AnTeS-NeCom

Analysis of the Failure Propagation in the Network Communication of Digital I&C Systems with a Test System

GRS - 764

Gesellschaft für Anlagen-
und Reaktorsicherheit
(GRS) gGmbH

AnTeS-NeCom

Analysis of the Failure Propagation in the Network Communication of Digital I&C Systems with a Test System

Final Report

Christian Müller
Joachim Herb
Patrick Gebhardt
Jaroslaw Shvab

April 2024

Remark:

This report refers to the research project RS1590. which has been funded by the Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (BMUV).

The work was conducted by the Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) gGmbH.

GRS - 764
ISBN 978-3-910548-56-5

Keywords

AnTeS, Cyberattack, Digital Instrumentation and Control (I&C), Fault Injection, Network Communication, Sensitivity Analysis, Simulation, Test System

Abstract

The research and development project RS1590, funded by the Federal Ministry for Economic Affairs and Energy (BMWi) and later by the Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (BMUV), was dedicated to investigating the impact of communication failures in the networks of digital instrumentation and control (I&C) systems in nuclear power plants. This project builds upon previous and, in part, concurrent initiatives in which the Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) developed fundamental, model-based methods for analyzing the behavior of digital I&C in the event of failures (funding codes 3615R01343, 4718R01314, 4722R01215). A crucial component of this research work was the development of the Analysis and Testing System AnTeS, which includes both real and simulated I&C systems.

The methods applied within the framework of AnTeS include Failure Mode and Effects Analyses (FMEAs), automated impact analyses (an automation and extension of FMEA developed by GRS), Fault Tree Analyses (FTAs), and Monte Carlo simulations. These methods serve to identify and evaluate potential causes of failures and their impacts.

Modern network technologies and topologies, used for both internal and external communication in I&C systems, also play a central role. The influence of these technologies on the reliability and safety of the systems was specifically examined in this project to address gaps in the existing methods and in the application of AnTeS.

The primary goal of the project was to develop an in-depth understanding of network communication within I&C systems. This included the development of methods for fault injection into network communication and the subsequent examination of the impacts of such failures on the reliability of various model systems. For these investigations, existing model systems were expanded, and new systems were designed and analyzed.

Through sensitivity analyses, the impact of different parameters on system reliability was evaluated. The project's findings indicate that digital I&C systems in nuclear power plants are highly robust against network failures, and that these failures only have a marginal impact on the overall reliability of the systems. These insights contribute significantly to the further development of the GRS methodology.

Table of contents

Abstract ..... **I**

1 Introduction **1**

1.1 State of science and technology 1

1.2 AnTeS – the GRS analysis and test system 2

1.3 Notes on this report 7

2 Fault injection into the network communication of the test system **9**

2.1 Network communication in the test system 9

2.2 Fault injection into the network communication 13

2.3 Manipulation in the sense of cyberattacks (excursus) 15

3 Failure propagation in network communication **17**

3.1 Relevant failure modes 17

3.2 Method development and validation 19

3.3 Analyses of model systems 28

4 Summary and overall results **35**

References **37**

List of figures **39**

List of tables **41**

Abbreviations **43**

A Detailed and additional descriptions **45**

A.1 TXS2Simulink 45

A.2 Analyses with the network manipulators 50

A.3 Model systems used 58

B Explanations of terms ... 63

B.1 Broadcast in networks ... 63

B.2 Bytes and their representation ... 63

B.3 Ethernet ... 64

B.4 MAC address ... 64

B.5 OSI model/layers ... 65

B.6 Sniffing ... 66

B.7 Wireshark ... 66

1 Introduction

Nuclear power plants around the world today often use instrumentation and control (I&C) systems with digital equipment[1]. Due to their more complex architectures, hardware and the use of software, these systems are more difficult to check for faults than analog, hard-wired systems with similar functions.

1.1 State of science and technology

To date, there is still a lack of detailed, generally recognized verification procedures and requirements for the reliable use of digital I&C in nuclear power plants (see also /MCH 21/). Internationally, the evaluation of digital I&C is therefore an important research topic, which GRS has also been consistently pursuing for several years. At GRS, model-based approaches are essentially applied.

As part of the BMU[2] project 3615R01343 ("Development and testing of a tool for sensitivity analysis of fault effects in safety-relevant digital I&C") /MCH 18/, a model-based procedure was developed and tested to analyze the dynamic behavior of digital I&C when system-internal faults occur. Based on generic models of modern system architectures, sensitivity analyses were used to investigate the influence of different parameters (e.g., repair times, degree of redundancy) on the reliability of the systems.

As part of the BMU project 4718R01314 ("AnTeS") /MCH 21/, the methodology developed was taken up and, above all, expanded and validated. A decisive factor here was the development of the GRS analysis and test system (AnTeS), which includes both simulated and real I&C systems as well as process engineering simulations. This system provides a flexible test environment for analyses and research work to investigate, verify and validate digital I&C systems.

[1] The safety requirements for nuclear power plants ("SiAnf") of the Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection in the March 2015 version (/BMU 15/, /BMU 15a/) differentiate between computer-based and programmable I&C equipment. By definition, programmable devices consist of at least one discrete programmable component (the application function is realized by wiring or by component functions), whereas computer-based devices contain at least one processor, and the application function is stored in the memory. In the context of this project, the commonly used 'digital' is used instead.

[2] BMU - abbreviation for the former name of the Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (outdated)

The methods developed and applied at GRS prior to this project did not explicitly consider the influence of the (digital) network technologies used in modern I&C systems. The signal processing of digital I&C systems uses different network technologies and topologies for both internal and external communication, the reliability and safety of which was the focus of the project described here.

In the context of the project presented here, a distinction was made between internal and external network communication as follows:

- Internal network communication:
  - Exchange of information within the I&C system using network technologies, for example between different redundancies of the same I&C system, but also between different I&C systems.
- External network communication:
  - Communication between an I&C system and devices that are not directly necessary for the execution of the I&C functions during operation. Typically, for example, the communication of an I&C system with a service unit (e.g., for programming the system ("engineering") or for monitoring the I&C system).

As already mentioned, AnTeS was originally developed as part of the BMU project 4718R01314 /MCH 21/. However, further development is currently also taking place as part of the BMUV project 4722R01215 /GRS 23/ (in particular with regard to operational I&C systems and priority modules). AnTeS is described in more detail in the following section.

1.2 AnTeS – the GRS analysis and test system

The analysis and test system of GRS (AnTeS) is a modular platform of different tools and methods for investigations into I&C technology. AnTeS basically has four modules (see also Figure 1.1):

AnTeS-SIC

- AnTeS-SIC-real: real safety I&C system (SIC)
  - based on Teleperm XS hardware and software from Framatome

- AnTeS-SIC-sim: simulated safety I&C systems
  - based on Matlab/Simulink /MAT 23/

AnTeS-OIC

- AnTeS-OIC-real: real operational I&C system (OIC)
  - based on Simatic S7 hardware and software from Siemens
- AnTeS-OIC-sim: simulated I&C systems
  - based on Matlab/Simulink

AnTeS-PRIO

- AnTeS-PRIO-real: real priority modules (PRIO)
  - AV42, SPLM1
  - Generic priority module (GRS in-house development for AnTeS)
- AnTeS-PRIO-sim: simulated priority modules
  - based on Matlab/Simulink

AnTeS-FIELD

- AnTeS-FIELD-real: real process engineering systems
  - vessels, drives, measuring devices/sensors, valves, pumps
- AnTeS-FIELD-sim: simulated process engineering systems
  - SimGen, see /MCH 21/

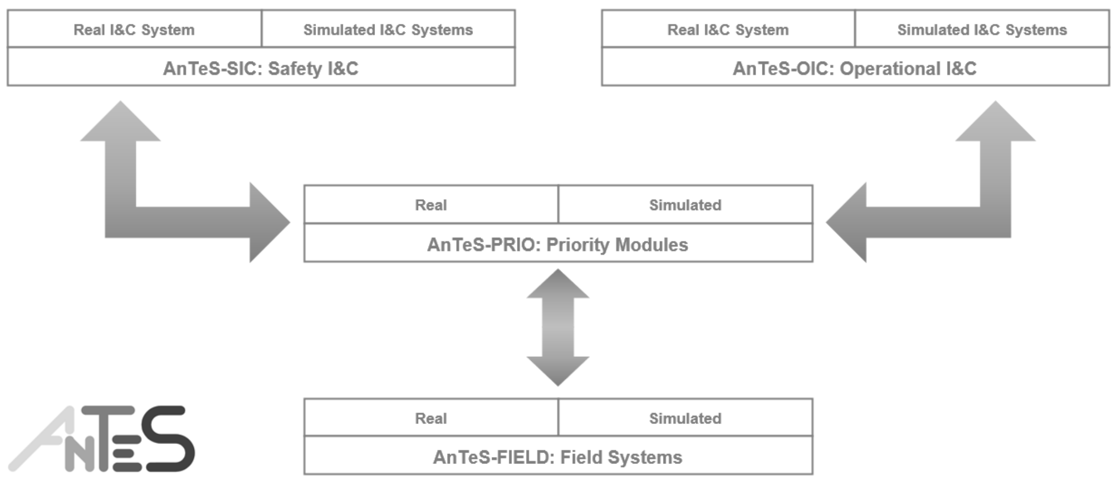


Figure 1.1 AnTeS overview

In addition, various analysis methods are available that can be used in conjunction with the AnTeS modules for investigations relating to I&C:

- FMEA – Failure Mode and Effects Analysis
    - FMEA is a systematic method for identifying, evaluating, and prioritizing potential faults or weaknesses in a product, process, or system. By analyzing possible causes of failure and the effects of these failures, FMEA helps to identify risks at an early stage and develop suitable measures to prevent, minimize or eliminate failures. The method is used in various sectors, including the automotive industry, aviation, medical technology, and the energy industry, to increase the reliability and safety of products and processes.
    - In the context of AnTeS and the methodology applied at GRS, FMEA is mainly used to determine the relevant failure modes (e.g., of components, subsystems) for further modeling. More detailed descriptions and further references can be found in /MCH 21/.
- Automatic impact analysis or failure effects analysis
    - This is an extended FMEA procedure that was developed as part of the 4718R01314 /MCH 21/ project. Here, a simulated or real system is used into which failures (e.g., of components or subsystems) can be injected with the aid of fault injection. By automatically varying all conceivable states of all considered parts of the system ("has failed self-reporting", "has failed non-self-reporting", "is functioning correctly") and

simultaneously recording the overall state of the system ("actuation of the safety function occurs as intended", "actuation of the safety function does not occur as intended"), all failure combinations can be determined in this way that are equivalent to an overall failure of the system.

  - In the context of AnTeS, the more comprehensive and less error-prone automatic impact analysis usually replaces a simple FMEA. The results of the automatic impact analysis in turn support the fault tree analysis. More detailed descriptions can be found in /MCH 21/.

- FTA - Fault tree analysis

  - Fault tree analysis is a systematic method for investigating potential causes of faults and their effects in complex systems. It visualizes the possible error paths in the form of a tree diagram, in which the top level represents the undesirable end state. By analyzing the failure paths step by step from the top of the tree to the root causes, critical weaknesses and potential combinations of events that could lead to an undesired event can be identified. Fault tree analysis is a powerful tool used in various industries to assess risks, develop safety measures, and improve the reliability of complex systems. By incorporating probabilities and data on individual events, fault tree analysis also enables the quantitative assessment of risks and the derivation of probabilities for the occurrence of undesirable events, which provides a sound basis for decision-making, for example for preventive measures.

  - In the context of AnTeS, fault tree analyses provide the same qualitative results as automatic impact analyses (whereby the two methods check each other). In addition, fault tree analyses can also be used to obtain quantitative results on the analyzed systems. More detailed descriptions and further references can be found in /MCH 21/. Comparable quantitative results can also be obtained with Monte Carlo simulations.

