszeit selbst nicht überprüfen. Dazu fehlen ihm die technischen Möglichkeiten, z.B. das Erzeugen einer Kommunikationsunterbrechung zum ungünstigsten Zeitpunkt oder dem Erzeugen einer schleichend erhöhten Latenzzeit, in ein Gesamtsystem einzuschleusen.

## 7.2 Ergebnisse zu PROFIsafe

Für PROFIsafe wird aufgezeigt, dass die Restfehlerwahrscheinlichkeit für den Verfälschungsschutz durch den 24-Bit CRC für kurze PROFIsafe Nachrichten für eine Bitfehlerrate von $10^{-3}$ das Level von 1% für SIL1 gemäß IEC 61508 nicht erreicht. Das geforderte Level von 1 % von SIL3 wird mit dem gewählten CRC erst ab einer Bitfehlerrate von $10^{-4}$ erreicht.

Weiter wird für lange PROFIsafe Nachrichten gezeigt, dass die Restfehlerwahrscheinlichkeit für den Verfälschungsschutz des dort eingesetzten 32-Bit CRCs bei einer Bitfehlerrate von $10^{-3}$ 1% von SIL1 nicht erreicht und ab einer Bitfehlerrate von $10^{-4}$ ebenfalls 1% von SIL3 gemäß IEC 61508 erreicht.

Der Verfälschungsschutz der beiden CRCs ist bei einer Bitfehlerrate von $10^{-2}$ unterhalb der für SIL1 geforderten Qualität. Der von PROFIsafe eingesetzte Mechanismus des SIL-Monitors ist nur begrenzt wirksam. So stellt diese Arbeit fest, dass er nicht geeignet ist, um ein System in Betrieb zu setzen, da

[247] jedoch nicht dokumentiert

[248] [PROFIsafeV2], [CIP5-2.2]

er erst nach einer Überwachungszeit von 8 Stunden[249] in der Lage ist eine Bewertung der Übertragungsgüte vornehmen zu können. Eine von Anfang an wirksame Überwachung wäre erst ab einer Bitfehlerrate von $10^{-4}$ gegeben, wobei ein SIL-Monitor dann jedoch gar nicht notwendig wäre. Die Anfangsproblematik wurde der PROFIBUS Nutzerorganisation in einer Sitzung des Arbeitskreises PROFIsafe mitgeteilt, fand aber bislang keinen Einzug in die PROFIsafe Spezifikation.

Weiter stellt die Arbeit fest, dass für den SIL-Monitor unterstellt werden muss, dass ein kontinuierliches Ansteigen der Bitfehlerrate gegeben sein muss, was in realen Umgebungen nicht zutrifft. Einerseits kann die Bitfehlerrate Fehler-bedingt sprunghaft ansteigen und andererseits stellen Nachrichten diskrete Ereignisse dar, die der kontinuierlichen Betrachtungsweise widersprechen. Als Folge ist es möglich, dass Fehler, die sich durch eine sprunghafte Erhöhung der Bitfehlerrate auswirken, nicht mit einer Qualität von 1% von SIL3 innerhalb der Überwachungszeit erkannt werden.

Da der Anwender den erforderlichen Grad der Fehlerfreiheit bei der Inbetriebsetzung des Kommunikationsnetzes, z.B. zum morgendlichen Schichtbeginn, nicht sicherstellen kann, ist es ihm nicht möglich die geeigneten Voraussetzungen zu schaffen. Verschärft wird diese Problematik, wenn es tatsächlich zu Abschaltung in Folge von Nachrichtenverfälschungen gekommen ist. Hierbei hat der Anwender in Folge die Verantwortung die ordnungsgemäße Wiederherstellung des Kommunikationssystems zu bestätigen. Dass er bei transienten umgebungsbedingten Störungen, z.B. Beeinflussung durch elektromagnetische oder elektrostatische Effekte, fast keine Möglichkeit hat einen Fehler zu finden, macht die Widersinnigkeit der Strategie deutlich und wird den Anwender auf Grund wirtschaftlicher Zwänge zur Quittierung nötigen können.

Eine weitere von diese Arbeit aufgezeigter Schwachstelle von PROFIsafe wurde bei der Betrachtung falscher Nachrichtenreihenfolgen herausgearbeitet. Da PROFIsafe eine Reihenfolgevertauschung bei unveränderter *vconsnr* nicht erkennt, jedoch die Aktualisierung der Prozessdaten zulässt, wodurch veraltete Daten unbeabsichtigt lange verwendet werden.

Bei der Betrachtung der Worst-Case Reaktionszeit von PROFIsafe zeigt die Arbeit, dass PROFIsafe bei der Betrachtung der SFRT von einem grundlegend falschen Modell für die Kommunikationsstrecke ausgeht.

Ein von dieser Arbeit erkannter Aspekt ist, dass durch die Eigenschaften des nicht sicherheitsgerichteten Anteils des Kommunikationssystems die Latenzzeit von Nachrichten erhöht werden kann, ohne dass dies zu einer Sicherheitsreaktion von PROFIsafe führt und in diesem Fall das Reaktionszeit-Modell von PROFIsafe eine zu kleine Worst-Case Reaktionszeit für die Kommunikationsstrecke bestimmt. Dies geschieht deshalb, weil PROFIsafe nur einen Fehler in der Strecke zwischen Eingangskomponente und Sicherheitssteuerung oder zwischen Sicherheitssteuerung und Ausgangskomponente unterstellt. Die Annahme, dass zwei derartige Fehler unwahrscheinlich sind, wird von der Arbeit jedoch als falsch aufgezeigt, da die oben genannten Fehler sicherheitstechnisch nicht erkannt werden ( → keine Sicherheitsreaktion) und somit eine gegenüber der Sicherheitszeit der Sicherheitsfunktion sehr große Zeitspanne zu betrachten ist. Zusätzlich ist die Erhöhung der Latenz, ohne dass diese von den Sicherheitsmaßnahmen erkannt wird[250], als Fehler gemeinsamer Ursache zu sehen, da für die Kommunikationsverbindungen zwischen Eingangs- und Ausgabekomponente zur Sicherheitssteuerung ein und die selbe physikalische Netzwerkinfrastruktur verwendet werden können[251]. Von dieser Arbeit aufgezeigt und von PROFIsafe ebenfalls nicht betrachtet werden Fehlerszenarien, in denen sich Fehler auf beide Kommunikationsstrecken auswirken. Dies können insbesondere Wirkungen auf Grund von Fehlern gemeinsamer Ursache sein. So ist zu unterstellen, dass z.B. einer Überlastung

[249] priv. Dis.

[250] keine Verletzung der Überwachungszeiten

[251] der Normalfall

des Übertragungssystems zu einer wellen-artigen, d.h. zeitlich verschobenen Ausbreitung von verzögerten Nachrichten oder verloren gegangenen Nachrichten führen kann.

Dies kann im schlechtesten Fall dazu führen, dass die Worst-Case Reaktionszeit der Kommunikationsstrecke wesentlich größer, als der von SFRT der PROFIsafe Spezifikation[252] berechnete Wert ist.

Auch für den Bereich der sicherheitsgerichteten Adressierung wird hier erstmals eine entscheidende Gefahrenstelle für das PROFIsafe Protokoll aufgezeigt. Das verwendete Adressierungsverfahren führt dazu, dass in realen Anlagen mit einer Wahrscheinlichkeit von z.B. 10%, eine sichere Zuordnung durch die Mechanismen von PROFIsafe von Konfigurationsdaten und Kommunikationsdaten nicht mehr gegeben sein kann. Hier verbleibt die Betrachtung der nicht sicherheitsgerichteten Techniken, die jedoch bei einem Black-Channel Modell eigentlich nicht angewendet werden dürfen.

Es gibt lediglich einen Sonderfall, bei dem dieser Fehler nicht zum Tragen kommt, da der CRC-Preset aller Ein- und Ausgangskomponenten immer unterschiedlich ist. Dazu darf im Netzwerk nur eine Sicherheitssteuerung definiert sein und die Parameter der Ein- und Ausgangskomponenten dürfen sich nur in ihrer Adresse unterscheiden. Alle anderen Parameter müssen bei allen Verbindungen zu den Ein- und Ausgangskomponenten gleich sein. Da sich in diesem Fall nur die 16 Bit Adressen in den Parametern unterscheiden, berechnet ein 16-Bit CRC immer unterschiedliche Ergebnisse, d.h. einen unterschiedlichen CRC-Preset.

Diese Arbeit zeigt weiter, dass das patentrechtlich geschützte Adressierungsverfahren[253] in Verbindung mit der Definition des Nachrichtenrahmens ebenfalls dazu führt, dass die „Adresse“, d.h. der CRC-Preset für Nachrichten von Sicherheitssteuerung an Ein-/Ausgangskomponente und von dieser zurück zur Sicherheitssteuerung, gleich ist. Damit kann die Sicherheitssteuerung eine von ihr generierte Nachricht nicht von der einer Komponente generierten Nachricht unterscheiden, wenn die Datenlänge für Eingangsdaten und Ausgangsdaten gleich ist. In diesem Fall ist die Nachricht der Sicherheitssteuerung eine gültige Antwort der Komponente. Die Konsequenz ist, dass die Sicherheitssteuerung falsche Eingangsdaten benutzt und dass die Überwachungszeit von der PROFIsafe Sicherheitssteuerung nicht abläuft, obwohl keine Nachricht vom der Ein-/Ausgangskomponente bei der Sicherheitssteuerung eintrifft.

Die Schwächen der Adressierung von PROFIsafe wurden der PROFIBUS Nutzerorganisation in einer Sitzung des Arbeitskreises PROFIsafe dargelegt. Dadurch wurde der Arbeitskreis veranlasst einen Entwurf für eine neue Version der PROFIsafe Spezifikation zu erstellen, die diese Schwachstellen für neue Protokollversionen abstellt.

## 7.3 Ergebnisse zu CIP-Safety

Die sicherheitstechnisch Analyse dieser Arbeit hat ergeben, dass die Nachrichtenformate für kurze und lange CIP-Safety Edition.1.1[254] Nutzdatennachrichten nur bei einer Bitfehlerrate von $10^{-3}$ die notwendige Qualität von 1% von SIL3 gemäß IEC 61508 erreichen. Bei einer Bitfehlerrate von $10^{-2}$ ist dies nicht mehr der Fall und man erreicht nicht einmal die Qualität für 1% von SIL1. Auch für die anderen Nachrichtenarten, Time-Stamp, Time-Correction und Time-Coordination wird aufgezeigt, dass 1% von SIL3 bei einer Bitfehlerrate von $10^{-2}$ nicht erreicht wird. Dies ist selbst bei der von Rockwell Automation ursprüngliche angesetzten Bitfehlerrate von $10^{-3}$ nicht der Fall.

Auf Grund der Schwäche der CRC Mechanismen von CIP-Safety Edition 1.1 zeigte dieser Arbeit eine Fragmentierungsproblematik auf. Diese kann dazu führen, dass zwar die Aktualität der empfangenen

[252] [PROFIsafeV2]

[253] [Bart99b]

[254] [CIP5]

Nachrichten vom Empfänger als gegeben angesehen wurden, die Nutzdaten aber in Wirklichkeit ein beliebiges Alter aufweisen konnten. Damit wäre jegliche zeitgerechte Reaktion hinfällig gewesen.