- Monte Carlo simulation

  - Monte Carlo simulations are a computer-aided method used in various fields to analyze complex problems for which analytical solutions are difficult or impossible. This method is based on random sampling and repeats the analysis of a model or system thousands or even millions of times, each time taking into account random variations in the input parameters. The results of these simulations provide statistical distributions of possible outcomes and enable the estimation of probabilities, risks, and other quantitative information. Monte Carlo simulations are used in finance, engineering, natural sciences, risk analysis and many other disciplines to get a better idea of the possible outcomes of complex systems or models.

- In connection with AnTeS, simulated I&C systems are used for Monte Carlo simulations, into which statistical failures of certain components are fed by fault injection. Quantitative results comparable to fault tree analyses can be achieved. Thus, Monte Carlo simulations can completely replace fault tree analyses in individual cases or at least verify their results. More detailed descriptions and further references can be found in /MCH 21/.

By combining real or simulated modules into an overall system, different configurations and I&C architectures can be flexibly implemented depending on the requirements and investigated using the available methods (see Figure 1.1).

For this project, only the AnTeS-SIC module was used (real and simulated safety I&C systems). Figure 1.2 shows the real AnTeS I&C system (TXS) in the left-hand image. The three TXS cabinets available at GRS can be seen in a closed state. Typically, however, only the left and middle cabinets are used for tests at GRS (right-hand image in Figure 1.2); the third cabinet serves as a reserve and is still in the state in which GRS took over all cabinets from the Krümmel nuclear power plant in 2017 (for details, see /MCH 21/).

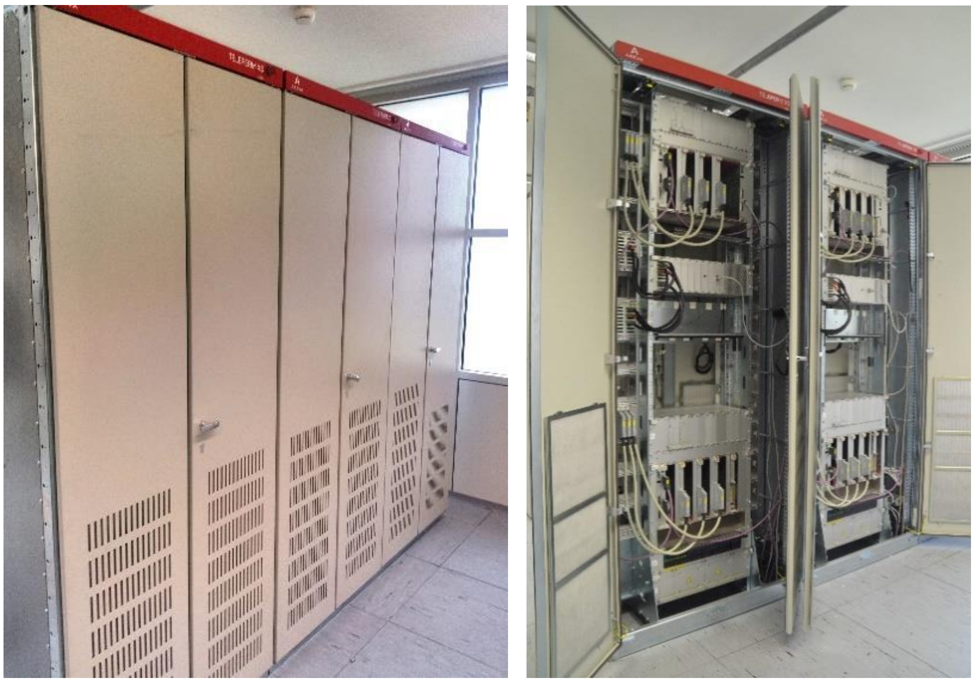


Figure 1.2 The AnTeS-SIC-real module, a real safety I&C system based on Teleperm XS

1.3 Notes on this report

This report has been structured in such a way that it is as clear and easy to read as possible. Therefore, only the relevant facts and results are presented in the main text, while more detailed descriptions (e.g., on the tests carried out) can be found in appendix A. Appendix A also contains additional information on work within the project that is not mentioned elsewhere (e.g., creation of software for the automatic conversion of TXS function diagrams into Matlab/Simulink simulation models, section A.1). In addition, appendix B explains some basic terms from the field of digital networks that were used in the main text but are not explained in detail there. These explanations may make it easier for the reader to understand the context.

2 Fault injection into the network communication of the test system

In order to be able to analyze the effects of failures in the network communication of I&C systems, a series of different tests were carried out using a (real) test system (specifically: AnTeS-SIC-real module).

The following section 2.1 will first of all take a general look at the network communication within this test system. Section 2.2 will then describe the possibilities for injecting failures into the network communication of the test system.

Although not the explicit aim of this project, the developed fault injection possibilities can also be used for the execution (or simulation and investigation) of cyberattacks. A brief excursus on this can be found in section 2.3.

2.1 Network communication in the test system

The test system used (AnTeS-SIC-real) is based on hardware and software components from the Teleperm XS (TXS) I&C platform of Framatome. For all developments, tests, and analyses, this was configured in such a way that the entire network communication (external and internal) of the test system took place exclusively via Ethernet (in accordance with IEEE 802.3)[3].

Through specific tests using the Wireshark /WIS 23/ software, important fundamental insights into the network communication of the test system were obtained. When communicating via Ethernet in the test system, information is generally exchanged unidirectionally between communication partners. This also means, for example, that the recipient of a message does not acknowledge it in any way. On the one hand, this avoids any unwanted feedback from the receiver to the sender, but on the other hand, the sender cannot usually draw any conclusions about the correct receipt of the transmitted data.

[3] Typically, TXS generation 2 used in the AnTeS-SIC-real module uses Profibus for internal communication between units or redundancies in the I&C system. However, many I&C systems from other manufacturers and Teleperm XS generation 4 (currently under development) tend to use Ethernet /FRA 23/ as standard, which was therefore used as a reference for the work in this project.

Network communication in the test system is close to the hardware and does not use any higher-level protocols (such as for example the Internet Protocol IP). All data packets sent within the test system can be schematically represented on OSI layer 2 (see appendix B.5 for the definition of OSI layers) as shown in Figure 2.1[4].

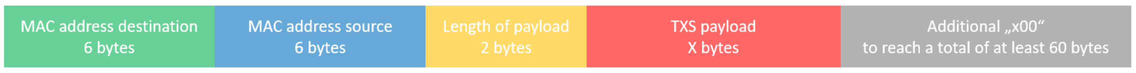


Figure 2.1 Schematic representation of data packets in the Ethernet communication of the test system (TXS, Generation 2)

A concrete TXS Ethernet data packet thus looks like Figure 2.2, for example, if the transmitted bytes are represented by hexadecimal numbers (indicated by a prefixed "x") (for the representation of bytes as hexadecimal numbers, see also appendix B.2).

```
x08 x00 x06 x01 xa0 x01 x08 x00 x06 x01 xa0 x00 x00 x1f
x14 x14 x03 x02 x00 xc0 x88 x05 x00 x44 x81 x0b x00 x0c
x00 x1c x00 x00 x01 x02 x1f x00 x00 x65 x00 x01 x00 x00
x02 x01 x00 x00 x00 x00 x00 x00 x00 x00 x00 x00 x00 x00
x00 x00 x00 x00
```

Figure 2.2 Example of a TXS data packet (within the test system)

The data packet in Figure 2.2 contains the following data/information in detail:

- MAC address of the destination:
  - x08 x00 x06 x01 xa0 x01 resp. 08-00-06-01-a0-01
- MAC address of the source:
  - x08 x00 x06 x01 xa0 x00 resp. 08-00-06-01-a0-00
- Length of the transmitted data (i.e., the payload):
  - 31 Bytes (x00 x1f = 31)

[4] Please note that the Ethernet used in TXS still complies with an early version of the IEEE 802.3 standard. Ethernet data packets in newer or current computer networks are generally structured somewhat differently today.

- Payload - the actual transmitted data (31 bytes):
  - `x14 x14 x03 x02 x00 xc0 x88 x05 x00 x44 x81 x0b x00 x0c x00 x1c x00 x00 x01 x02 x1f x00 x00 x65 x00 x01 x00 x00 x02 x01 x00`
  - In addition to the transmitted TXS data (e.g., values of variables, etc.), this payload also contains:
    - A counter that counts up from data packet to data packet.
    - A checksum that is calculated over part of the payload (incl. counter).
- Appended "x00" so that the total length of 60 bytes required by the specification is reached.[5]

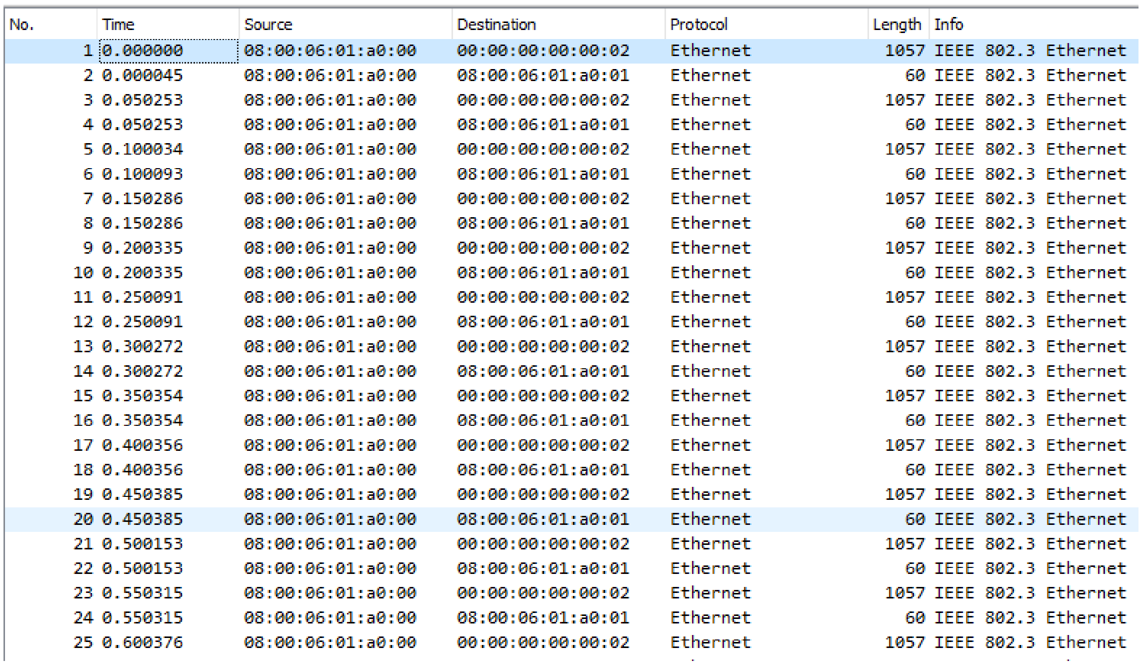
Through systematic tests (with Wireshark /WIS 23/) and in particular by comparing different data packets (at different times and for different model systems), the entire network communication could be fully analyzed and understood. This also made it possible to create software that can be used, for example, to correctly calculate and set both the counter and the checksum present in the payload (see also sections 2.2 and 2.3 below).

The statements made so far are not only valid for internal network communication. Apart from special cases (see end of this section), external network communication works in exactly the same way.

If a service unit (or alternatively a so-called gateway for the "decoupling" of signals) is present in the network plan of the TXS software (in the TXS engineering environment SPACE[6]), messages are only sent (unidirectionally) from the I&C system to this external device (and not vice versa).

[5] In this example, only a comparatively small data packet was transmitted. To ensure that the data packet was at least 60 bytes long in accordance with the specification (IEEE 802.3), a further 15 bytes were appended with zeros. This is not necessary for longer payloads.