Die Arbeit zeigt eine weitere Schwachstelle des Time-Stamp Verfahrens von CIP-Safety Edition 1.1[254] auf. Auf Grund des kleinen 16-Bit Wertebereichs, in Kombination mit dem zulässigen Intervall der Erwartungshaltung (EPI), wurde herausgestellt, dass die Wahrscheinlichkeit für das nicht Erkennen einer eingefügten Nachricht bei 0,6914 liegen kann; ein für SIL3 völlig ungeeigneter Zustand.

Diese Umstände wurden der im Rahmen der ODVA federführend am CIP-Safety Protokoll arbeitenden Rockwell Automation dargelegt.

Zur Abhilfe der Schwächen von CIP-Safety Edition 1.1 stellt diese Arbeit verschiedene Lösungen vor. Ein Lösungskonzept besteht darin, dass das CRC Sicherungsverfahren die Nutzdaten und den Time-Stamp gemeinsam enthält und es somit erreicht, dass eine fehlerhafte Fragmentierung an der Grenze zwischen den beiden Datentypen erkannt wird.

Das zweite Lösungskonzept besteht in der Definition eines 32-Bit Time-Stamps, der zwecks optimierter, das heißt reduzierter Nachrichtenlänge, nicht vollständig übertragen wird und der nicht übertragene Anteil durch die passende Erwartungshaltung und die Kodierung in den CRC realisiert wird. Mit dieser Strategie konnte das Problem der Einfügung von Nachricht gelöst werden.

Die Lösungskonzepte wurden Rockwell Automation vorgestellt. Diese übernahmen die ebenfalls aus dieser Arbeit hervorgegangenen Nachrichtenrahmen auf Grund der gewünschten Nähe zur Edition 1.1 nicht in die Edition 2.2 des CIP-Safety Protokolls. Die im Rahmen dieser Arbeit erstellten Konzepte zum Schutz gegen Fragmentierungsfehler und das verbesserte Time-Stamp Handling wurde von Rockwell Automation und der ODVA hingegen in die Spezifikation des CIP-Safety Protokolls Edition 2.2 übernommen.

## 7.4 Ergebnisse zu FF-SIF

Die Analysen zur Reaktionszeit von FF-SIF in dieser Arbeit haben ergeben, dass die Mechanismen Stale-Count-Limit und STALE_DATA_Time, die die hinreichende Aktualität eines empfangenen Datums und das unzulässig lange Ausbleiben einer Nachricht beherrschen sollen, gegenüber den Fehlerannahmen Queuing und Drift nicht ausreichend wirksam sind.

Dazu wurde im Rahmen dieser Forschungstätigkeiten ein Konzept erarbeitet, dass die Rückkopplung des FF-SIF-DO Blocks mit dem Logic-Solver definiert. Das Verfahren besteht darin, dass sich der Logic-Solver bei der Ausgabekomponente, z.B. dem FF-SIF-DO-Block, subscibed. Mit den durch die Subscription ermittelten Nachrichten erhält der Logic-Solver den von der Ausgabekomponente genutzten Zeitstempel (MCN) und kann diesen mit seinem Zeitstempel des aktuellen Macro-Zyklusses vergleichen. Die Gleichheit bestätigt das zeitgerechte Arbeiten der Ausgangskomponente. Bei einer Abweichung publiziert der Logic-Solver seinerseits sichere Ausgangsdaten, was zu einer beschleunigten Fehlerreaktion führt, als wenn der Logic-Solver nur seine Kommunikation, das heißt das Publizieren einstellen würde. Das Konzept wurde der Fieldbus-Foundation mitgeteilt und von ihr in die neue Version der FF-SIF Spezifikation[255] übernommen.

Die Untersuchungen von FF-SIF im Rahmen dieser Arbeit ermittelten eine weitere Schwachstelle des FF-SIF Protokolls. Da die Basis der Zeitsynchronisation durch nicht sicherheitsgerichtete Komponenten realisiert wird, ist es möglich, dass diese einen Offset in der Zeit erzeugen, der durch die sicherheitsgerichteten Komponenten nicht erkannt werden kann. Zur Beherrschung der Fehlerannahme Zeitoffset wurde hier ebenfalls ein Konzept erarbeitet, dass die Bindung der Zeit an den Logic-Solver definiert.

Die hier ebenfalls ausgeführte Betrachtung der Maßnahmen von FF-SIF, eingefügte Nachrichten zu erkennen, hat ergeben, dass dies nicht mit der für SIL3 geforderten Qualität geschieht, da nur ein 16-

[255] [FF-SIF]

Bit Zeitstempel Anwendung findet. Dabei wurde ebenfalls aufgezeigt, dass der verwendete Black-Channel Ansatz sich nicht nur auf die Kommunikationsleitung beziehen darf, sondern auch die Verbindung zwischen Standard FF-H1-Stack und Sicherheitssteuerung betrachtet werden muss.

Zur Beherrschung beider Aspekte definiert die Arbeit einen Mechanismus, der darin besteht, dass die Quelle des MCN durch den Logic-Solver realisiert wird und ferner die DATA_LINK_TIME, die zusammen mit der MCN den eigentlichen Zeitstempel repräsentiert, mit in die CRC Berechnung der FF-SIF Nachrichten eingeht. Die Restfehlerwahrscheinlichkeit für eine eingefügte Nachricht sinkt damit von $1{,}5 \bullet 10^{-5}$ auf $3{,}5 \bullet 10^{-15}$ und wird so dem Einsatz im Rahmen eines Black-Channel Modells mit den Anforderungen der IEC 61508 für SIL3 gerecht.

Die Arbeit zeigt erstmals die zu betrachteten maximalen Reaktionszeiten und die Worst-Case Reaktionszeiten von FF-SIF für einfache 1-Link-Anwendungen, das heißt einer Sicherheitsfunktion, deren beteiligte Eingangskomponente, Logic-Solver und Ausgabekomponente sich ein einem H1-Link befinden.

Da für das beabsichtigte Einsatzgebiet von FF-SIF, die Prozessautomatisierung, der Aspekt Verfügbarkeit besondere Relevanz hat, definiert die Arbeit ein 2oo3 Eingangsmodell mit 3 Eingangskomponenten auf 3 unterschiedlichen H1-Links und 2 redundante, verfügbar kombinierte Ausgangskomponenten, wiederum auf 2 unterschiedlichen H1-Links. Dieses Modell zu Grunde legend wurden in der Arbeit sicherheitsgerichtete Anwendungsrichtlinien erstellt, ohne die eine solche Betrachtung der Reaktionszeiten nicht möglich ist.

Mit dem Modell und den Anwendungsrichtlinien definiert diese Arbeit erstmals die maximale und die Worst-Case Reaktionszeit einer FF-SIF basierten Sicherheitsfunktion mit redundanten FF-SIF Komponenten auf redundanten H1-Links.

Durch die Betrachtung erweiterter Gefahren von Kommunikationsprotokollen ergaben sich für FF-SIF zwei weitere Schwachstellen. Die C/S-Verbindungen von Engineering-Werkzeugen, z.B. über Linking-Devices, verwenden den gleichen Nachrichtenrahmen wie FF-SIF und daher können nicht sicherheitsgerichtete Komponenten (PCs), sichere Nachrichten erzeugen. Die daraus resultierende in dieser Arbeit ermittelte Restfehlerwahrscheinlichkeit genügt nicht dem Anspruch von 1% SIL3 und führt zum Ergebnis, dass C/S-Verbindungen, bzw. allgemein, nicht sicherheitsgerichtete Komponenten, keine sicherheitsgerichteten Protokolle verwenden dürfen.

Ebenfalls im Rahmen von C/S-Verbindungen wurde in dieser Arbeit aufgezeigt, dass der Schutz vor unberechtigter Manipulation, wie er durch das FF-SIF Konzept des Write-Locks erreicht werden sollte, ungeeignet ist. Erstens kann dieser Write-Lock mittels C/S-Verbindung ohne weitere Schutzmaßnahmen, wie z.B. Passwort oder Zustands-basierter Schutz, ausgeschaltet und dann sicherheitsgerichtete Parameter verändert werden. Andererseits gibt es im Rahmen von FF-SIF kein Konzept, dass den Betrieb von Geräten mit „nicht passender“ Parametrierung erkennt.

## 7.5 TSP

Einen großen Teil dieser Arbeit nimmt die Strukturierung, Definition und die sicherheitstechnische Analyse des Transport Safety Protokolls ein. Das Transport Safety Protokoll ist für den Einsatz der sicherheitsgerichteten Datenübertragung innerhalb von Automatisierungssystemen vorgesehenen. Das Transport Safety Protokoll kann bis zum Level SIL3 nach IEC 61508 und bis zur Kategorie CAT4/PL-e nach IEC 13849 eingesetzt werden.

Zur Bestimmung der Qualität zur Aufdeckung von Verfälschungen von TSP1-Nachrichten durch den CRC1 wurde im Rahmen der Arbeit nicht nur die bekannte Gleichung (2.6), sondern es wurden die konkreten Gewichte mittels Simulation ermittelt und berechnet. So konnte für die Nachrichtenlängen 2, 4 und 6 Nutzdaten-Bytes gezeigt werden, dass die Ergebnisse der Approximation nach Gleichung (2.6) sich in der gleichen Größenordnung befinden und für die zum Einsatz kommenden Nachrichten-

längen als gesichert angesehen werden können. Da der gewählte CRC1 für die Nachrichtenlängen bis 8 Nutzdaten-Bytes eine gute Qualität aufweist, kann mit TSP1 und einer Bitfehlerrate von $10^{-2}$ in realen Anwendungen der Sicherheitslevel SIL3 ohne Probleme erreicht werden. Dies ist um 5 Größenordnungen besser als das im Anwendungsbereich konkurrierende PROFIsafe V2 und um 3 Größenordnung besser als CIP-Safety Edition 2.2, jeweils mit ihren kurzen Nachrichtenformaten und im Gegensatz zu diesen insbesondere ausreichend um eine Bitfehlerrate von $10^{-2}$ zu beherrschen.

Für TSP2 Nachrichten legt diese Arbeit eine Analyse zur kombinierten Anwendung von CRC2 und CRC3 vor. Es wird gezeigt, dass für die Nachrichten mit 2, 4 und 6 Nutzdaten-Bytes die Restfehlerwahrscheinlichkeit einer durch CRC2 und CRC3 gesicherter TSP2 Nachricht kleiner als das Produkt der Einzelwahrscheinlichkeiten gemäß Gleichung (2.9) und kleiner als das Produkt der Wahrscheinlichkeiten der aus den einzelnen Gewichten berechneten Restfehlerwahrscheinlichkeiten für CRC2 und CRC3 ist.

Die Wirksamkeit der drei verschiedenen Transport Safety Protokoll CRCs ist so groß, dass TSP, im Gegensatz zu anderen Sicherheitsprotokollen, wie z.B. PROFIsafe und CIP-Safety, bei einer als verfälscht erkannten Nachricht keine Sicherheitsreaktion einleiten muss. Dies erhöht die Verfügbarkeit einer Anwendung maßgeblich, da sich bei den anderen Protokollen auch eine eingefügte Nachricht als „verfälschte“ Nachricht darstellt. TSP2 ist im Vergleich zu PROFIsafe und CIP-Safety als einziges Protokoll in der Lage 238 Bytes Nutzdaten, auch bei einer Bitfehlerrate von $10^{-2}$, sicher zu transportieren.

Als Maßnahme zum Schutz vor Wiederholungen und Einfügungen von Nachrichten, insbesondere auch von Nachrichtensequenzen, verfügt das Transport Safety Protokoll über einen einzigartigen Mechanismus. Dieser Mechanismus, einen asymmetrischen Schlüssel[256] zu verwenden, mit dem sowohl der Master, wie auch der Slave erkennen können, dass es sich bei empfangenen Nachrichten der Gegenseite um von ihm authentifizierte Nachrichten handelt. Diese Maßnahme ist so wirksam, dass das Transport Safety Protokoll im Gegensatz zu herkömmlichen Sicherheitsprotokollen, wie z.B. PROFIsafe, FF-SIF oder CIP-Safety, keine Anforderungen für den Anlauf des Protokolls an den Anwender stellen muss und einen automatischen Wiederanlauf unterstützt.

Der Vorteil des zulässigen automatischen Wiederanlaufs ist, dass der Anwender der Verantwortung entbunden wird, beurteilen zu müssen, ob sich das Kommunikationssystem in einem Gefahr-bringenden Zustand befindet. Dies ist insbesondere deshalb bedeutsam, weil der Anwender dies in der Praxis nicht entscheiden oder bestimmen kann.

Durch die Einbettung der Konfiguration der Slaves in das Transport Safety Protokoll wird zudem die oben bei FF-SIF genannte Schwäche vermieden, dass mit Geräten sicherheitsgerichtet kommuniziert wird, die eine von der Master-Erwartungshaltung abweichende Parametrierung aufweisen. Dies ist insbesondere auch für den Bereich der Wartungsarbeiten relevant, da somit zusätzliche Mechanismen zur Verfügung stehen, die ein getauschtes Gerät mit falschen Parametern erkennen können.

Schlussendlich bleibt für das Transport Safety Protokoll noch herauszustellen, dass für in der Praxis relevante Szenerien die Anwendungsgrenzen um eine Größenordnung unterschritten werden. Zusammen mit den anderen hoch wirksamen Maßnahmen ist das Transport Safety Protokoll 2 daher für Anwendungen mit einem Safety Integrity Level 4 gemäß IEC 61508 geeignet. Dies wird von den Sicherheitsprotokollen, wie z.B. PROFIsafe, FF-SIF oder CIP-Safety nicht erreicht.

Dies qualifiziert das Transport Safety Protokoll und insbesondere die TSP2-Variante zum Beispiel für den Einsatz in Bahnanwendungen gemäß IEC 50159 und CENELEC SIL4.

[256] den jeweiligen CRC-Preset

# 8 Ausblick

Die Ansprüche, die bei der Zertifizierung der sicherheitsgerichteten Protokolle von den Auditierungsstellen gestellt wurden, sind offensichtlich sehr unterschiedlich. Vielfach sind sie mit den aktuellen Ansprüchen gemäß IEC 61508-2 und IEC 61784-3 nicht vereinbar[257].

Aus diesem Grund ist es wünschenswert, wenn zukünftig die quantitativen Bewertungen der Sicherheitsmaßnahmen der Protokolle öffentlich gemacht und in die jeweiligen Spezifikationen aufgenommen werden. Dies stellt dann auch eine geeignete Maßnahme dar, dass Organisationen, die erstmals ein sicherheitsgerichtetes Protokoll spezifizieren, von den Erfahrungen der bestehenden Protokolle profitieren können. Viele Organisationen sind sich nicht bewusst, dass sie für alle Maßnahmen einen quantitativen Nachweis der Wirksamkeit erbringen müssen. So erreichte kaum ein Protokoll bei seiner ersten veröffentlichten Version alle Ansprüche für sicherheitsgerichtete Protokolle[258].

Mit der Offenlegung der quantitativen Bewertungen hätte dann auch der Anwender die Möglichkeit die sicherheitstechnische Leistungsfähigkeit der Protokolle zu vergleichen und die Auflagen und Annahmen für den praktischen Betrieb zu bewerten.

Hier hat eine Veröffentlichung[259] im Rahmen dieser Arbeit dazu geführt, dass der Arbeitskreis DKE 914.1[260], der für die Pflege der IEC 61784-3 zuständig ist, den Autor zur Zusammenarbeit eingeladen hat, um die quantitative Analyse und die dazu gehörenden Anforderungen für die IEC 61784-3 zur erarbeiten.

## 8.1 Security

Die hier betrachteten Protokolle sind durchweg nur für geschlossene Übertragungseinrichtungen ausgelegt. An dieser Stelle ist der Anwender gefordert, diese Voraussetzung umzusetzen und insbesondere während der gesamten Lebensdauer einzuhalten. Dies ist bei Ethernet basierten Netzen, auf Grund der einzurechnenden Fehlbedienung, kein leichtes Unterfangen.

Security relevante Aspekte haben bislang kaum Einzug in das Thema sicherheitsgerichtete Kommunikation gefunden. Einige Anwender haben diese Problematik bereits erkannt und definieren selbst Kriterien und Maßnahmen, die von den Kommunikationssystemen zu erfüllen sind[261].

Dabei werden nicht sicherheitsgerichtete Komponenten ebenso wie sicherheitsgerichtete betrachtet. Für die meisten Anwendungen sicherheitsgerichteter Kommunikation sind diese Maßnahmen nicht innerhalb der sicherheitsgerichteten Protokolle erforderlich, sondern können durch externe Komponenten umgesetzt werden[262].

In diesem Bereich sind die qualitativen Ansprüche, die eine Security Implementierung aufweisen muss, bislang nicht festgelegt. Sind hier für die Software Entwicklung die Maßstäbe der IEC 61508-3 anzulegen? Wie prüft eine Auditierungsstelle die Security Funktion?

## 8.2 TSP Erweiterungen

Die Weiterentwicklung von TSP für den Einsatz in offenen Kommunikationsnetzen stellt einen Bereich dar, der die Anwendung von TSP, z.B. im Bahnbereich, noch universeller machen würde. Hierzu können kryptographische Techniken verwendet werden. Mit diesen ließe sich dann der Anspruch an die nicht sicherheitsgerichteten Security Komponenten reduzieren und eine Qualifizierung der Security Maßnahme durchführen. Da TSP keine Anforderungen kennt, eine Sicherheitsreaktion einzulei-

[257] [CIP5],[FSoE]

[258] [PROFIsafeV1], [CIP5], [FF-SIF]

[259] [Han12]

[260] Deutsche Kommission Elektrotechnik Elektronik Informationstechnik, Profile für sichere Kommunikationssysteme

[261] private Diskussion, Shell, Total u.a.

[262] [PROFIsafeV2]

ten, wenn nicht korrekte Nachrichten empfangen werden, ist es auch gegenüber diversen Denial-of-Service Attacken mit einer Grundsicherheit ausgerüstet.

Soll bei hohen Anforderungen an die Sicherheit nicht immer gleich TSP2 zum Einsatz kommen, so liegt in der Qualifizierung des nicht sicheren Übertragungssystems eine gewinnbringende Möglichkeit. Mit der dann möglichen Verwendung von TSP1 werden HW-Ressourcen eingespart, beziehungsweise die Performance verbessert und ein wirtschaftlicherer Einsatz erreicht. Dazu ist es notwendig die verwendeten Komponenten zu definieren und bezüglich ihrer Fehlermodelle zu analysieren. Hier sind die Hersteller der integrierten Schaltkreise besonders gefordert, da nur sie die Möglichkeit besitzen entsprechende Qualitätszahlen und Fehlermodelle zu ermitteln.

## 8.3 Das Unified Safety Protocol

Was bei der Übertragungstechnik mit dem TCP/IP-Protokoll gelungen ist, nämlich dass sich ein Protokoll durchsetzt, wäre auch für sicherheitsgerichtete Protokolle möglich. Betrachtet man die Leistungsfähigkeit und die Restriktionen der unterlagerten Transportprotokolle, so könnten die Protokolle PROFIsafe, Fail-Safe-over-EtherCAT, Foundation-Fieldbus FF-SIF und TSP durch ein einheitliches Protokoll realisiert werden. Verhindert wird dies zum Teil durch Patente[263], aber auch auf Grund der Tatsache, dass sich die Hersteller mit individuellen Protokollen voneinander abheben möchten.

## 8.4 Wirksamkeit von CRC Sicherungsmechanismen

Für die Berechnung der Restfehlerwahrscheinlichkeit für CRC Blocksicherungen fehlen mathematische Modelle und effiziente Simulationstechniken, die eine praxistaugliche Restfehlerberechnung für lange Nachrichten ermöglichen. Hierbei stellen Arbeiten von Mattes[264] erste Ansätze dar, die den Simulationsaufwand reduzieren.

So ist die Approximation mit dem Faktor $1/2^k$ nicht allgemeingültig, sowie das Produkt der Einzelwahrscheinlichkeiten für kombinierte CRCs nicht exakt mit den simulierten Ergebnissen vergleichbar. Da die sicherheitstechnisch zu betrachtenden Datenmengen immer größer werden, besteht auf diesem Gebiet ein vermehrter Bedarf an praktikablen Lösungen.

Werden keine seriellen Übertragungstechniken genutzt, so ist zu prüfen, ob das Modell vom binär symmetrischen Kanal zur Anwendung kommen kann oder ob andere Modelle erstellt werden müssen. So ist z.B. zu untersuchen, wie sich ein Übersprechen paralleler Datenleitungen auf das Fehlermodell auswirkt, da hier nicht mehr von einer Unabhängigkeit der Verfälschung einzelner Bits ausgegangen werden kann.

[263] [Bart99], [Bart99b]

[264] [Matt08]

# Anhang

## Begriffe

### Black-Channel

Black-Channel beschreibt einen Kommunikationskanal, dessen Aufbau und Design keinen Bezug zur sicherheitstechnischen Bewertung des Protokolls aufweist[265].

### CRC

CRC steht für *Cyclic Redundancy Code* und die Zahl hinter dem CRC, z.B. CRC32, gibt den Grad des Polynoms – 1 an und steht für die Größe des Rests der Polynomdivision in Bits. Die CRC Berechnung bezeichnet eine Technik zur Erkennung von Nachrichtenverfälschungen, bei der die Reste der Polynomdivision einer Nachricht in der Nachricht mit übertragen und beim Empfänger geprüft werden.

### Geschlossenes Übertragungssystem

Ein geschlossenes Übertragungssystem ist ein Netzwerk, dass durch technische und/oder organisatorische Maßnahmen davor geschützt wird, dass eine unzulässige Beeinflussung des Netzwerks erfolgt. Im Gegensatz dazu ist ein offenes Übertragungssystem vor solchen Einflüssen nicht oder nur unzureichend geschützt.

### Sicherheit

Der Begriff „Sicherheit“ wird in dieser Arbeit häufig verwendet werden. Mit Sicherheit, im Englischen eindeutiger mit dem Wort „safety” beschrieben, ist hier die „funktionale Sicherheit“ gemeint, wie sie in IEC 61508-4[266] definiert ist. Der Begriff Sicherheit im Sinne von Zugriffsschutz oder Englisch „security”, wird hier nicht verwendet.

### Sicherheitsfunktion

In der Sicherheitstechnik bezeichnet man mit dem Begriff Sicherheitsfunktion (engl. Safety Loop oder Safety Function) alle dazu erforderlichen Komponenten, bestehend aus Eingangs-, Logikverarbeitungs-, Ausgangskomponenten und den Verbindungen zwischen diesen, die zusammen dazu dienen eine Funktion zu realisieren, die einen sicheren Zustand beibehält oder herstellt[267]. Beim Einsatz von sicherheitsgerichteten Kommunikationstechniken werden für die Verbindungen Kommunikationsverbindungen eingesetzt.