[6] SPACE – **SP**ecification **A**nd **C**oding **E**nvironment (TXS engineering environment)



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
2	0.000045	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
3	0.050253	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
4	0.050253	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
5	0.100034	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
6	0.100093	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
7	0.150286	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
8	0.150286	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
9	0.200335	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
10	0.200335	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
11	0.250091	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
12	0.250091	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
13	0.300272	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
14	0.300272	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
15	0.350354	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
16	0.350354	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
17	0.400356	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
18	0.400356	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
19	0.450385	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
20	0.450385	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
21	0.500153	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
22	0.500153	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
23	0.550315	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet
24	0.550315	08:00:06:01:a0:00	08:00:06:01:a0:01	Ethernet	60	IEEE 802.3 Ethernet
25	0.600376	08:00:06:01:a0:00	00:00:00:00:00:02	Ethernet	1057	IEEE 802.3 Ethernet

Figure 2.3 Example of messages sent from one redundancy (of AnTeS-SIC-real)

Specifically, one redundancy of AnTeS-SIC-real (with the MAC address 08-00-06-01-a0-00, "source") sends alternately to another redundancy of AnTeS-SIC-real (with the MAC address 08-00-06-01-a0-01, "destination") and to the service unit (with the MAC address 00-00-00-00-00-02[7], "destination"). The recording was made with the Wireshark software (see appendix B.7).

Figure 2.3 shows this context for a specific example. Here, a message was sent from one redundancy of the I&C system to another redundancy of the I&C system and also to the service unit in each cycle (i.e., every 50 ms). The basic structure of the data packets to the service unit is identical to the structure of the data packets described above. The only difference is that significantly more information is transmitted (in this case 1057 bytes in each case), as the status of all parameters (e.g., the function blocks in the function diagrams) must be transmitted to the service unit in the corresponding software of the sending redundancy (e.g., for visualization in the GSM graphical service monitor).

As already mentioned, communication (as in the example) with the service unit is basically unidirectional. In particular, no requests or feedback are transmitted from the service unit to the I&C system. With the service unit configured in the I&C system software, there is therefore only a permanent transmission of information from the I&C system to

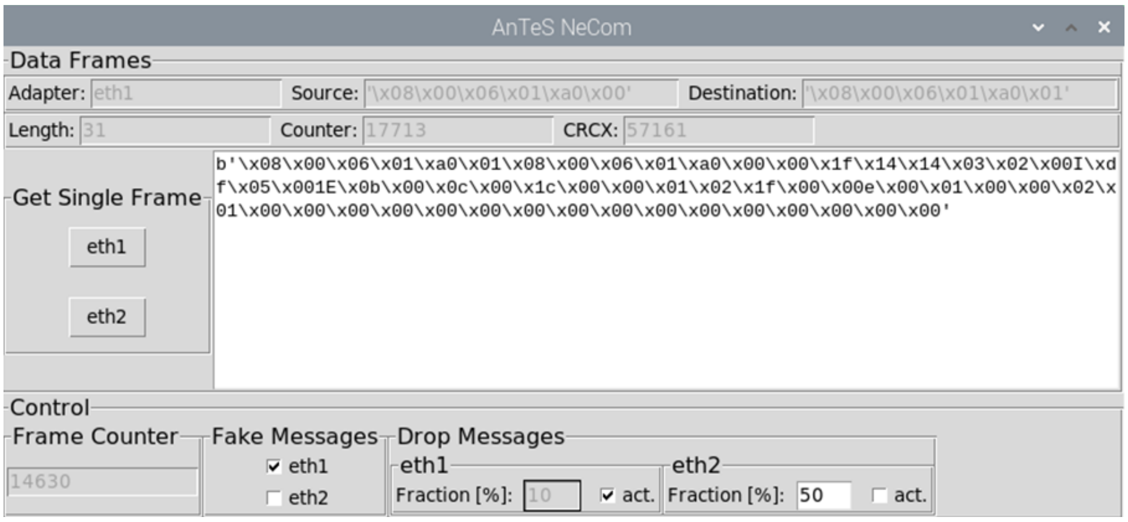
[7] The MAC addresses are configurable in TXS; within AnTeS, the service device has exactly this address by default.

the service unit (regardless of whether the information is used for visualization or not, for example) and not vice versa.

An exception to this is the transfer of new/modified software from the service unit to the I&C system. However, special prerequisites must apply for this, which need not to be assumed to be given in a safety relevant I&C system (e.g., reactor protection system).[8]

2.2 Fault injection into the network communication

Based on the findings on network communication in the test system (see previous section 2.1), software was developed (Figure 2.4) that runs on specially designed devices (based on Raspberry Pi 4 microcomputers, Figure 2.5) and allows the controlled injection of failures into the network communication. Together, software and microcomputers thus form devices for the systematic influencing of any network communication (hereinafter referred to as network manipulators).



Adapter: eth1 Source: '\x08\x00\x06\x01\xa0\x00' Destination: '\x08\x00\x06\x01\xa0\x01'
Length: 31 Counter: 17713 CRCX: 57161

```
b'\x08\x00\x06\x01\xa0\x01\x08\x00\x06\x01\xa0\x00\x00\x1f\x14\x14\x03\x02\x00I\xd
f\x05\x001E\x0b\x00\x0c\x00\x1c\x00\x00\x01\x02\x1f\x00\x00e\x00\x01\x00\x00\x02\x
01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
```

Frame Counter: 14630
Fake Messages: [✓ eth1] eth2
Drop Messages: eth1 Fraction [%]: 10 [✓ act.] eth2 Fraction [%]: 50 act.

Figure 2.4 Developed software for reading ("sniffing") and influencing the network communication of the test system

More details on this software can be found in appendix A.2.2.

---

[8] To enable the upload of new/modified software via the external network connection, corresponding jumpers or switches must be set on the hardware of the processing units (processor cards) of the I&C system (TXS) and a function block must be present in the running software (function diagrams) of the processing units that explicitly grants the corresponding rights (see also /MCH 21/). By default, uploading software via the external network connection is completely excluded; instead, software upload is only enabled via a direct serial connection of a service unit on site. The same applies to the mere modification of parameters in the software of the control system by the service unit.

In total, GRS has five network manipulators (including the prototype, which only differs externally from the four devices in Figure 2.5). Each of these devices has three Ethernet connections, one internal Ethernet connection and two external network cards connected via USB. The internal Ethernet ports are used exclusively to control and monitor the network manipulators, while the external Ethernet ports can be used to forward all network traffic in both directions, read it and change it if desired.

Each network manipulator can be inserted into any Ethernet connection of any system. All data packets that would have been sent via the original cable are then routed through the network manipulator (in both directions). In addition, the entire network traffic can also be read or even specifically influenced.

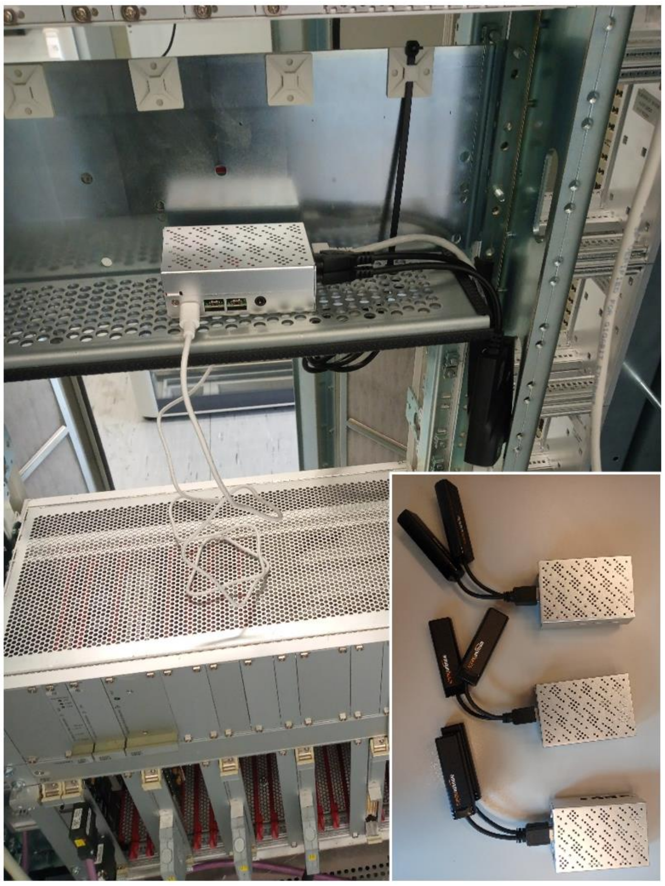


Figure 2.5 Devices for manipulating network communication (for error injection) developed as part of this project

In addition to the undisturbed forwarding of data packets, there are the following possibilities for manipulating network traffic:

- Forwarding of only a proportion of the data packets (e.g., a proportion of x % of randomly selected data packets or only every nth data packet).
- Changing an adjustable number of bits of each data packet.
  - Their positions can either be randomly selected or fixed.
- Creating your own data packets ("fake messages").
  - With the help of previously recorded data packets, for example, valid data packets can be generated that are evaluated as genuine and valid by the receiver.
    - In particular, the counter contained in the TXS payload, and the checksum also contained therein must be calculated and set correctly for each "fake" data packet.
  - In principle, any network manipulator can therefore completely replace the "real" sender from the receiver's point of view.

In terms of random failures in network communication, only the first two manipulation possibilities mentioned are relevant. The third possibility represents a manipulation possibility in the sense of cyberattacks, which is briefly examined in the following excursus.

2.3 Manipulation in the sense of cyberattacks (excursus)

The network manipulators also allow so-called man-in-the-middle attacks (MitM attacks). MitM attacks are a form of cyberattack in which an attacker intercepts, manipulates, or even completely controls the communication between two parties without the parties involved realizing it. The attacker places himself "in the middle" of the communication link, so to speak, and can intercept and modify the data traffic or even smuggle in falsified information.[9]

[9] MitM attacks in a general context can occur, for example, on public Wi-Fi networks, insecure websites, or other insecure communication channels. MitM attacks have the potential to steal confidential information, capture passwords, manipulate financial transactions or even compromise the integrity of data. To prevent such attacks, the use of secure encryption protocols and awareness of suspicious activity in communications is crucial.

In specially conducted tests, which are not explained in detail here (as this was not an objective of the project), the capabilities of the developed network manipulators to carry out such attacks were tested and verified using specific examples. The decisive factor here was that correct counters and checksums could be calculated and set in the TXS payload for modified or completely falsified data packets (see section 2.1). In summary, it can be stated that it was possible, for example, to record valid data packets and then send them to the recipient again and again (with recalculated counters and checksums in the TXS payload), so that the recipient of these data packets could be completely decoupled from the real sender of the data packets without being noticed.

3 Failure propagation in network communication

This chapter deals with the development, application, and validation of the GRS methodology for the investigation of digital I&C systems, which has been extended to include the consideration of network failures. In particular, by applying the extended methodology to a series of model systems (see section 3.3), it was possible to draw some generally valid conclusions about the significance and effects of network failures on I&C systems.

3.1 Relevant failure modes

With the help of the developed network manipulators (see section 2.2) and the real I&C system (AnTeS-SIC-real, TXS) available at GRS, the relevant failure modes within the network communication of the test system were first generally determined, which then had to be taken into account in the method development. The corresponding tests are explained in more detail in appendix A.2; only the results of these tests are summarized here.

The test system (TXS) proved to be extremely robust in terms of network communication failures. Up to around 50 % of the data packets sent can be completely lost or faulty without any significant impact on functionality. It is irrelevant whether, for example, exactly every second data packet is affected or whether, purely statistically, 50% of the data packets are changed or not forwarded by the network manipulators. Only with even higher loss rates of data packets can a "flickering" (alternating between apparently undisturbed behavior and apparently interrupted communication) be observed, and finally, with even higher loss rates (~ 70 %), the communication is assessed as completely failed on the receiver side.[10]

Further tests carried out showed that statistical changes to the data packets (e.g., by randomly changing one or more bits) are detected extremely reliably by the test system. Theoretically, faulty but valid data packets can also occur by chance, which then also lead to faulty, undesired behavior. However, their unintentional occurrence is so unlikely

[10] The numbers given are only approximate values. Presumably due to the asynchronous operation of the two redundancies, the actual values varied slightly in individual cases between the different tests.

that they can be excluded with a high degree of probability.[11] This is because the checksum of the TXS contained in the user data (payload) of the data packet must still be correct, the destination and source addresses in the packet must not have been changed, and the identifiers and variable names contained in the user data must not have changed, too.

If we assume, for example, that a bit is randomly changed in every single transmitted data packet and that a single transmitted binary value (e.g., of a transmitted binary variable) is randomly changed in such a way that a valid data packet is still created by chance, the probability of this can be (roughly) estimated as follows:

Probability for the change of a bit (of a data packet):

$$P_{bf} = 1 \text{ (by definition, see above)}$$

Probability that the change will occur at the location of the binary value:

$$P_{bp} = {}^{1}/_{480} \text{ (minimum length of the packet = 60 bytes = 480 bits)}$$

Probability that the calculated checksum (TXS) is still correct[12]:

$$P_{CS} = 2 \cdot 10^{-5}$$

Total probability (for a single data packet):

$$P = P_{bf} \cdot P_{bp} \cdot P_{CS} \approx 8{,}5 \cdot 10^{-8}$$

With a (typical) cycle time (time interval between two data packets) of 50 ms, a single packet would then only transmit a faulty value undetected by chance every 7 days or

[11] Please note that we are talking about accidental failures here. Deliberate changes (e.g., in the sense of a cyberattack) are not meant here (see section 2.3).

[12] The value used actually describes the probability for the algorithm used to calculate the checksum that two different data packets result in the same checksum by chance (value from /TXS 12/).

so.[13] All other packages would be recognized as faulty. The probability of this occurring several times in succession is of course much lower.

This is only a rough estimate and does not claim to be a universally valid consideration. Nevertheless, it illustrates why such errors can be practically ruled out.

Overall, a single network connection can therefore be assumed only in two states during modeling:

- communication is working as expected
- communication is self-reporting failed (detected by the I&C system)

This means that only one relevant failure mode for network connections needs to be taken into account in the further development of the methods.

3.2 Method development and validation

This section uses a simple model system as an example to explain the extension and validation of the GRS methods for analyzing digital I&C systems. The starting point is the model system A120[14] (Figure 3.1).

[13] Note that every single data packet contains a faulty bit. In this case, the system would report a failure for around 7 days before transmitting a single incorrect but valid data packet.

[14] The nomenclature used and the A120 model system were already developed and used in project 4715R01343 /MCH 18/ (at that time, however, without explicit failures in the network communication paths). A more detailed description of all model systems used in this project can be found in appendix A.3.

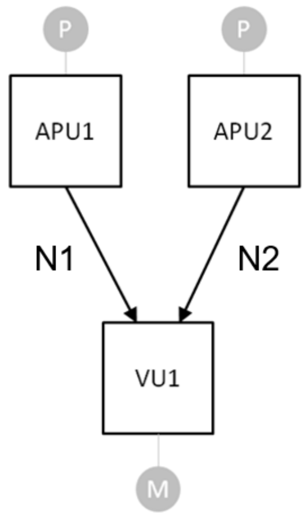


Figure 3.1 Model system A120

This consists of two APUs (<u>a</u>cquisition and <u>p</u>rocessing <u>u</u>nits) and one VU (<u>v</u>oting <u>u</u>nit) and can be described as follows:

- Two transducers ("P" for pressure measurement in Figure 3.1), which for simplicity's sake are assumed to always function faultlessly and transmit their signals to the top level of the I&C system without failures, are each connected to an APU.
- The input signals (measured values) are read in by the APUs and monitored for exceeding a MAX limit value. If the limit value is exceeded, the respective APU outputs a logical "1", otherwise a logical "0".
- The output signals generated by the APUs are transmitted via two separate network connections between the APUs and the VU1 (N1 and N2 in Figure 3.1)
- The voting unit VU1 evaluates the input signals with a 1-out-of-2 selection (“OR”). If one or two signals with a logical "1" are present at the input of the VU1, it issues a start command to the connected motor (M).
- The connected motor always responds correctly to the signals from the VU1.

Furthermore, it is assumed for the sake of simplicity that both the APUs and the VU can only fail non-self-reporting.[15] Their non-self-reporting failures (NSF) are therefore not

[15] This simplified approach only applies to this example. When the methods were later applied to more complex model systems, self-reporting failures of all components were also taken into account.

detected automatically and can only be detected and subsequently repaired by means of specific tests (recurring tests).

In contrast, failures in network communication are always self-reporting failures (SF) - see section 3.1. All parameters for the complete description of the A120 model system are shown in Table 3.1.[16]

Table 3.1 Parameters used for model system A120

Parameter	Description	Value
APU1.NSF	Failure rate for non-self-reporting failures of APU1	$8 \cdot 10^{-8}\ h^{-1}$
APU2.NSF	Failure rate for non-self-reporting failures of APU2	$8 \cdot 10^{-8}\ h^{-1}$
VU1.NSF	Failure rate for non-self-reporting failures of VU1	$8 \cdot 10^{-8}\ h^{-1}$
N1.SF	Failure rate for self-reporting failures of N1	$2 \cdot 10^{-5}\ h^{-1}$
N2.SF	Failure rate for self-reporting failures of N2	$2 \cdot 10^{-5}\ h^{-1}$
MTTR	Repair time for all detected failures (MTTR – mean time to repair)	8 h
TI	Test interval (i.e., the time between two tests within a redundancy): 6 x 30 days = 4320 h	4320 h
TF1[1)]	Time until the first test of redundancy 1 (APU1 und VU1)	0 h
TF2[1)]	Time until the first test of redundancy 2 (APU2): 3 x 30 days = 2160 h	2160 h
[1)] Recurring tests are therefore carried out every three months, alternating between redundancy 1 and redundancy 2.		

In accordance with the GRS procedure, the first analysis step is typically a failure effects analysis (an extended FMEA - failure mode and effects analysis). In this analysis, all

[16] The specific values have been assumed arbitrarily (albeit plausibly). In this example, only the basic principles are explained.

combinations of all conceivable individual states of the components considered are recorded in tabular form and each of these combinations is evaluated with regard to the overall failure of the entire system.

Table 3.2 (without network faults) and Table 3.3 (with network faults) show an example of this for the model system A120. Accordingly, the number of combinations to be considered increases from 8 to 32 if the failure possibilities of the network communication (N1, N2) are also taken into account.

Table 3.2 Failure effects analysis for A120 (without network failures)

#	APU1	APU2	VU1	Overall failure?
1	OK	OK	OK	no
2	NSF	OK	OK	no
3	OK	NSF	OK	no
4	NSF	NSF	OK	yes
5	OK	OK	NSF	yes
6	NSF	OK	NSF	yes
7	OK	NSF	NSF	yes
8	NSF	NSF	NSF	yes
NSF – non-self-reporting failure				

Table 3.3 Failure effects analysis for A120 (with network failures)

#	APU1	APU2	N1	N2	VU1	Overall failure?
1	OK	OK	OK	OK	OK	no
2	NSF	OK	OK	OK	OK	no
3	OK	NSF	OK	OK	OK	no
4	NSF	NSF	OK	OK	OK	yes
5	OK	OK	SF	OK	OK	no
6	NSF	OK	SF	OK	OK	no
7	OK	NSF	SF	OK	OK	yes
8	NSF	NSF	SF	OK	OK	yes
9	OK	OK	OK	SF	OK	no
10	NSF	OK	OK	SF	OK	yes
11	OK	NSF	OK	SF	OK	no
12	NSF	NSF	OK	SF	OK	yes
13	OK	OK	SF	SF	OK	yes
14	NSF	OK	SF	SF	OK	yes
15	OK	NSF	SF	SF	OK	yes
16	NSF	NSF	SF	SF	OK	yes
17	OK	OK	OK	OK	NSF	yes
18	NSF	OK	OK	OK	NSF	yes
19	OK	NSF	OK	OK	NSF	yes
20	NSF	NSF	OK	OK	NSF	yes
21	OK	OK	SF	OK	NSF	yes
22	NSF	OK	SF	OK	NSF	yes
23	OK	NSF	SF	OK	NSF	yes
24	NSF	NSF	SF	OK	NSF	yes
25	OK	OK	OK	SF	NSF	yes
26	NSF	OK	OK	SF	NSF	yes
27	OK	NSF	OK	SF	NSF	yes
28	NSF	NSF	OK	SF	NSF	yes
29	OK	OK	SF	SF	NSF	yes
30	NSF	OK	SF	SF	NSF	yes
31	OK	NSF	SF	SF	NSF	yes
32	NSF	NSF	SF	SF	NSF	yes

NSF – non-self-reporting failure
SF – self-reporting failure

In more complex systems, the failure effects analysis can on the one hand facilitate the fault tree generation, but on the other hand at least always make the correct modeling verifiable during the fault tree analysis (since the combinations that lead to an overall failure of the system according to the failure effects analysis must appear as a minimal cut set in the fault tree analysis).[17]

As explained in the report on project 4718R01314 /MCH 21/, such failure effects analyses can also be carried out fully automatically with the help of simulated or real I&C systems. For this purpose, either a simulation model (using Matlab/Simulink) of the I&C system to be investigated is created or the system to be investigated is realized using a real test system. All conceivable failure combinations are then automatically set in the respective models (fault injection) and the behavior (actuation: yes/no) is recorded.

When implemented with AnTeS-SIC-real (TXS), the model system A120 looks as shown in Figure 3.2 (with additional network manipulators inserted into the network connections between the APUs and the VU). The corresponding simulation model (Matlab/Simulink) is shown in Figure 3.3 (further information on the model systems can be found in appendix A.3).

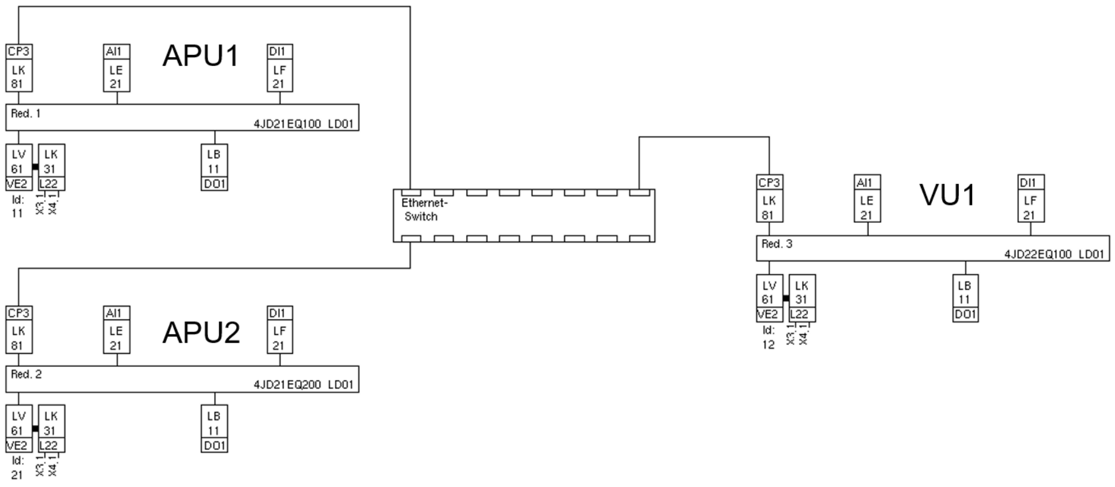


Figure 3.2 A120 realized with AnTeS-SIC-real (TXS), network plan

[17] Note that there are usually fewer cut sets in the fault tree analysis than entries in the table for a failure effects analysis that lead to an overall failure. For example, sequential numbers 5 to 8 in Table 3.2 all belong to the same minimal cut set “VU1 has NSF” (non-self-reporting failure).