### Watch-Dog-Timer

Watch-Dog-Timer ist eine Technik zur zeitlichen Überwachung des Abstands von Ereignissen und soll insbesondere das Ausbleiben eines erwarteten Ereignisses aufdecken. Im Rahmen dieser Arbeit ist mit dem Begriff Watch-Dog-Timer in der Regel der Überwachungs-Timer für den Abstand empfangener Nachrichten oder die Antwortzeit auf Nachrichten gemeint.

## CRC1 Berechnung für TSP1

Die nachfolgende Funktion berechnet den CRC1 von TSP1 auf einer Big-Endian[268] Maschine. Dabei wird auf das Tabellenverfahren zurückgegriffen, um eine möglichst optimale Performance zu erzielen.

Als Repräsentation des Polynoms CRC1 kommt die gespiegelte Form, zusammen mit der gespiegelten Berechnungsmethode zum Einsatz. Damit wird der CRC normal berechnet, bis auf den Umstand, dass die für serielle Übertragungen übliche Technik, zuerst das least significant Bit, das heißt die Nutzdaten-Bytes in gespiegelter Form, in die Berechnung eingesetzt werden.

---

[265] [IEC 61508-2]

[266] [IEC 61508-4] Kapitel 3.1.11 und 3.1.12

[267] [IEC 61508-4]

[268] Bei einer Little-Endian Maschine müssen die Bytes von `val` in umgekehrter Reihenfolge bearbeitet werden.

*Anmerkung: Bei der Anwendung der CRC Routine GetCRC_TSP1() müssen die Nutzdaten-Bytes nicht explizit gespiegelt werden, dies geschieht implizit durch den Algorithmus der Routine.*

```
typedef unsigned long udword;
typedef unsigned char ubyte;

udword
GetCRC_TSP1(const void* pStart, size_t len, udword preset)
{
  udword crcTSP1 = preset;
  const ubyte *buf = (const ubyte *) pStart;
  if (len >= 4) {
    size_t count= len >> 2; // Anzahl 4 Byte Blöcke
    do {
      udword val= ((const udword*)buf)[0];
      crcTSP1 = maCRC_TSP1[(crcTSP1^(val>>24)) & 0x0ff]^( crcTSP1 >> 8L );
      crcTSP1 = maCRC_TSP1[(crcTSP1^(val>>16)) & 0x0ff]^( crcTSP1 >> 8L );
      crcTSP1 = maCRC_TSP1[(crcTSP1^(val>>8))  & 0x0ff]^( crcTSP1 >> 8L );
      crcTSP1 = maCRC_TSP1[(crcTSP1^(val))     & 0x0ff]^( crcTSP1 >> 8L );
      count--;
      buf+=4;
    } while (count > 0);
    len= len & 0x03; // der Rest
  }
  // Rechnen des restlichen CRC
  for (size_t count=len; count > 0; count--,buf++) {
    crcTSP1 = maCRC_TSP1[(crcTSP1^*buf) & 0x0ff]^( crcTSP1 >> 8L );
  }
  return crcTSP1;
}
```

Tabelle zur Berechnung von CRC1 des TSP1.

```
udword maCRC_TSP1[256] = { // gespiegelter CRC1 = 0xA814498F
0x00000000,0x7dff4f11,0xfbfe9e22,0x8601d133,0xa7d5af5b,0xda2ae04a,0x5c2b3179,0x21d47e68,
0x1f83cda9,0x627c82b8,0xe47d538b,0x99821c9a,0xb85662f2,0xc5a92de3,0x43a8fcd0,0x3e57b3c1,
0x3f079b52,0x42f8d443,0xc4f90570,0xb9064a61,0x98d23409,0xe52d7b18,0x632caa2b,0x1ed3e53a,
0x208456fb,0x5d7b19ea,0xdb7ac8d9,0xa68587c8,0x8751f9a0,0xfaaeb6b1,0x7caf6782,0x01502893,
0x7e0f36a4,0x03f079b5,0x85f1a886,0xf80ee797,0xd9da99ff,0xa425d6ee,0x222407dd,0x5fdb48cc,
0x618cfb0d,0x1c73b41c,0x9a72652f,0xe78d2a3e,0xc6595456,0xbba61b47,0x3da7ca74,0x40588565,
0x4108adf6,0x3cf7e2e7,0xbaf633d4,0xc7097cc5,0xe6dd02ad,0x9b224dbc,0x1d239c8f,0x60dcd39e,

0x5e8b605f,0x23742f4e,0xa575fe7d,0xd88ab16c,0xf95ecf04,0x84a18015,0x02a05126,0x7f5f1e37,
0xfc1e6d48,0x81e12259,0x07e0f36a,0x7a1fbc7b,0x5bcbc213,0x26348d02,0xa0355c31,0xddca1320,
0xe39da0e1,0x9e62eff0,0x18633ec3,0x659c71d2,0x44480fba,0x39b740ab,0xbfb69198,0xc249de89,
0xc319f61a,0xbee6b90b,0x38e76838,0x45182729,0x64cc5941,0x19331650,0x9f32c763,0xe2cd8872,
0xdc9a3bb3,0xa16574a2,0x2764a591,0x5a9bea80,0x7b4f94e8,0x06b0dbf9,0x80b10aca,0xfd4e45db,
0x82115bec,0xffee14fd,0x79efc5ce,0x04108adf,0x25c4f4b7,0x583bbba6,0xde3a6a95,0xa3c52584,
0x9d929645,0xe06dd954,0x666c0867,0x1b934776,0x3a47391e,0x47b8760f,0xc1b9a73c,0xbc46e82d,
0xbd16c0be,0xc0e98faf,0x46e85e9c,0x3b17118d,0x1ac36fe5,0x673c20f4,0xe13df1c7,0x9cc2bed6,

0xa2950d17,0xdf6a4206,0x596b9335,0x2494dc24,0x0540a24c,0x78bfed5d,0xfebe3c6e,0x8341737f,
0xa814498f,0xd5eb069e,0x53ead7ad,0x2e1598bc,0x0fc1e6d4,0x723ea9c5,0xf43f78f6,0x89c037e7,
0xb7978426,0xca68cb37,0x4c691a04,0x31965515,0x10422b7d,0x6dbd646c,0xebbcb55f,0x9643fa4e,
0x9713d2dd,0xeaec9dcc,0x6ced4cff,0x111203ee,0x30c67d86,0x4d393297,0xcb38e3a4,0xb6c7acb5,
0x88901f74,0xf56f5065,0x736e8156,0x0e91ce47,0x2f45b02f,0x52baff3e,0xd4bb2e0d,0xa944611c,
0xd61b7f2b,0xabe4303a,0x2de5e109,0x501aae18,0x71ced070,0x0c319f61,0x8a304e52,0xf7cf0143,
0xc998b282,0xb467fd93,0x32662ca0,0x4f9963b1,0x6e4d1dd9,0x13b252c8,0x95b383fb,0xe84cccea,
0xe91ce479,0x94e3ab68,0x12e27a5b,0x6f1d354a,0x4ec94b22,0x33360433,0xb537d500,0xc8c89a11,

0xf69f29d0,0x8b6066c1,0x0d61b7f2,0x709ef8e3,0x514a868b,0x2cb5c99a,0xaab418a9,0xd74b57b8,
0x540a24c7,0x29f56bd6,0xaff4bae5,0xd20bf5f4,0xf3df8b9c,0x8e20c48d,0x082115be,0x75de5aaf,
0x4b89e96e,0x3676a67f,0xb077774c,0xcd88385d,0xec5c4635,0x91a30924,0x17a2d817,0x6a5d9706,
0x6b0dbf95,0x16f2f084,0x90f321b7,0xed0c6ea6,0xccd810ce,0xb1275fdf,0x37268eec,0x4ad9c1fd,
```

```
0x748e723c,0x09713d2d,0x8f70ec1e,0xf28fa30f,0xd35bdd67,0xaea49276,0x28a54345,0x555a0c54,
0x2a051263,0x57fa5d72,0xd1fb8c41,0xac04c350,0x8dd0bd38,0xf02ff229,0x762e231a,0x0bd16c0b,
0x3586dfca,0x487990db,0xce7841e8,0xb3870ef9,0x92537091,0xefac3f80,0x69adeeb3,0x1452a1a2,
0x15028931,0x68fdc620,0xeefc1713,0x93035802,0xb2d7266a,0xcf28697b,0x4929b848,0x34d6f759,
0x0a814498,0x777e0b89,0xf17fdaba,0x8c8095ab,0xad54ebc3,0xd0aba4d2,0x56aa75e1,0x2b553af0
};
```

## CRC2 + CRC3 Berechnung für TSP2

Die nachfolgende Funktion berechnet den CRC2 + CRC3 von TSP2 auf einer Big-Endian Maschine. Dabei wird auf das Tabellenverfahren zurückgegriffen, um eine möglichst optimale Performance zu erzielen.

Als Repräsentation der Polynome CRC2/3 kommt die jeweils gespiegelte Form, zusammen mit der gespiegelten Berechnungsmethode zum Einsatz. Damit werden die CRCs normal berechnet, bis auf den Umstand, dass die für serielle Übertragungen übliche Technik, zuerst das least significant Bit, das heißt die Nutzdaten-Bytes in gespiegelter Form, in die Berechnung eingesetzt werden.

*Anmerkung: Bei der Anwendung der CRC Routine GetCRC_TSP2() müssen die Nutzdaten-Bytes nicht explizit gespiegelt werden, dies geschieht implizit durch den Algorithmus der Routine.*

```
typedef unsigned long udword;
typedef unsigned char ubyte;

void
GetCRC_TSP2( const void* pStart, size_t len,
             udword &preset2a, // Preset und Ergebnis für CRC2
             udword &preset2b) // Preset und Ergebnis für CRC3
{
  const ubyte *buf = (const ubyte *) pStart;
  if (len >= 4) {
    size_t count= len >> 2; // Anzahl 4 Byte Blöcke
    do {
      udword val= ((const udword*)buf)[0];
      preset2a = maCRC_TSP2a[(preset2a^(val>>24)) & 0x0ff]^( preset2a >> 8L );
      preset2b = maCRC_TSP2b[(preset2b^(val>>24)) & 0x0ff]^( preset2b >> 8L );
      preset2a = maCRC_TSP2a[(preset2a^(val>>16)) & 0x0ff]^( preset2a >> 8L );
      preset2b = maCRC_TSP2b[(preset2b^(val>>16)) & 0x0ff]^( preset2b >> 8L );
      preset2a = maCRC_TSP2a[(preset2a^(val>> 8)) & 0x0ff]^( preset2a >> 8L );
      preset2b = maCRC_TSP2b[(preset2b^(val>> 8)) & 0x0ff]^( preset2b >> 8L );
      preset2a = maCRC_TSP2a[(preset2a^(val))     & 0x0ff]^( preset2a >> 8L );
      preset2b = maCRC_TSP2b[(preset2b^(val))     & 0x0ff]^( preset2b >> 8L );
      count--;
      buf+=4;
    } while (count > 0);
    len= len & 0x03; // der Rest
  }
  // Rechnen des restlichen CRC
  for (size_t count=len; count > 0; count--,buf++) {
    preset2a = maCRC_TSP2a[(preset2a^*buf) & 0x0ff]^( preset2a >> 8L );
    preset2b = maCRC_TSP2b[(preset2b^*buf) & 0x0ff]^( preset2b >> 8L );
  }
}
```