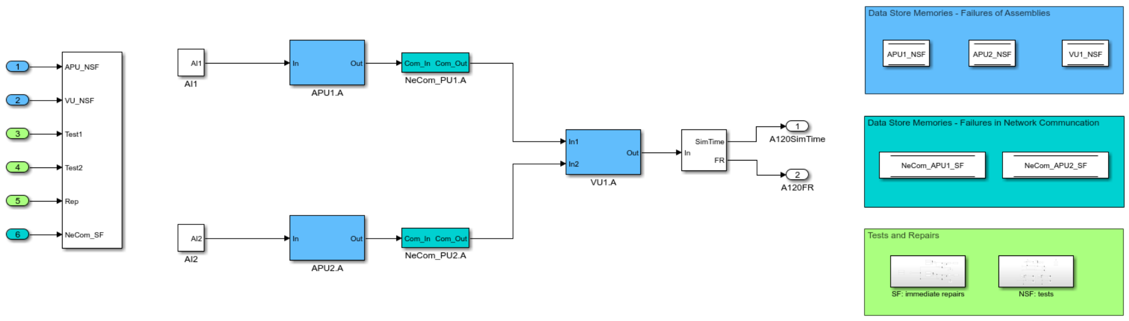


Figure 3.3 Matlab/Simulink model of A120

Both variants (simulated and real) deliver exactly the same results for the model system A120 in automatic failure effects analyses like the previously shown manually filled tables (Table 3.2 and Table 3.3).

For quantitative analyses, fault trees or alternatively simulation models (in Monte Carlo simulations) can be used for the model system under consideration. Figure 3.4 shows the fault tree created (with the RiskSpectrum software) for the model system A120, albeit without basic events for failures in network communication. If these are also taken into account, the fault tree in Figure 3.5 results.

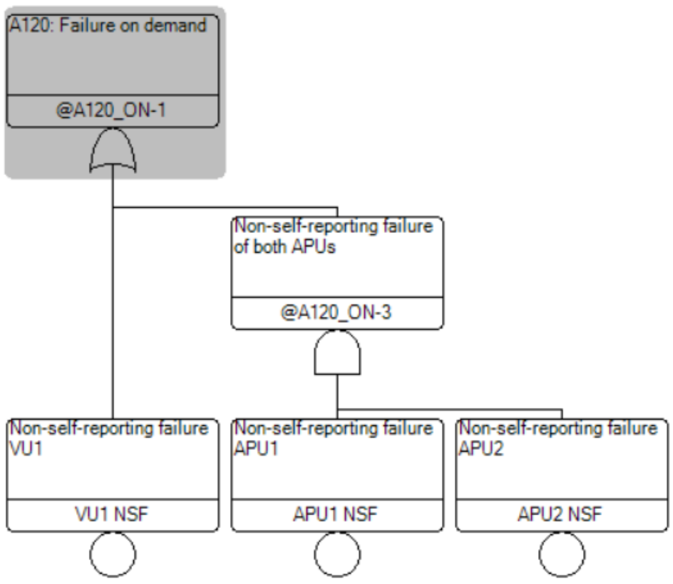

A120: Failure on demand
@A120_ON-1
Non-self-reporting failure of both APUs
@A120_ON-3
Non-self-reporting failure VU1
VU1 NSF
Non-self-reporting failure APU1
APU1 NSF
Non-self-reporting failure APU2
APU2 NSF

Figure 3.4 Fault tree for A120 (without network failures)

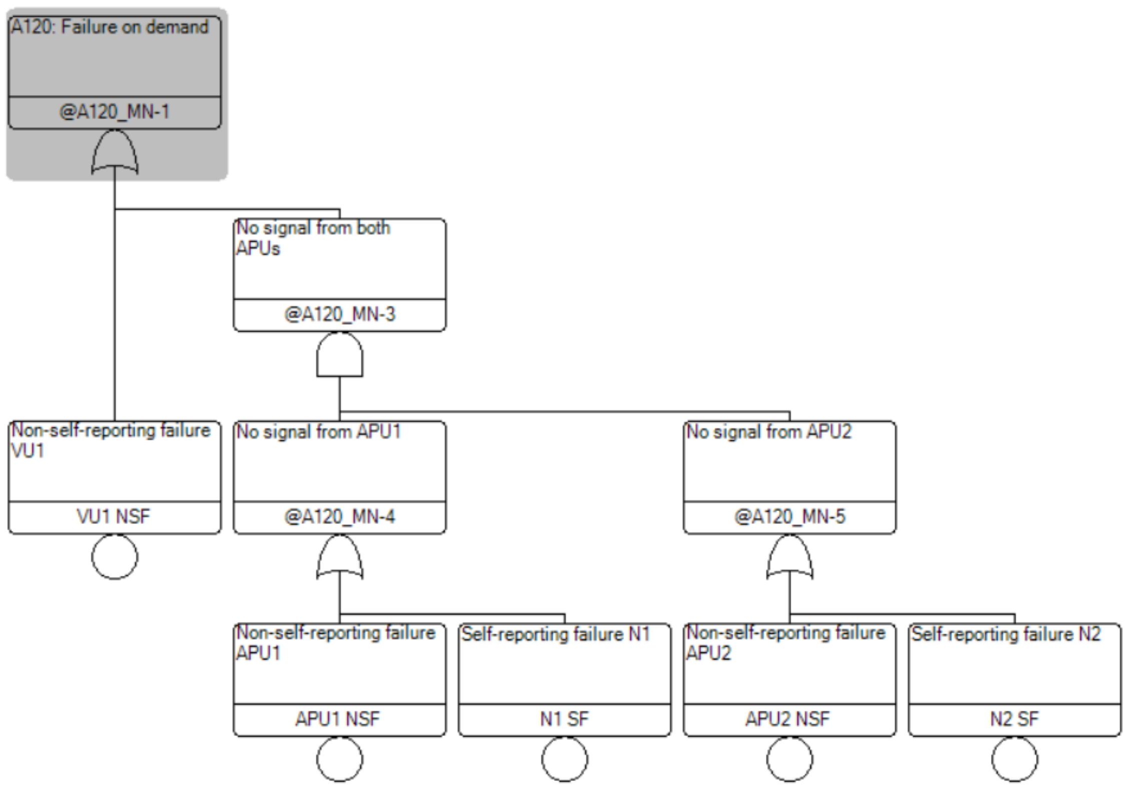


Figure 3.5 Fault tree for A120 (with network failures N1, N2)

The minimal cut sets[18] (MCS) for the two fault trees for the model system A120 (with and without consideration of network failures) are shown in Table 3.4 and Table 3.5.

Table 3.4 Minimal cut sets for the model system A120 (without network failures)

No	Probability	%	Event 1	Event 2
1	1,73E-04	99,98	VU1 NSF	
2	3,01E-08	0,02	APU1 NSF	APU2 NSF

[18] Minimal cut sets (MCS) are combinations of fault causes in which each individual cause is necessary to cause the undesired event. If even one of these causes is removed, the top event can no longer occur.

Table 3.5 Minimal cut sets for the model system A120 (with network failures)

No	Probability	%	Event 1	Event 2
1	1,73E-04	99,98	VU1 NSF	
2	3,01E-08	0,02	APU1 NSF	APU2 NSF
3	2,77E-08	0,02	N1 SF	APU2 NSF
4	2,77E-08	0,02	APU1 NSF	N2 SF
5	2,56E-08	0,01	N1 SF	N2 SF

The failure combinations determined using the fault trees, which lead to a total failure of the system (minimal cut sets - MCS, Table 3.4 and Table 3.5), confirm both the results obtained manually and those obtained using the automatic failure effects analyses (compare to Table 3.2 and Table 3.3).[19]

The representative comparison of the quantitative results of the fault tree analysis with the Monte Carlo simulation for the model system A120 (with network faults) in Figure 3.6 shows that both methods provide matching values for the probability of a total failure ("failure on demand"). After about 100,000 simulated years (~ 900 million "repetitions" in the sense of a Monte Carlo simulation), the calculated mean probability settles well at the value obtained by a fault tree analysis.

[19] Please also refer to footnote 17.

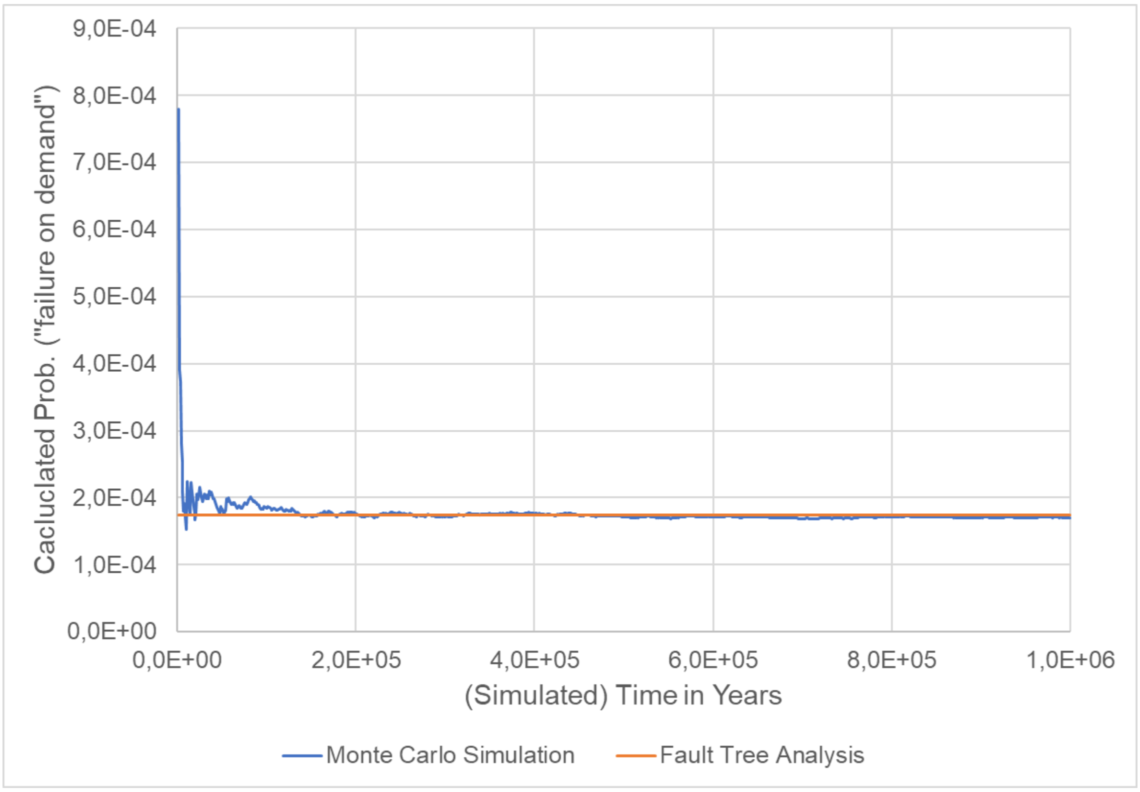


Figure 3.6 Exemplary comparison of a fault tree analysis with a Monte Carlo simulation for the model system A120

Details on these types of simulations can also be found in /MCH 21/.

3.3 Analyses of model systems

As demonstrated in the previous section for a simple model system (A120), detailed analyses were carried out for a number of representative, increasingly complex model systems:

- A122
  - 1 voting unit (VU), 2 processing units (PU), 2 acquisition units (AU) of I&C platform A
- A122mod
  - 1 voting unit (VU), 2 processing units (PU), 2 acquisition units (AU) of I&C platform A

  - Obtained from A122 by changing a single function block. Details on this can be found in /MCH 21/.

- A222
  - 2 voting units (VU), 2 processing units (PU), 2 acquisition units (AU) of I&C platform A

- A222mod
  - 2 voting units (VU), 2 processing units (PU), 2 acquisition units (AU) of I&C platform A
  - Obtained from A222 by changing two function blocks. Details on this can be found in /MCH 21/.

- A133
  - 1 voting unit (VU), 3 processing units (PU), 3 acquisition units (AU) of I&C platform A

- A333
  - 3 voting unit (VU), 3 processing units (PU), 3 acquisition units (AU) of I&C platform A

- A133B133
  - 1 voting unit (VU), 3 processing units (PU), 3 acquisition units (AU) of I&C platform A and 1 voting unit (VU), 3 processing units (PU), 3 acquisition units (AU) of I&C platform B
  - The I&C platforms A and B are diverse to each other

- A333B333
  - 3 voting units (VU), 3 processing units (PU), 3 acquisition units (AU) of I&C platform A and 3 voting units (VU), 3 processing units (PU), 3 acquisition units (AU) of I&C platform B
  - The I&C platforms A and B are diverse to each other

More detailed descriptions of the model systems can be found in appendix A.3. For the following considerations, however, it is only important that the listed model systems tend to become more complex from top to bottom (and also have more redundancies) and that the last two model systems also have diversities (subsystems A and B).

Parameters based on /MCH 18/ and /MCH 21/ were used as reliability parameters (see Table 3.6).

Table 3.6 Parameters used for self-reporting (SF) and non-self-reporting failures (NSF) as well as CCFs of the modules of the model systems (see also /MCH 21/)

Parameter	**FR according to /MCH 21/**	**Remarks**
FR AL NSF	1E-10 h-1	Analog Voter
FR AU SF	2.10E-05 h-1	
FR AU NSF	8.26E-08 h-1	
FR PU SF	1.57E-05 h-1	
FR PU NSF	8.26E-08 h-1	
FR VU SF	6.97E-06 h-1	
FR VU NSF	8.26E-08 h-1	
FR NeCom SF	1.00E-04 h-1	No CCFs20
FR AU CCF	4.35E-09 h-1	CCF of all AU in one subsystem
FR PU CCF	4.35E-09 h-1	CCF of all PU in one subsystem
FR VU CCF	4.35E-09 h-1	CCF of all VU in one subsystem

Since explicit failures in the network connections were now also considered, a value had to be assumed for their failure rate. As a first approximation, a value of (arbitrary, but comparatively large) 1.0E^{-4} h^{-1} was defined. However, as this assumption cannot be justified beyond doubt, sensitivity analyses were carried out on this parameter instead.

If the failure rate for network failures is varied with otherwise constant parameters, the relationship shown in Figure 3.7 for the model system A122mod, for example, results. In

[20] As all network failures are basically self-reporting, CCFs do not need to be explicitly considered here. Every failure that occurs is detected immediately and rectified within the repair time. Typically, however, CCFs are only relevant when undetected (i.e., for NSF).

addition, this figure also shows how well the results obtained with Monte Carlo simulations agree with the corresponding fault tree analyses.

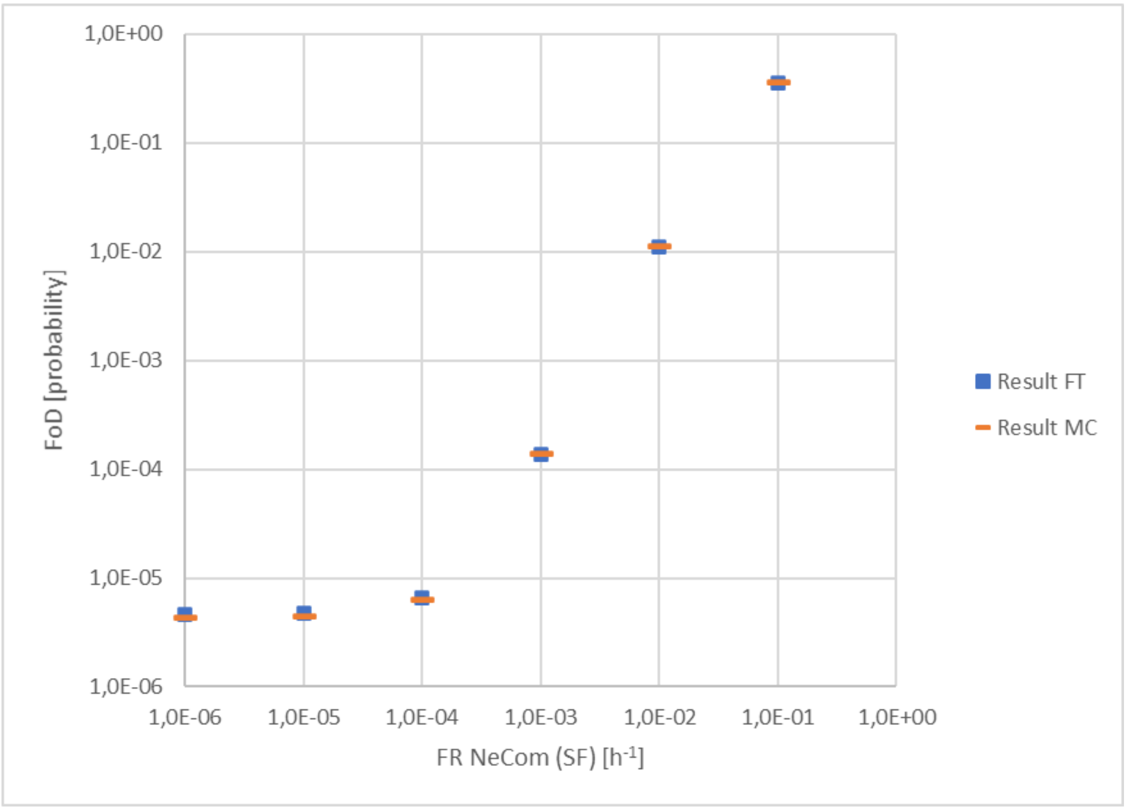


Figure 3.7 Sensitivity analysis: Influence of the failure rate in network communication (FR NeCom) on "failures on demand" (FoD) for A222mod

Fault tree analyses (FT) and Monte Carlo simulations (MC)

Corresponding analyses were carried out for all model systems. The results of the sensitivity analyses with regard to the failure rate of network communication failures for all model systems are shown together in Figure 3.8. In this illustration, the higher reliability of systems with more redundancies and, in particular, with diversity is immediately apparent. For example, the model system A333B333 is more than six orders of magnitude more reliable than the model system A222.

On the other hand, the diagram in Figure 3.8 also gives an impression of how network communication failures with different assumed failure rates affect the model systems.

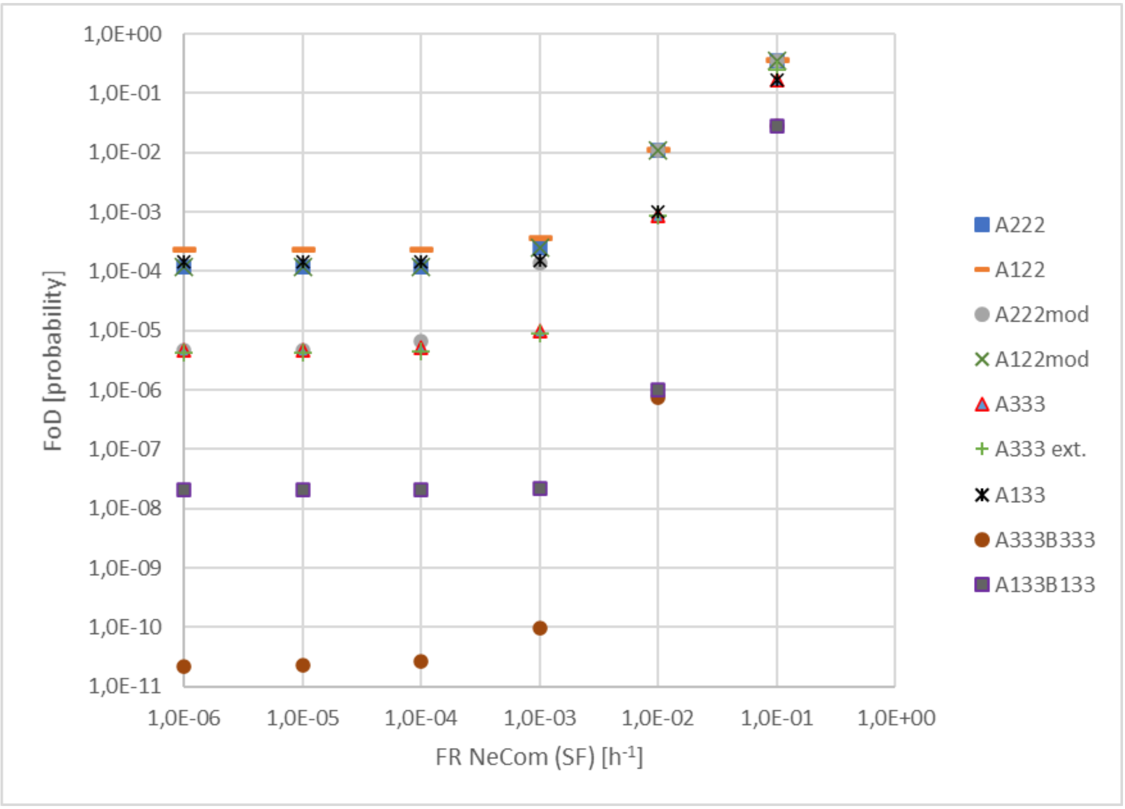


Figure 3.8 Sensitivity analyses: Influence of the failure rate in network communication (FR NeCom) on "failures on demand" (FoD) for all model systems (I)

The corresponding correlations can be visualized even more clearly by normalizing the results. If the probability of a failure on demand for all model systems and a network communication failure rate FR NeCom = $1 \cdot 10^{-6}$ h^{-1} is normalized to 1, the diagram shown in Figure 3.9 results.

Figure 3.9 shows impressively that failures in network communication only have a significant effect at failure rates above $1 \cdot 10^{-4}$ h^{-1}. Below this value, virtually no influence can be observed. Above this value, the model systems behave similarly, although the **relative** influence tends to be somewhat higher for more reliable model systems (due to the larger number of network connections within these systems).