Tabelle zur Berechnung von CRC2 des TSP2

```
udword maCRC_TSP2a[256] = { // gespiegelter CRC2 = 0x992c_1a4c
0x00000000,0xce3f0db3,0xae262fff,0x6019224c,0x6e146b67,0xa02b66d4,0xc0324498,0x0e0d492b,
0xdc28d6ce,0x1217db7d,0x720ef931,0xbc31f482,0xb23cbda9,0x7c03b01a,0x1c1a9256,0xd2259fe5,
0x8a099905,0x443694b6,0x242fb6fa,0xea10bb49,0xe41df262,0x2a22ffd1,0x4a3bdd9d,0x8404d02e,
0x56214fcb,0x981e4278,0xf8076034,0x36386d87,0x383524ac,0xf60a291f,0x96130b53,0x582c06e0,
```

```
0x264b0693,0xe8740b20,0x886d296c,0x465224df,0x485f6df4,0x86606047,0xe679420b,0x28464fb8,
0xfa63d05d,0x345cddee,0x5445ffa2,0x9a7af211,0x9477bb3a,0x5a48b689,0x3a5194c5,0xf46e9976,
0xac429f96,0x627d9225,0x0264b069,0xcc5bbdda,0xc256f4f1,0x0c69f942,0x6c70db0e,0xa24fd6bd,
0x706a4958,0xbe5544eb,0xde4c66a7,0x10736b14,0x1e7e223f,0xd0412f8c,0xb0580dc0,0x7e670073,

0x4c960d26,0x82a90095,0xe2b022d9,0x2c8f2f6a,0x22826641,0xecbd6bf2,0x8ca449be,0x429b440d,
0x90bedbe8,0x5e81d65b,0x3e98f417,0xf0a7f9a4,0xfeaab08f,0x3095bd3c,0x508c9f70,0x9eb392c3,
0xc69f9423,0x08a09990,0x68b9bbdc,0xa686b66f,0xa88bff44,0x66b4f2f7,0x06add0bb,0xc892dd08,
0x1ab742ed,0xd4884f5e,0xb4916d12,0x7aae60a1,0x74a3298a,0xba9c2439,0xda850675,0x14ba0bc6,
0x6add0bb5,0xa4e20606,0xc4fb244a,0x0ac429f9,0x04c960d2,0xcaf66d61,0xaaef4f2d,0x64d0429e,
0xb6f5dd7b,0x78cad0c8,0x18d3f284,0xd6ecff37,0xd8e1b61c,0x16debbaf,0x76c799e3,0xb8f89450,
0xe0d492b0,0x2eeb9f03,0x4ef2bd4f,0x80cdb0fc,0x8ec0f9d7,0x40fff464,0x20e6d628,0xeed9db9b,
0x3cfc447e,0xf2c349cd,0x92da6b81,0x5ce56632,0x52e82f19,0x9cd722aa,0xfcce00e6,0x32f10d55,

0x992c1a4c,0x571317ff,0x370a35b3,0xf9353800,0xf738712b,0x39077c98,0x591e5ed4,0x97215367,
0x4504cc82,0x8b3bc131,0xeb22e37d,0x251deece,0x2b10a7e5,0xe52faa56,0x8536881a,0x4b0985a9,
0x13258349,0xdd1a8efa,0xbd03acb6,0x733ca105,0x7d31e82e,0xb30ee59d,0xd317c7d1,0x1d28ca62,
0xcf0d5587,0x01325834,0x612b7a78,0xaf1477cb,0xa1193ee0,0x6f263353,0x0f3f111f,0xc1001cac,
0xbf671cdf,0x7158116c,0x11413320,0xdf7e3e93,0xd17377b8,0x1f4c7a0b,0x7f555847,0xb16a55f4,
0x634fca11,0xad70c7a2,0xcd69e5ee,0x0356e85d,0x0d5ba176,0xc364acc5,0xa37d8e89,0x6d42833a,
0x356e85da,0xfb518869,0x9b48aa25,0x5577a796,0x5b7aeebd,0x9545e30e,0xf55cc142,0x3b63ccf1,
0xe9465314,0x27795ea7,0x47607ceb,0x895f7158,0x87523873,0x496d35c0,0x2974178c,0xe74b1a3f,

0xd5ba176a,0x1b851ad9,0x7b9c3895,0xb5a33526,0xbbae7c0d,0x759171be,0x158853f2,0xdbb75e41,
0x0992c1a4,0xc7adcc17,0xa7b4ee5b,0x698be3e8,0x6786aac3,0xa9b9a770,0xc9a0853c,0x079f888f,
0x5fb38e6f,0x918c83dc,0xf195a190,0x3faaac23,0x31a7e508,0xff98e8bb,0x9f81caf7,0x51bec744,
0x839b58a1,0x4da45512,0x2dbd775e,0xe3827aed,0xed8f33c6,0x23b03e75,0x43a91c39,0x8d96118a,
0xf3f111f9,0x3dce1c4a,0x5dd73e06,0x93e833b5,0x9de57a9e,0x53da772d,0x33c35561,0xfdfc58d2,
0x2fd9c737,0xe1e6ca84,0x81ffe8c8,0x4fc0e57b,0x41cdac50,0x8ff2a1e3,0xefeb83af,0x21d48e1c,
0x79f888fc,0xb7c7854f,0xd7dea703,0x19e1aab0,0x17ece39b,0xd9d3ee28,0xb9cacc64,0x77f5c1d7,
0xa5d05e32,0x6bef5381,0x0bf671cd,0xc5c97c7e,0xcbc43555,0x05fb38e6,0x65e21aaa,0xabdd1719
};
```

## Tabelle zur Berechnung von CRC3 des TSP2

```
udword maCRC_TSP2b[256] = { // gespiegelter CRC3 = 0xC8DF356F
0x00000000,0xf85a3a8b,0x610a1fc9,0x99502542,0xc2143f92,0x3a4e0519,0xa31e205b,0x5b441ad0,
0x159615fb,0xedcc2f70,0x749c0a32,0x8cc630b9,0xd7822a69,0x2fd810e2,0xb68835a0,0x4ed20f2b,
0x2b2c2bf6,0xd376117d,0x4a26343f,0xb27c0eb4,0xe9381464,0x11622eef,0x88320bad,0x70683126,
0x3eba3e0d,0xc6e00486,0x5fb021c4,0xa7ea1b4f,0xfcae019f,0x04f43b14,0x9da41e56,0x65fe24dd,
0x565857ec,0xae026d67,0x37524825,0xcf0872ae,0x944c687e,0x6c1652f5,0xf54677b7,0x0d1c4d3c,
0x43ce4217,0xbb94789c,0x22c45dde,0xda9e6755,0x81da7d85,0x7980470e,0xe0d0624c,0x188a58c7,
0x7d747c1a,0x852e4691,0x1c7e63d3,0xe4245958,0xbf604388,0x473a7903,0xde6a5c41,0x263066ca,
0x68e269e1,0x90b8536a,0x09e87628,0xf1b24ca3,0xaaf65673,0x52ac6cf8,0xcbfc49ba,0x33a67331,

0xacb0afd8,0x54ea9553,0xcdbab011,0x35e08a9a,0x6ea4904a,0x96feaac1,0x0fae8f83,0xf7f4b508,
0xb926ba23,0x417c80a8,0xd82ca5ea,0x20769f61,0x7b3285b1,0x8368bf3a,0x1a389a78,0xe262a0f3,
0x879c842e,0x7fc6bea5,0xe6969be7,0x1ecca16c,0x4588bbbc,0xbdd28137,0x2482a475,0xdcd89efe,
0x920a91d5,0x6a50ab5e,0xf3008e1c,0x0b5ab497,0x501eae47,0xa84494cc,0x3114b18e,0xc94e8b05,
0xfae8f834,0x02b2c2bf,0x9be2e7fd,0x63b8dd76,0x38fcc7a6,0xc0a6fd2d,0x59f6d86f,0xa1ace2e4,
0xef7eedcf,0x1724d744,0x8e74f206,0x762ec88d,0x2d6ad25d,0xd530e8d6,0x4c60cd94,0xb43af71f,
0xd1c4d3c2,0x299ee949,0xb0cecc0b,0x4894f680,0x13d0ec50,0xeb8ad6db,0x72daf399,0x8a80c912,
0xc452c639,0x3c08fcb2,0xa558d9f0,0x5d02e37b,0x0646f9ab,0xfe1cc320,0x674ce662,0x9f16dce9,

0xc8df356f,0x30850fe4,0xa9d52aa6,0x518f102d,0x0acb0afd,0xf2913076,0x6bc11534,0x939b2fbf,
0xdd492094,0x25131a1f,0xbc433f5d,0x441905d6,0x1f5d1f06,0xe707258d,0x7e5700cf,0x860d3a44,
0xe3f31e99,0x1ba92412,0x82f90150,0x7aa33bdb,0x21e7210b,0xd9bd1b80,0x40ed3ec2,0xb8b70449,
0xf6650b62,0x0e3f31e9,0x976f14ab,0x6f352e20,0x347134f0,0xcc2b0e7b,0x557b2b39,0xad2111b2,
```

```
0x9e876283,0x66dd5808,0xff8d7d4a,0x07d747c1,0x5c935d11,0xa4c9679a,0x3d9942d8,0xc5c37853,
0x8b117778,0x734b4df3,0xea1b68b1,0x1241523a,0x490548ea,0xb15f7261,0x280f5723,0xd0556da8,
0xb5ab4975,0x4df173fe,0xd4a156bc,0x2cfb6c37,0x77bf76e7,0x8fe54c6c,0x16b5692e,0xeeef53a5,
0xa03d5c8e,0x58676605,0xc1374347,0x396d79cc,0x6229631c,0x9a735997,0x03237cd5,0xfb79465e,

0x646f9ab7,0x9c35a03c,0x0565857e,0xfd3fbff5,0xa67ba525,0x5e219fae,0xc771baec,0x3f2b8067,
0x71f98f4c,0x89a3b5c7,0x10f39085,0xe8a9aa0e,0xb3edb0de,0x4bb78a55,0xd2e7af17,0x2abd959c,
0x4f43b141,0xb7198bca,0x2e49ae88,0xd6139403,0x8d578ed3,0x750db458,0xec5d911a,0x1407ab91,
0x5ad5a4ba,0xa28f9e31,0x3bdfbb73,0xc38581f8,0x98c19b28,0x609ba1a3,0xf9cb84e1,0x0191be6a,
0x3237cd5b,0xca6df7d0,0x533dd292,0xab67e819,0xf023f2c9,0x0879c842,0x9129ed00,0x6973d78b,
0x27a1d8a0,0xdffbe22b,0x46abc769,0xbef1fde2,0xe5b5e732,0x1defddb9,0x84bff8fb,0x7ce5c270,
0x191be6ad,0xe141dc26,0x7811f964,0x804bc3ef,0xdb0fd93f,0x2355e3b4,0xba05c6f6,0x425ffc7d,
0x0c8df356,0xf4d7c9dd,0x6d87ec9f,0x95ddd614,0xce99ccc4,0x36c3f64f,0xaf93d30d,0x57c9e986
};
```

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Abkürzungsverzeichnis

| Kürzel | Bedeutung |
|---|---|
| Ack | Acknowledge |
| B.K. | Binominialkoeffizient |
| BGIA | Berufs-genossenschaftliches Institut für Arbeitssicherheit, heute IFA |
| *c* | Anzahlt der Kommunikationsverbindungen einer Sicherheitsfunktion |
| CAN | Controller Area Network |
| CAT4 | Gefährdungskategorie 4 gemäß EN 954-1 |
| CENELEC | Comité Européen de Normalisation Électrotechnique / European Committee for Electrotechnical Standardization |
| CID | Consumer Identifier[269] |
| CIP | Common Industrial Protocol der ODVA |
| CPF | Communication Profile Family[270] |
| CRC | Cyclic Redundancy Code |
| CRC16 | Cyclic Redundancy Code mit Polynom des Grads 16 |
| CRC32 | Cyclic Redundancy Code mit Polynom des Grads 32 |
| CSMA/CD | Carrier Sense Multiple Access / Collision Detection |
| *d* | Hamming-Distanz |
| DAT | Device Acknowledge Time[270] |
| DPRAM | Dual-Port-RAM |
| EA-Punkte | Eingangs-/Ausgangspunkte |
| EN | Europäische Norm |
| EPI | Expected Packet Interval[269] |
| FB | Function-Block, Verwendungen bei asynchronen Diensten von PROFIBUS, PROFINET und Bezeichnung von logischen Komponente im Rahmen der FF-SIF Spezifikation |
| F-Device | Begriff für die sicherheitsgerichtete Slave Komponente im Rahmen des PROFIsafe Protokolls |
| F-Host | Begriff für die Sicherheitssteuerung im Rahmen des PROFIsafe Protokolls |

[269] [CIP5]

[270] [IEC61784-3-3]

| **Kürzel** | **Bedeutung** |
|---|---|
| F-Modul | Fail-Safe Modul[270] |
| FF | Foundation-Fieldbus |
| FF-H1 | Foundation-Fieldbus Protocol H1 |
| FF-SIF | Foundation-Fieldbus Safety Integrated Function |
| FF-SIF-AI | FF-SIF Funktionsbaustein für einen analogen Eingang[271] |
| FF-SIF-C/S | FF-SIF Client/Server |
| FF-SIF-DI | FF-SIF Funktionsbaustein für einen digitalen Eingang[271] |
| FF-SIF-DO | FF-SIF Funktionsbaustein für einen digitalen Ausgang[271] |
| FIT | Failure in Time, 1 FIT ist 1 Ausfall pro $10^9$ Stunden[272] |
| Frag | Fragment einer Information, z.B. Nachricht oder Konfiguration |
| FSCP | Fail Safe Communication Profile[270] |
| FSoEC | Fail Safe over EtherCAT[273] |
| GBit/s | Giga-Bit pro Sekunde ($10^9$) |
| HAT | Host Acknowledge Time[270] |
| HD | Hamming-Distanz |
| HFB | FF-SIF Host-Function-Block[271] |
| HW | Hardware |
| ID | eineindeutige Nummer zur Identifikation eines Absenders, Empfängers oder einer Verbindung |
| IEC | International Electrotechnical Commission |
| IEEE | Institute of Electrical and Electronics Engineers |
| IFA | Institut für Arbeitsschutz der deutschen Gesetzlichen Unfallversicherung, früher BGIA. |
| Ind | Indication |
| $k$ | Grad eines CRC Polynoms |
| LO | Local Override[271] |

[271] [FF-SIF]

[272] [Boer06]

[273] [FSoEC]

| Kürzel | Bedeutung |
|---|---|
| Logic-Solver | Bezeichnung für eine Sicherheitssteuerung im Rahmen von FF-SIF |
| LSB | Least Significant Bit |
| MAC-Adresse | Media Access Control Adresse |
| MBit/s | Mega-Bit pro Sekunde ($10^6$) |
| *MCN* | Macro Cycle Number von FF-SIF[271] |
| MSB | Most Significant Bit |
| N | Nachrichtenlänge in Bit ohne CRC |
| No | Number |
| NTE | Network Time Expectation[269] |
| ODVA | Open DeviceNet Vendors Association |
| OI | Object-Index[271] |
| *p* | Bitfehlerrate einer Kommunikationsverbindung |
| PA | Prozessautomatisierung |
| PADT | Programming and Debugging Tool[274] |
| PFD | Probability of dangerous Failure on Demand[272] |
| PFH | Average frequency of dangerous failure [$h^{-1}$] per Hour[272] |
| PID | Producer Identifier[269] |
| PL-e | Performance Level e gemäß IEC 13849[275] |
| *R(p)* | Restfehlerwahrscheinlichkeit einer unerkannt verfälschten Nachricht |
| Req | Request |
| Resp | Response |
| SCL | Stale-Count-Limit[271] |
| SDT | Stale-Data-Time[271] |
| Seq-No | Sequence-Number |
| SFRT | Safety Function Response Time[276] |

[274] [IEC 61508-4]

[275] [IEC13849]

[276] [IEC61784-3]

| **Kürzel** | **Bedeutung** |
|---|---|
| SIL | Safety Integrity Level gemäß IEC 61508[272] |
| SPS | Speicher-Programmierbare Steuerung |
| SSPS | Sicherheitsgerichtete Speicher-Programmierbare-Steuerung (Sicherheitssteuerung) |
| SW | Software |
| TBit/s | Terra-Bit pro Sekunde ($10^{12}$) |
| $t_{block}$ | Maximale Zeit, die das Versenden einer Nachricht blockiert ist |
| $t_{com}$ | Der zeitliche Abstand, den eine korrekt empfangene Nachricht nicht gegenüber ihrem Referenzzeitpunkt, verspätet sein darf. |
| $t_{com\text{-}in}$ | Die Zeit, die die Übertragung eines neuen von der Eingangskomponente erkannten Signalwerts in einer Nachricht zur Sicherheitssteuerung dauert. |
| $t_{com\text{-}out}$ | Die Zeit, die die Übertragung eines neuen berechneten Ergebnisses der Sicherheitssteuerung in einer Nachricht zur Ausgangskomponente dauert. |
| $t_{cyc\text{-}in}$ | Maximale Zykluszeit einer Eingangskomponente |
| $t_{cyc\text{-}out}$ | Maximale Zykluszeit einer Ausgangskomponente |
| $t_{cycle}$ | Maximale Zykluszeit der Zyklusverarbeitung einer Komponente |
| $t_{in}$ | Das Alter eines Eingangssignals |
| $t_{logic}$ | Die Zeit, die die Sicherheitssteuerung mit ihrer Logik benötigt, bis sie auf den mit der Nachricht eingetroffenen Signalwert ein Ergebnis berechnet hat. |
| $t_{max}$ | maximale Reaktionszeit |
| TMO | Timeout |
| $t_{out}$ | Die Zeit, die die Ausgangskomponente (siehe DO in Abbildung 3.1) benötigt, bis sie das, mit der Nachricht eingetroffene, Ergebnis an ihrem Ausgang einstellt. |
| TS | Time Stamp |
| TSP | Transport Safety Protocol |
| $t_u$ | Die Unschärfe, mit der eine zeitliche Überwachung realisiert werden kann. |
| TÜV | Technischer Überwachungs-Verein |
| $t_{wc1}$ | Worst-Case-1 Reaktionszeit |
| $t_{wc2}$ | Worst-Case-2 Reaktionszeit |
| Tx | Identifikation einer Transition x eines Zustandsdiagramms |

| Kürzel | Bedeutung |
|---|---|
| UNID | Uniqe Network Identifier[277] |
| $v$ | Anzahl der Nachrichten einer Kommunikationsverbindung je Sekunde |
| vconsnr | Virtual Consecutive Number[278] |
| VLAN | Virtual Local Area Network |
| $WCDT_i$ | Worst-Case-Delay-Time der Komponente i bei PROFIsafe[278] |
| WCRT | Worst-Case Reaction Time |
| WCZZ | Worst-Case Zykluszeit |
| WDT | Watch-Dog-Time |
| $WDTime_i$ | Watch-Dog-Time der Komponente i bei PROFIsafe[278] |
| WDZ | Watch-Dog-Zeit |
| WiFi | Kunstbegriff, Wi-Fi als Synonym für WLAN genutzt[279] |
| WLAN | Wireless Local Area Network |
| ZZ | Zykluszeit |
| $\Lambda$ | Lambda, die Restfehlerrate eines sicherheitsgerichteten Kommunikationssystems |

[277] [CIP5]

[278] [IEC 61784-3]

[279] IEEE-802.11

# Literaturverzeichnis


[Adam04] Alanen Jarmo, Hietikko Marita, Malm Timo, "Safety of Digital Communications in Machines", ESPOO 2004, VTT Industrial Systems – Research Notes 2265, 2004

[Alan04] Adamski Bob, "Safety Bus Design Requirements for Process Industry Sector Applications", SAFETY USER GROUP, May 2004

[Ande11] Andersen B. Scott, Romanski George, "Verification of safety-critical software", Communications of the ACM, Volume 54, Issue 10, ACM, October 2011

[ASIs09] AS-i Safety, http://www.as-interface.net , 2012

[Baum05] Baumann R. C., "Radiation-induced soft errors in advanced semiconductor technologies" Device and Materials Reliability, 5(3):305–316, 2005

[Bart99] Barthel Herbert, Patentschrift EP1024639A1, „Verfahren und Einrichtung zur Bestimmung der Zuverlässigkeit von Datenübertragung", Siemens Aktiengesellschaft, 25. Mai 1999

[Bart99b] Barthel Herbert, Patentschrift WO99/49373, „Shortened Data Message of an Automation System", Siemens Aktiengesellschaft, 30. September 1999

[Benv10] Benveniste Albert, Bouillard Anne, Caspi Paul, "A unifying view of loosely time-triggered architectures", EMSOFT 2010, Proceedings of the tenth ACM international conference on Embedded software, ACM, October 2010

[Boer02] Börcsök J., „Netzwerke im industriellen Einsatz", VDE Verlag, 2002

[Boer04] Börcsök J., „Elektronische Sicherheitssysteme, Hardwarekonzepte, Modelle und Berechnung", Hüthig Verlag Heidelberg, 2004

[Boer06] Börcsök J., „Funktionale Sicherheit, Grundzüge sicherheitstechnischer Systeme", Hüthig Verlag Heidelberg, 2006

[Boer07] Börcsök J., Hannen H.-T., "Determination of Bit Error and Residual Error Rates for Safety Critical Communications", IEEE/IARIA - 2nd International Conference on Systems - 1st The First International Workshop on Safety in Industrial Systems (SAFESYS07), 22-28. April 2007, Sainte-Luce, Martinique, French Caribbean

[BoeSU] Börcsök J., „Safety Bus Systems", SAFETY USERS GROUP

[Boye09] Boyes W., "Safety, Security and Complex Systems in Critical Infrastructure Protection", SAFECOMP 2009, 28th International Conference on Computer Safety, Reliability and Security, Hamburg, Germany, September 15-18, 2009

[CANc04] CAN in Automation GmbH, „CANopen Safety Chip", Kurzbeschreibung 31.08.2004

[CANt04] TÜV Rheinland Group, "Bericht über die Prüfung des CANopen Safety Chip CSC01, Bericht-Nr. 968/EL 215.01/04, 30.07.2004

[CANs09] CAN in Automation GmbH, "CANopen Safety", http://www.can-cia.org, 2012

[Cast93] Castangnoli, Guy, Bräuer, Stefan, Herrmann, Martin, „Optimization of Cyclic Redundancy-Check Codes with 24 and 32 Parity Bits", IEEE Transactions on communications, Vol. 41, No. 6, June 1993

[CCLS] CC-Link Partner Association http://www.cc-link.org/eng/t_html/cclinksafety und Mitsubishi Electric Factory Automation - Americas, http://www.meau.com, "CC-Link Safety", 2012

[CIP5] Open DeviceNet Vendors Association, "THE CIP NETWORKS LIBRARY", Volume 5, „CIP Safety", Ed. 1.1, 2006, www.odva.org