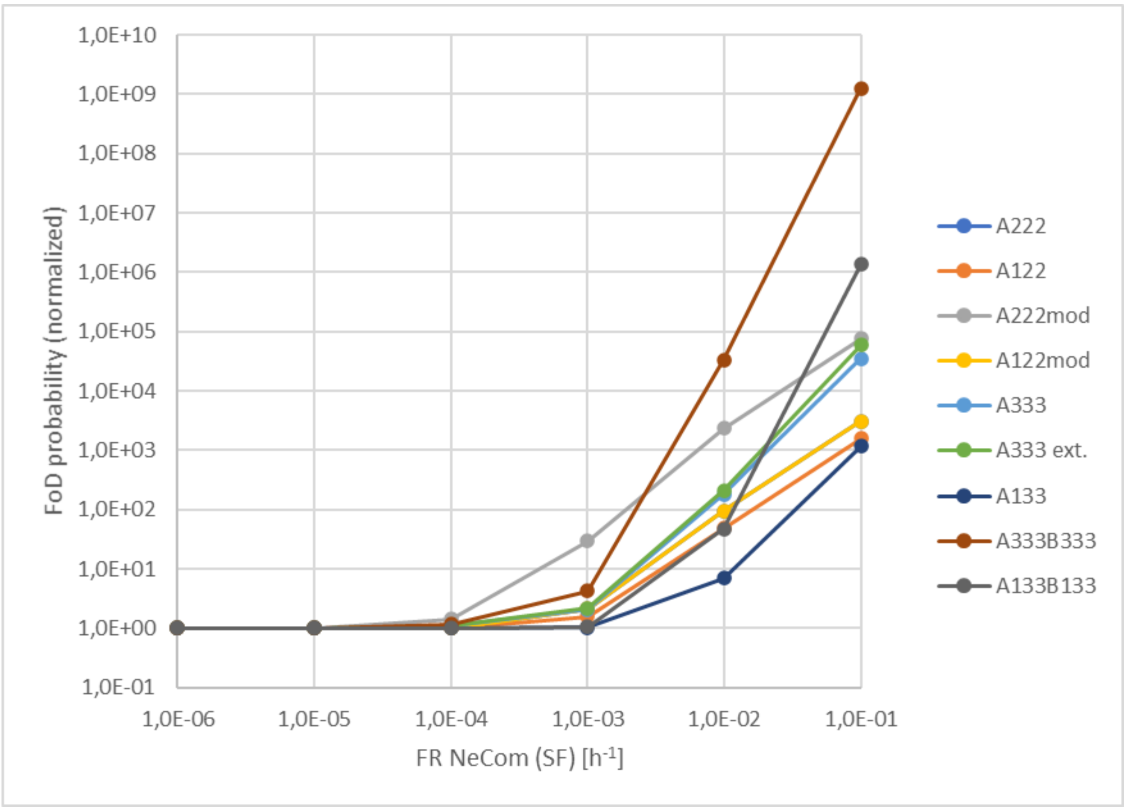


Figure 3.9 Sensitivity analyses: Influence of the failure rate in network communication (FR NeCom) on "failures on demand" (FoD) for all model systems (II)

As Figure 3.8, except that the probabilities for a failure on demand were normalized to 1 (for FR NeCom = $1.0 \cdot 10^{-6}$ h^{-1})

In order to have a significant influence on the reliability of the model systems, comparatively high failure rates must therefore be assumed for the network communication. These are then more than an order of magnitude higher than the failure rates of other self-reporting failures (see Table 3.6). With even higher assumed failure rates, the failures in network communication eventually become dominant (see Figure 3.8), but no corresponding observations were made in real digital I&C systems. Overall, it can therefore be assumed that the originally assumed failure rate is already more than sufficiently conservative.

Overall, it can also be concluded that failures in network communication have only an extremely small influence on the overall reliability of I&C systems.

4 Summary and overall results

The aim of the project was to investigate failures in the network communication of digital I&C systems in order to be able to explicitly and therefore more accurately take these into account in the methods used by GRS in the future.

To this end, various tests were initially carried out using a test system, a real I&C system based on components and software from the Teleperm XS platform of Framatome. This made it possible to understand the network communication within the test system so well that so-called network manipulators could be developed as part of the project. These not only allow the entire network traffic of any network connection to be read ("sniffed") but can also be used to manipulate the network traffic in a variety of ways in the sense of fault injection.

It was shown that the representative test system is extremely robust against randomly occurring failures in the network connections. It can be assumed that failures do not occur undetected in networks, as all failures are basically self-reporting.[21] This means that only one relevant type of failure in network connections needs to be assumed when modeling systems. Corresponding failures were therefore integrated into existing and newly created model systems as additional failure types (in addition to the failures already considered in the past, e.g., of hardware components).

Subsequently, failures that can occur in network communication, as well as their propagation and effects, were examined in more detail in a series of increasingly complex model systems. In addition to fault tree analyses and Monte Carlo simulations, specific implementations of individual model systems with real I&C components were also used for this purpose.

As the specific probability of failures occurring in network communication is not known, corresponding sensitivity analyses were carried out instead. It was found that a significant influence on the reliability of all model systems can only be observed for quite high

[21] However, this does not rule out the possibility that failures may still remain undetected, e.g., due to faulty planning. Although every (network) failure is in principle self-reporting, it must also be processed or registered accordingly. An analogy in the world of hard-wired I&C systems would be if a fault could always be detected in principle, but no message (e.g., in a message slot) was available in the control room. Such faults were not additionally considered in this project.

assumed failure rates[22]. Overall, the (additional) effects of explicitly considering network failures can therefore be classified as low, which is mainly due to the fact that these are basically self-reporting.[23] Nevertheless, they can and will become an integral part of the GRS methodology after completion of this project.

In addition, although this was not an original goal, this project was able to show how the network manipulators developed can be used to carry out (or simulate) cyberattacks on I&C systems. Although physical access to the network connection to be attacked is necessary for this, further investigations in this direction appear worthwhile in subsequent projects.

A further future extension of the analyses carried out in this project could be more hardware-related investigations. For example, the network messages were examined both when using the Wireshark software and in the network manipulators at the Ethernet level. This is the lowest level that can be reached directly within computers (i.e., also the network manipulators). Below this is only the physical layer (i.e., concrete voltages/currents in the digital network). Even deeper analyses at this level would be conceivable, e.g., by using logic analyzers.

[22] Significant effects can only be observed if values are assumed for the failure rates in network communication that are at least one order of magnitude higher than the previously assumed highest failure rates of other components.

[23] Other self-reporting failures also tend to have a lower impact on the overall system than non-self-reporting failures (see e.g., /MCH 21/).

References

/BMU 15/	Bundesministerium für Umwelt, Naturschutz, Bau und Reaktorsicherheit: *Änderungen und Neufassung der Bekanntmachung zu den „Sicherheits-anforderungen an Kernkraftwerke“*, BAnz AT 30.03.2015 B2, March 2015
/BMU 15a/	Bundesministerium für Umwelt, Naturschutz, Bau und Reaktorsicherheit: *Bekanntmachung der Interpretationen zu den Sicherheitsanforderungen an Kernkraftwerke*, BAnz AT 30.03.2015 B3, March 2015
/CRC 23/	https://crccalc.com/, last accessed on 08/30/2023
/FRA 23/	Discussions with Framatome employees during their visit to GRS on 03/22/2023
/GEE 22/	Geeks for Geeks: *OSI Model Full Form in Computer Networking*, https://www.geeksforgeeks.org/osi-model-full-form-in-computer-network-ing/, last accessed on 04/17/2023
/GRS 23/	BMUV project 4722R01215 underway at GRS („*AnTeS-PRIO*“)
/ISO 94/	ISO/IEC 7498-1:1994: *Information technology - Open Systems Intercon-nection - Basic Reference Model: The Basic Model*, see e.g., https://www.iso.org/standard/20269.html, last accessed on 04/17/2023
/MAT 23/	https://de.mathworks.com/products/simulink.html, last accessed on 08/31/2023
/MCH 18/	C. Müller, J. Peschke, E. Piljugin: *Entwicklung und Erprobung eines Werkzeugs zur Sensitivitätsanalyse der Fehlerauswirkungen in der si-cherheitsrelevanten digitalen Leittechnik*, GRS-494, March 2018
/MCH 21/	C. Müller, E. Piljugin, P. Gebhardt, J. Shvab: *AnTeS – Entwicklung und Anwendung des Analyse- und Testsystem der GRS*, GRS-648, March 2021

/TXS 12/	Areva (remark: today Framatome): *Teleperm XS User Manual, TXS Core-SW Release 3.6.5*, PTLSD-G/2012/en/0217 A (Restricted), 2012
/WIK 23/	https://de.wikipedia.org/wiki/OSI-Modell, last accessed on 04/17/2023
/WIK 23a/	https://de.wikipedia.org/wiki/Single_Event_Upset, last accessed on 08/30/2023
/WIK 23b/	https://de.wikipedia.org/wiki/Latch-up-Effekt, last accessed on 08/30/2023
/WIS 23/	https://www.wireshark.org/, last accessed on 08/17/2023

List of figures

Figure 1.1 AnTeS overview ... 4

Figure 1.2 The AnTeS-SIC-real module, a real safety I&C system based on Teleperm XS ... 7

Figure 2.1 Schematic representation of data packets in the Ethernet communication of the test system (TXS, Generation 2) ... 10

Figure 2.2 Example of a TXS data packet (within the test system) ... 10

Figure 2.3 Example of messages sent from one redundancy (of AnTeS-SIC-real) ... 12

Figure 2.4 Developed software for reading ("sniffing") and influencing the network communication of the test system ... 13

Figure 2.5 Devices for manipulating network communication (for error injection) developed as part of this project ... 14

Figure 3.1 Model system A120 ... 20

Figure 3.2 A120 realized with AnTeS-SIC-real (TXS), network plan ... 24

Figure 3.3 Matlab/Simulink model of A120 ... 25

Figure 3.4 Fault tree for A120 (without network failures) ... 25

Figure 3.5 Fault tree for A120 (with network failures N1, N2) ... 26

Figure 3.6 Exemplary comparison of a fault tree analysis with a Monte Carlo simulation for the model system A120 ... 28

Figure 3.7 Sensitivity analysis: Influence of the failure rate in network communication (FR NeCom) on "failures on demand" (FoD) for A222mod ... 31

Figure 3.8 Sensitivity analyses: Influence of the failure rate in network communication (FR NeCom) on "failures on demand" (FoD) for all model systems (I) ... 32

Figure 3.9 Sensitivity analyses: Influence of the failure rate in network communication (FR NeCom) on "failures on demand" (FoD) for all model systems (II) ... 33

List of tables

Table 3.1 Parameters used for model system A120 21

Table 3.2 Failure effects analysis for A120 (without network failures)...................... 22

Table 3.3 Failure effects analysis for A120 (with network failures)........................... 23

Table 3.4 Minimal cut sets for the model system A120 (without network failures).. 26

Table 3.5 Minimal cut sets for the model system A120 (with network failures)......... 27

Table 3.6 Parameters used for self-reporting (SF) and non-self-reporting failures (NSF) as well as CCFs of the modules of the model systems (see also /MCH 21/).. 30

Abbreviations

AnTeS	Analysis and Test System of GRS for (digital) I&C
AnTeS-FIELD	AnTeS module FIELD (field systems)
AnTeS-OIC	AnTeS module OIC (operational I&C)
AnTeS-PRIO	AnTeS module PRIO (priority modules or priority and actuation control modules)
AnTeS-SIC	AnTeS module SC (safety I&C)
APU	Acquisition and Processing Unit (combined AU and PU) within an I&C system
AU	Acquisition Unit within an I&C system
AV42	Concrete Priority Actuation and Control module (digital) of TXS
BMU	Bundesministerium für Umwelt, Naturschutz und nukleare Sicherheit - Federal Ministry for the Environment, Nature Conservation and Nuclear Safety (until 2021)
BMUV	Bundesministerium für Umwelt, Naturschutz, nukleare Sicherheit und Verbraucherschutz - Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (since 2021)
CCF	Common Cause Failure
(S)CP3	Communication Processor of TXS
CRC	Cyclic Redundancy Check – Method for Determining a Checksum, also used for the Designation of the Checksum
DLL	Dynamic Link Library
FMEA	Failure Mode and Effects Analysis
FTA	Fault Tree Analysis
GRS	Gesellschaft für Anlagen- und Reaktorsicherheit gGmbH
GSM	Graphical Service Monitor (Software) of TXS
I&C	Instrumentation and Control (System)
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
MitM	Man-in-the-Middle (Cyberattack)

MTTR	Mean Time to Repair (in FTA)
NSF	Non-Self-Reporting Failure
OIC	Operational I&C
OSI	Open Systems Interconnection
PU	Processing Unit within an I&C System
SEL	Single Event Latch-up
SEU	Single Event Upset
SF	Self-Reporting Failure
SIC	Safety I&C
SimGen	Simulation Generator, Software developed by GRS as part of Project 4718R01314 for the flexible Simulation of Process Engineering Systems
SPACE	Specification And Coding Environment (Engineering Environment of TXS)
SPLM1	Concrete Priority Actuation and Control module (hard-wired) of TXS
TF	Time to First Test (used within FTA)
TI	Test Interval (of Recurring Tests) (used within FTA)
TXS	I&C platform Teleperm XS from Framatome
USB	Universal Serial Bus
(S)VE2	Processing Unit (German: “Verarbeitungseinheit”) of TXS
VU	Voting Unit within an I&C System

A Detailed and additional descriptions

A.1 TXS2Simulink

Model systems often have to be considered equally with both AnTeS-SIC-real (real I&C system TXS) and AnTeS-SIC-sim (simulated I&C system) as part of investigations and analyses. In these cases, the I&C functions normally have to be created manually in the form of function diagrams for both the real and the simulated system. As part of the /MCH 21/ project, however, a proof-of-concept demonstrated that it is also possible in principle to automatically translate function diagrams of the real I&C system (TXS) into Matlab/Simulink files (simulated I&C system).