[CIP5-2.2] Open DeviceNet Vendors Association, "THE CIP NETWORKS LIBRARY", Volume 5, „CIP Safety", Ed. 2.2, 2008, www.odva.org

[Dala93] Dalal S. R., Horgan J. R., Kettenring J. R., "Reliable software and communication: software quality, reliability, and safety", ICSE 1993, Proceedings of the 15th international conference on Software Engineering, IEEE Computer Society Press, May 1993

[Davi89] Davida, G. I., Desmedt, Y. G. & Matt, B. J. (1989), "Defending Systems Against Viruses through Cryptographic Authentication", Proceedings of the 1989 IEEE Symposium on Security and Privacy, Oakland, California, USA, IEEE Computer Society Press, pp. 312 – 318, May 1989

[DIN0801] DIN V EN VDE 0801, „Grundsätze für Rechner in Systemen mit Sicherheitsaufgaben", August 1998 (abgelöst im Jahr 2000 durch IEC 61508)

[DO178B] RTCA DO-178B, EUROCAE ED-12B, "Software consider-ations in airborne systems and equipment certification", RTCA Inc.,Washington, DC, 1992

[DO278] RTCA DO-278 "Guidelines For communication, navigation, surveillance, and air traffic management (Cns/Atm) systems soft-ware integrity assurance", RTCA Inc., Washington, DC, 2002

[EN 50128] DIN EN 50128, "Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Software für Eisenbahnsteuerungs- und Überwachungssysteme", Oktober 2011

[EN 50159-1] EN 50159-1, "Railway applications – Communication, signaling and processing systems – Part 1: Safety-related communication in closed transmission systems", November 2001

[EN 50159-2] EN 50159-2, "Railway applications – Communication, signaling and processing systems – Part 2: Safety-related communication in open transmission systems", December 2001

[EN 50159] DIN EN 50159, „Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Sicherheitsrelevante Kommunikation in Übertragungssystemen", April 2011 (Nachfolger von [EN50159-1] und [EN50159-2])

[EN 50325-5] DIN EN 50325-5, „Industrielles Kommunikationssubsystem basierend auf ISO 11898 (CAN) – Teil 5: Funktionale sichere Kommunikation basierend auf EN 50325-4", September 2009

[EN 60880] DIN EN 60880, „Kernkraftwerte – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorie A", März 2010

[EN 62138] DIN EN 62138, „Kernkraftwerte – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorien B oder C", März 2010

[EN954-1] DIN EN 954-1, „Sicherheit von Maschinen, Sicherheitsbezogene Teile von Steuerungen, Teil 1: Allgemeine Gestaltungsleitsätze“, März 1997. (abgelöst durch IEC 13489-1)

[EPLS] Bernecker + Rainer, „Ethernet-Powerlink Safety“, Bernecker + Rainer Industrie-Elektronik Ges.m.b.H., http://www.br-automation.com, sowie http://www.ethernet-powerlink.org, 2012

[Ferr02] Ferreira J., Pedreiras P., Almeida L., Fonseca J. A., “The FTT-CAN protocol for flexibility in safety-critical systems”, Micro, 22(4):46–55, 2002

[Fiel00] Fields Robert, Paternò Fabio, Santoro Carmen, Tahmassebi Sophie, “Communication Media in Cooperative Safety-Critical Contexts: A Method and a Case Study”, ACM Transactions on Computer-Human Interaction, Vol. 6, No. 4, Pages 370-398, December 1999

[Flex05] FlexRay Consortium, “FlexRay Communication System”, Protocoll Specification Version 2.1 Revision A, 22-December-2005

[Fuji89] T. Fujiwara, T. Kasami, S. Lin, “Error Detecting Capabilities of the Shortened Hamming Codes Adopted for Error Detection in IEEE Standard 802.3”, IEEE Transactions on Communications, Vol. 37, No. 9, Sept. 1989

[FFSA01] Foundation™ Specification: “System Architecture”, Revision FS1.1, September 31, 2001

[FFNM01] Foundation™ Specification: “Network Management”, Revision FS1.5, November 5, 2001

[FFSM99] Foundation™ Specification: “System Management”, Revision FS1.4, June 29, 1999

[FF-SIF] Foundation™ Specification: “FF-SIF Protocol Specification”, Revision FS1.1, July 13, 2007

[FFSA06] Foundation™ Specification: “FF-SIF System Architecture Overview”, Revision FS1.1, July 30, 2006

[FFSP06] Foundation™ Specification: “FF-SIS Application Model Phase1”, Revision PS1.1, December 7, 2006

[FFSG05] Foundation™ Specification: “FF-SIS User Application Guide”, Revision PS1.0, November 1, 2005

[FFSF07] Foundation™ Specification: “FF-SIS Function Blocks Phase 1”, Revision PS1.1, August 2, 2007

[FFSD07] Foundation™ Specification: “FF-SIS Device Development Requirements Specification”, Revision DPS0.1, September 14, 2007

[FSoE] Beckhoff, “Safety over EtherCAT”, EtherCAT Technology Group http://www.ethercat.org, sowie Beckhoff http://www.beckhoff.de, 2012

[FSoE07] Beckmann, Guido, „Die Sicherheitslösung für EtherCAT, Safety-over -EtherCAT“, ATP 9, 2007

[Gobl98] Goble, William M., “Control systems safety and reliability”, ISA, 2nd Edition, 1998

[GSET26] Fachausschuss Elektrotechnik, „Grundsatz für die Prüfung und Zertifizierung von Bussystemen für die Übertragung sicherheitsrelevanter Nachrichten“, Köln, Ausgabe 05.02, GS-ET-26

[Hann12] Hannen H.-T., Börcsök J., “Adressierungssicherheit von Kommunikationsprotokollen”, Anforderungen und wirksame Maßnahmen, ATP-Edition, Oldenburg-Industrieverlag, Heft 1-2/2012

[Hille99] Hillenbrand, W., “GSM-R The Railways Integrated Mobile Communication System”, Technical Report, Siemens, 1999

[Hütt07] Hüttinger Simon, Haller Herbert, Krämer Werner, Weichhold Peter, Wolski Jürgen, Patentschrift EP1763168A1, „Verfahren zum Erzeugen von Datentelegrammen, die CRC-Sicherungsanhänge aufweisen, welche eine verringerte Restfehlerwahrscheinlichkeit bieten“, Siemens Aktiengesellschaft, 4. August 2007

[Huss00] Hussey A., Leadbetter D., Lindsay P., Neal A., Humphreys M., “A Method for Analysing Hazards and Error Rates Related to Operator Activities”, Technical Report TR00-25, Software Verification Research Center, The University of Queensland, July 2000

[IBS] Phoenix-Contact, “INTERBUS-Safe”, INTERBUS-Club http://www.interbusclub.-com, sowie http://www.phoenixcontact.de, 2012

[IEC 13849-1] EN ISO 13849-1, “Safety of machinery – Safety-related parts of control systems – Part 1: General principles for design”,September 2002

[IEC 61131-x] IEC 61131, “Programmable Controllers”, parts 1 – 8, 2003

[IEC 61158-x] IEC 61158, “Industrial communication networks – Fieldbus Specifications”, all parts

[IEC 61508-x] IEC 61508, “Functional safety for electrical/electronic/programmable electronic safety related systems”, parts 1 – 7, Edition 2, 2010
Part 1: “General requirements”
Part 2: “Requirements for electrical/electronic/programmable electronic safety-related systems”
Part 3: “Software Requirments”
Part 4: “Definitions and Abbreviations”
Part 5: “Examples of methods fo the determination of safety integrity levels”
Part 6: “Guidelines on the application of IEC 61508-2 and IEC 61508-3”
Part 7: “Overview of techniques and measures”

[IEC 61511-x] IEC 61511, “Functional safety: Safety Instrumented Systems for the process industry sector”, 2003
Part 1: “Framework, definitions, system, hardware and software requirements”
Part 2: “Guidelines for the application of IEC 61511-1 ”
Part 3: “Guidance for the determination of the required safety integrity levels ”

[IEC 61513] DIN IEC 61513, „Kernkraftwerte – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Allgemeine Systemanforderungen”, Oktober 2002

[IEC 61784-x] IEC 61784-x, "Industrial Process Measurement and Control"
Part 1: "Profile sets for continuous and discrete manufacturing relative to field bus use in industrial control systems", FDIS
Part 2: "Additional profiles or ISO/IEC 8802-3 based communication networks in real-time applications", September 2003, Entwurf
Part 3: "Profiles for functional safety communications in industrial networks – General rules and profile definitions", 2010 (deutsche Fassung DIN EN 61784-3, Februar 2011)

[IEC 61784-3-3] IEC 61784-3-3, Industrial Process Measurement and Control, Part 3-3: „Profiles for functional safety communications in industrial networks – Additinal service and protocol specifications for CPF 3", June 2010. (PROFIsafeV2)

[IEC 62439] IEC 62439, „Digitale Datenkommunikation in der Leittechnik – Hochverfügbare Automatisierungsnetzwerke", April 2007

[IEEE802] IEEE 802 Standard for Local and Metropolitan Area Networks: "Overview and Architecture", February 7, 2002

[Kais09] Kaiser Alexander, Dolinar Sam, Cheng Michael, "Undetected Errors in Quasi-cyclic LDPC Codes Caused by Receiver Symbol Slips", Proceedings IEEE GLOBECOM, 2009

[Klub10] Klobedanz Kay, Kuznik Christoph, Thuy Andreas, Müller Wolfgang, "Timing modeling and analysis for AUTOSAR-based software development: a case study", DATE 2010, Proceedings of the Conference on Design, Automation and Test in Europe, ACM, March 2010

[Knig07] Knight John C., Graydon Patrik J., "Engineering, Communication, and Safety", SCS 2007, Proceedings of the twelfth Australian workshop on Safety critical systems and software and safety-related programmable systems - Volume 86 , Australian Computer Society, Inc., December 2007

[Koop02] Koopman, Philip, „32-Bit Cyclic Redundancy Codes for Internet Applications", Preprint to DSN02, 2002

[Koop04] Koopman, Philip & Chakravarty, Tridib, „Cyclic Redundancy Code (CRC) Polynomial Selection for Embedded Networks", DSN04, Juli 2004

[Koop06] Koopman, Philip, „Efficient High Hamming Distance CRCs for Embedded Networks", Preprint to DSN02, June 2006

[Korm04] Korkmaz G., Ekici E., Ozguner F., Ozguner U. Urban, "Multi-hop Broadcast Protocol for Inter-vehicle Communication Systems", Proceedings of the 1st ACM Workshop on Vehicular Ad Hoc Networks(VANET), pages 76–85, 2004.

[Korn05] Kornecki A., J. Zalewski, "Experimental Evaluation of Software Development Tools for Safety-Critical Real-Time Systems", Innovations in Systems and Software Engineering - A NASA Journal, Vol. 1, No. 2, pp. 176-188, 2005

[Korn09] Kornecki A., J. Zalewski, "Certification of software for real-time safety-critical systems: state of the art", Innovations System Software Engineering 2009, Springer-Verlag London Limited, April 2009

[Korn10] Kornecki Andrew J., Zalewski Janusz, “Safety and security in industrial control”, CSIIRW 2010, Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research, ACM, April 2010

[Koun05] Kounavis, M. E., Berry F. L., “A systematic approach to building high performance software-based CRC generators”, Proceedings of the 10th IEEE Symposium on Computers and Communications, 2005

[Kris02] Krishnan H., Kellum C., “Use of Communication in Vehicle Safety Application”, Internal Report of General Motors Company, 2002

[Krut96] Krut G., “Justification for the format of safety telegram”. AD-tranz corporation technical document, 1996

[Leve86] Leveson Nancy G., “Software Safety: Why, What, and How”, ACM Computing Surveys 18(2), pages 125– 163, 1986

[Leve95] Leveson Nancy G., “Safeware: System Safety and Computers”. Addison-Wesley, Boston, Mass., 1995

[Lu11] Lu Hongsheng, Poellabauer Christian, “Analysis of application-specific broadcast reliability for vehicle safety communications”, VANET 2011, Proceedings of the Eighth ACM international workshop on Vehicular inter-networking, ACM, September 2011

[Mak05] Mak Tony K., Laberteaux Kenneth P., Sengupta Raja, “A multi-channel VANET providing concurrent safety and commercial services”, VANET 2005, Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks, 2005

[Matt06] Mattes Tina, Schiller Frank, “An Efficient Method to Evaluate CRC-Polynomials for Safety-Critical Industrial Communication”, Journal Of Applied Computer Science, vol. 14, No. 1, 2006

[Matt07] Mattes Tina, Schiller Frank, Büttner T., Sachs H., J., “A New Method to Obtain Sufficient Independency of Nested Cyclic Redundancy Checks”, 5th International Conference Safety of Industrial Automated Systems, SIAS 2007, Tokyo, Japan, pp. 149–154, 2007

[Matt08] Mattes Tina, Schiller Frank, Mörwald Annemarie, Honold Thomas, “Analysis of Nested CRC with Additional Net Data in Communication”, SAFECOMP 2008, Springer, 2008

[Matt10] Mattes Tina, Schiller Frank, “Residual error probability of embedded CRC by stochastic automata”, SAFECOMP 2010, Springer-Verlag, September 2010

[Max06] Maxino, Theresa C., “The Effectiveness of Checksums for Embedded Networks”, Master Thesis – Department of Electrical and Computer Engineering, Carnegie Mellon University, 2006

[Maxe87] Maxemchuk, N.F., Sabnani, K.K., “Probabilistic verification of communication protocols” PSTV, pages 307–320, 1987

[Mentz06] Mentzel Martin., “Specification Safety related Adressing, Binding and Configuration for Safetylon”, Innotec GmbH, August 28nd 2006

[Merc06] Merchant Kamal, „Grenzwerte der Restfehlerwahrscheinlichkeit (Limits of the Residual Error Probability)“, DKE Internal Report, Oktober 2006

[Mich05] Michas, Christian, Bühler, Cornelia, "Zusammenstellung sicherheitstechnischer Anforderungen an Interfaces der Mess- und Stelltechnik in software-basierten Leittechniksystemen mit sicherheitstechnischer Bedeutung in Kernkraftwerken", TÜV Industrie Service GmbH TÜV SÜD Gruppe Energie und Technologie, Abschlussbericht SR2499, November 2005

[Misr82] Misra J., Chandy K. M., Smith T., "Proving Safety and Liveness of Communicating Processes, with Examples", Proceedings of the ACM SIGOPS/SIGACT Conference on the Principles of Distributed Computing, Ottawa, Canada, August 18-20, 1982

[Mont01] Monteiro F., Dandache A., M'sir A., Lepley B., "A fast CRC implementation on FPGA using a pipelined architecture for the polynomial division", The 8th IEEE International Conference on Electronics Circuits and Systems, volume 3, pages 1231 – 1234, 2001

[Mura10] Kiyoshi Murata, Yohko Orito, "Japanese Risk Society: Trying to Create Complete Security and Safety Using Information and Communication Technology", SIGCAS Computers and Society, Volume 40, No. 2, June 2010

[NE97] NAMUR, NE97 – "Fieldbus for safety-related uses", 2003

[NORS11] NORSOK STANDARD, "Risk based maintenance and consequence classification", Z-008, Edition 3, June 2011

[Osse83] Ossefort Marty, "Proving safety properties for a general communication protocol", SIGCOMM 1983, Proceedings of the symposium on Communications Architectures & Protocols, ACM, April 1983

[Pei92] Pei T.-B., Zukowski C., "High-speed parallel CRC circuits in VLSI", IEEE Transactions on Communications, 40(4), page 653–657, 1992

[Pete81] Peterson, W. Wesley, „Error-Correction Codes", 2nd Edition 1981, MIT-Press

[Popp02] Manfred Popp, „The New Rapid Way to PROFIBUS DP", 2002. Order-No. 4.072

[Popp07] Manfred Popp, Industrial Communication with PROFINET, 2007. Order-No. 4.182

[PNIOP05] PROFIBUS INTERNATIONAL, PROFINET Specification: „PROFINET IO Application Layer Services Definition, Application Layer Protocol Specification", V2.0, April 2005.. Order-No. 2.332

[PNIOD06] PROFIBUS INTERNATIONAL, PROFINET Specification: „Application Layer protocol for decentralized periphery and distributed automation", V2.1, June 2006.. Order-No. 2.722

[PNIOS06] PROFIBUS INTERNATIONAL, PROFINET Specification: „Application Layer services for decentralized periphery and distributed automation", V2.1, June 2006.. Order-No. 2.712

[PROFIsafeV1] PROFIBUS INTERNATIONAL, PROFIBUS Specification: „PROFIsafe – Profile for Safety Technology", V1.30, June 2004.

[PROFIsafeV2] PROFIBUS INTERNATIONAL, PROFIBUS Specification: "PROFIsafe – Profile for Safety Technology on PROFIBUS DP and PROFINET IO", V2.5f - FDIS, December 2010

[Rahm06] Rahmani Mehrnoush, Hintermaier Wolfgang, Müller-Rathgeber Bernd, Steinbach Eckehard, "Error Detection Capabilities of Automotive Network Technologies and Ethernet - A Comparative Study", Munich University of Technology, 2006

[Raje10] Rajeev A. C., Mohalik Swarup, Dixit Manoj G., Chokshi Devesh B., Ramesh S., "Schedulability and end-to-end latency in distributed ECU networks: formal modeling and precise estimation", EMSOFT 2010, Proceedings of the tenth ACM international conference on Embedded software, ACM, October 2010

[Roma09] Romanski G., "Safe and Secure Partitioned Systems and Their Certification", Proc. WRTP 2009, 30 IFAC Workshop on Real-Time Programming, Mragowo, Poland, October 12-14, 2009

[safe**ethernet**] HIMA Paul Hildebrandt GmbH + Co. KG, safe**ethernet**, http://www.hima.de/Produkte/safeethernet_default.php, 2012

[SBp] PILZ, "SafetyBUS-p", PILZ GmbH & Co. KG, http://www.pilz.com , sowie Safety Network International e. V. http://www.safety-network.de (vormals SafetyBUS p Club International e.V.), 2012

[Schi08] Schiller Frank, Mattes Tina, Weber Uwe, Mattes Rainer, "Undetectable Manipulation of CRC Checksums for Communication and Data Storage", ChinacomBiz 2008, Hangzhou P.R. China, August 2008

[Schu10] Schulz Oliver, Peleska Jan, "Reliability analysis of safety-related communication architectures", SAFECOMP 2010, Springer-Verlag, September 2010

[SDSS07] SIEMENS, "SIMATIC S7 Distributed Safety", Projektieren und Programmieren, Programmier- und Bedienhandbuch, 10/2007

[Smit03] Smith J., Russell S., Looi M., "Security as a Safety Issue in Rail Communications", SCS 2003, 8th Australian Workshop on Safety Critical Systems and Software, Canberra, Australian Computer Society, Inc., 2003

[STML03] ALCATEL, ALSTOM, ANSALDO SIGNAL, BOMBARDIER, INVENSYS RAIL, SIEMENS, "STM FFFIS Safe Link Layer, ERTMS/ETCS – Class 1", Ref: SUBSET-057, ISSUE 2.2.0, 11. April 2003

[STMT03] ALCATEL, ALSTOM, ANSALDO SIGNAL, BOMBARDIER, INVENSYS RAIL, SIEMENS, "STM FFFIS Safe Time Layer, ERTMS/ETCS – Class 1", Ref: SUBSET-056, ISSUE 2.2.0, 19. June 2003

[Stev94] Stevens W. Richard, "TCP/IP Illustrated I: The Protocols", Addison-Wesley Longman, Amsterdam; February 1, 1994

[Stri08] Stripf Wolfgang, Barthel Herbert, "Guideline for the Assessment, Test and Certification of Bus Systems for the Transmission of Safety-related Messages", DKE UK914.1 Proposal on Meeting July 24., 2008

[Tana11] Tanasa Bogdan, Bordoloi Unmesh Dutta, Eles Petru, Peng Zebo, "Reliability-aware frame packing for the static segment of flexray", EMSOFT 2011, Proceedings of the ninth ACM international conference on Embedded software, ACM, October 2011

[Tane02] Tanenbaum Andrew S., „Computer Networks", 4th Edition, Prentice Hall, N.J., 2002

[TÜV06] TÜV Rheinland, "Report on the concept approval of Safetylon", Reoport-No. 968/EL 426.00/06, September 9th 2009

[TÜV10] TÜV Rheinland, „Bericht über das Konzept des Transport Safety Protokoll (TSP) der Firma HIMA Paul Hildebrandt GmbH + Co. KG", Bericht-Nr.: 968/EL 700.00/10, August 2010

[Wick94] Wicker, Stephen B., "Error Control Systems for Digital Communication and Storage", Prentice Hall, 29. July 1994

[Wilh08] Wilhelm Reinhard, Engblom Jakob, Ermedahl Andreas, Holsti Niklas, Thesing Stephan, Whalley David, Bernat Guillem, Ferdinand Christian, Heckmann Reinhold, Mitra Tulika, Müller Frank, Puaut Isabelle, Puschner Peter, Staschulat Jan, Stenström Per, "The worst-case execution-time problem—overview of methods and survey of tools", Transactions on Embedded Computing Systems, Volume 7, Issue 3, ACM, April 2008

[Wrat05a] Wratil P., "Specification Safety related System Structure for Safetylon", Innotec GmbH, September 22nd 2005

[Wrat05b] Wratil P., "Specification Safety, SRS, FMEA, Safety related Devices for Safetylon", Innotec GmbH, September 10nd 2005

[Wrat07] Wratil P., "Frame Format for Safetylon", Innotec GmbH, September 10nd 2007

[Yu11] Yu F. Richard, Boukerche Azzedine, "Security and quality of service (QoS) co-design in vehicular ad hoc networks with cooperative communications", DIVANet 2011, Proceedings of the first ACM international symposium on Design and analysis of intelligent vehicular networks and applications, ACM, November 2011

## Eigene Veröffentlichungen

[Boer07] Börcsök J., Hannen H.-T., "Determination of Bit Error and Residual Error Rates for Safety Critical Communications", IEEE/IARIA - 2nd International Conference on Systems - 1st The First International Workshop on Safety in Industrial Systems (SAFESYS07), 22-28. April 2007, Sainte-Luce, Martinique, French Caribbean

[Hann12] Hannen H.-T., Börcsök J., "Adressierungssicherheit von Kommunikationsprotokollen", Anforderungen und wirksame Maßnahmen, ATP-Edition, Oldenburg-Industrieverlag, Heft 1-2/2012

[Holu07] Holub P., Börcsök J., Hannen H.-T., "Models for Determining of MTTF for Safety Related Electronical Systems by Monte Carlo Simulation by means of a 2oo4-System", Safety and Reliability for Managing Risk, Safety and Reliability Conference (ESREL2007), 25 - 27 June, Stavanger, Norway