During the engineering process ("programming") of TXS, I&C functions are created graphically as function diagrams with the help of SPACE ("Specification and Coding Environment"). Before these can be uploaded to the actual I&C system, a two-stage process[24] takes place in which the functions created are first translated into C-files[25] and then compiled. These automatically generated C-files have good automatically generated comments, so that it was comparatively easy to analyze them.

On this basis, the TXS2Simulink software (in the Python programming language) was created as part of this project, which automatically translates TXS function diagrams (described by C-files) into Simulink models. This software is briefly presented below.

[24] In the version of TXS available to GRS, both steps (translation into C files and compilation) are initiated individually by the programmer. In newer versions, both steps can be accessed together by a single program call, but the basic procedure in TXS has not changed.

[25] Program code in the C programming language.

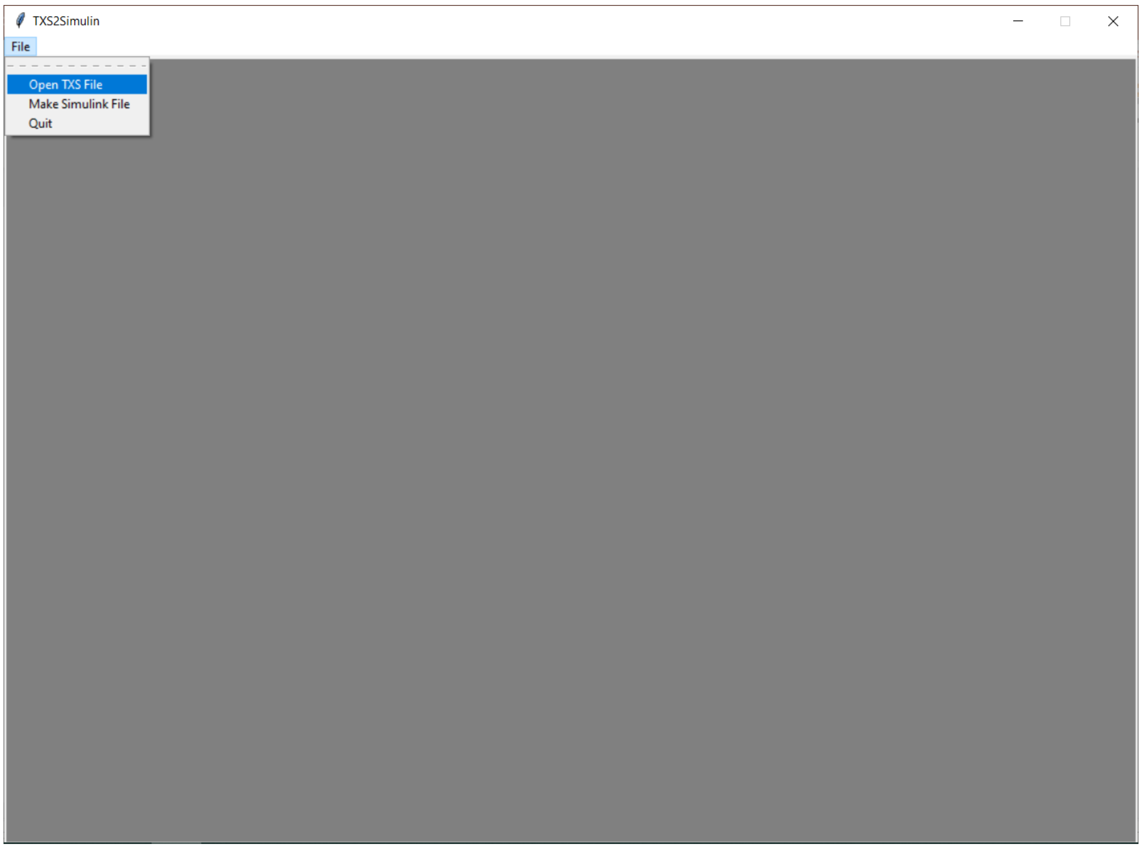


Figure A 1 Start window of TXS2Simulink, screenshot

Figure A 1 shows a screenshot of TXS2Simulink immediately after starting the software. If you open the C-file of a TXS function diagram via the menu ("File" → "Open TXS File"), it is displayed graphically in the window (Figure A 2). By subsequently selecting "Make Simulink File" in the menu, the opened TXS function diagram can then be automatically translated into a Matlab/Simulink model. Additional information on the converted TXS function diagram is displayed on the computer's console (Figure A 3).

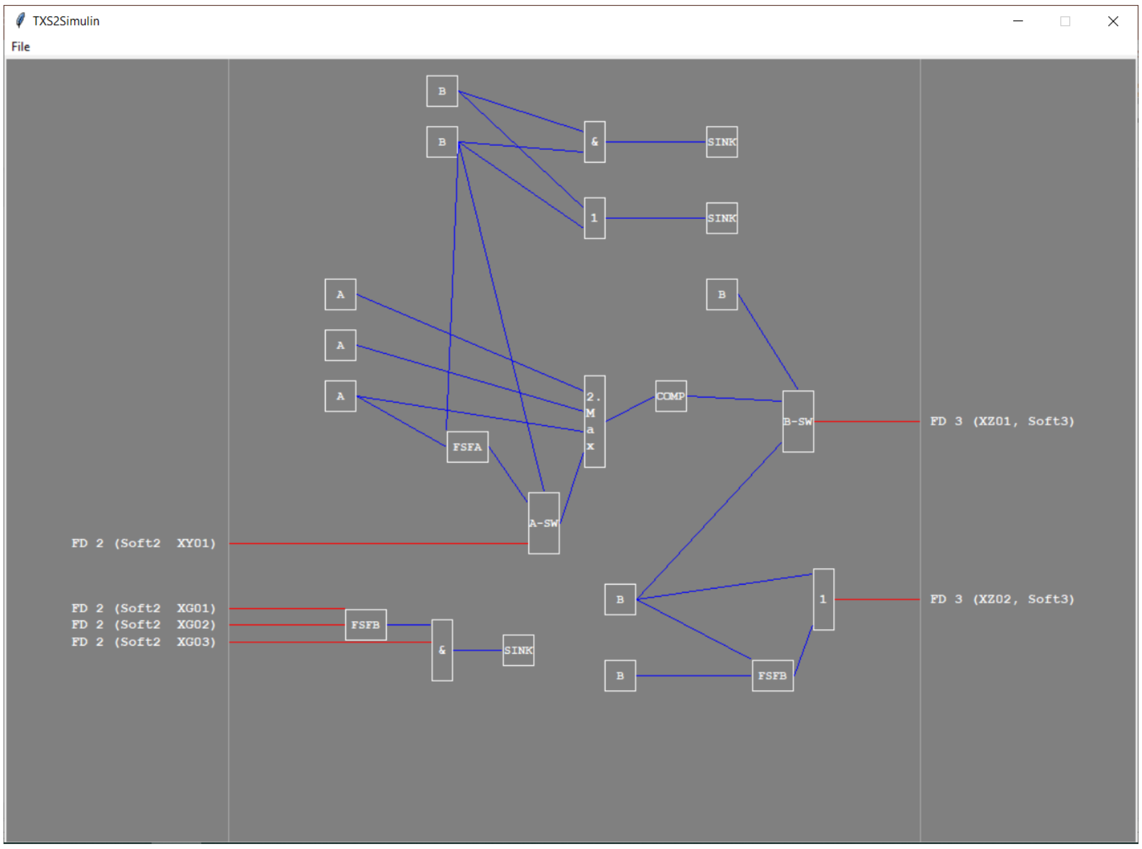
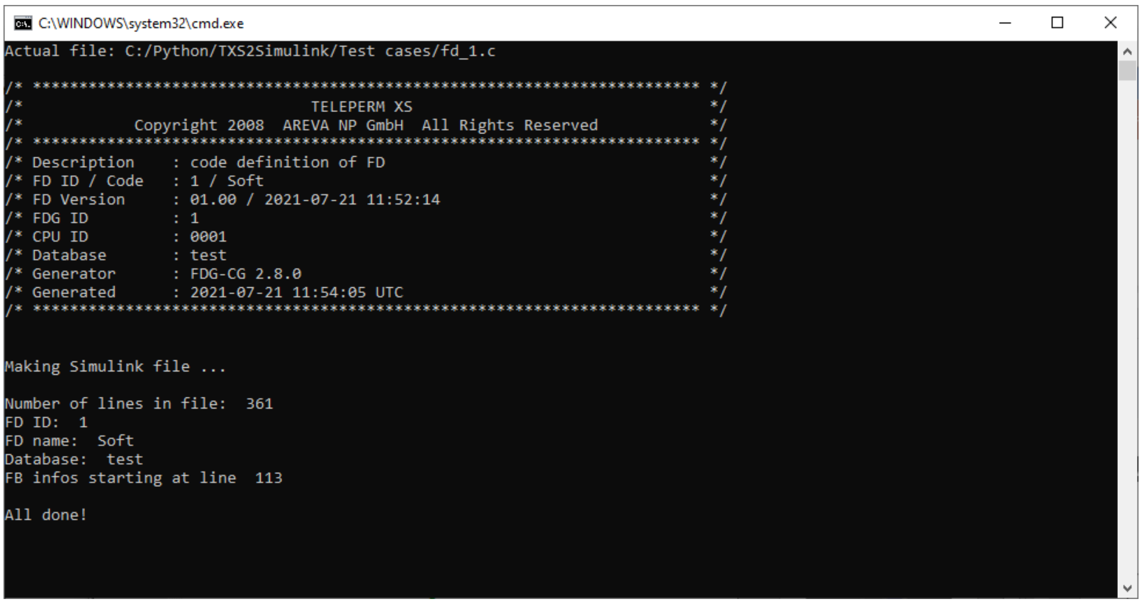


Figure A 2 Representation of a TXS function diagram in TXS2Simulink



```
C:\WINDOWS\system32\cmd.exe
Actual file: C:/Python/TXS2Simulink/Test cases/fd_1.c

/* ************************************************************************ */
/*                              TELEPERM XS                                 */
/*            Copyright 2008  AREVA NP GmbH  All Rights Reserved            */
/* ************************************************************************ */
/* Description      : code definition of FD                                 */
/* FD ID / Code     : 1 / Soft                                              */
/* FD Version       : 01.00 / 2021-07-21 11:52:14                           */
/* FDG ID           : 1                                                     */
/* CPU ID           : 0001                                                  */
/* Database         : test                                                  */
/* Generator        : FDG-CG 2.8.0                                          */
/* Generated        : 2021-07-21 11:54:05 UTC                               */
/* ************************************************************************ */


Making Simulink file ...

Number of lines in file:  361
FD ID:  1
FD name:  Soft
Database:  test
FB infos starting at line  113

All done!
```

Figure A 3 Console output of TXS2Simulink when converting a TXS function diagram into a Matlab/Simulink model

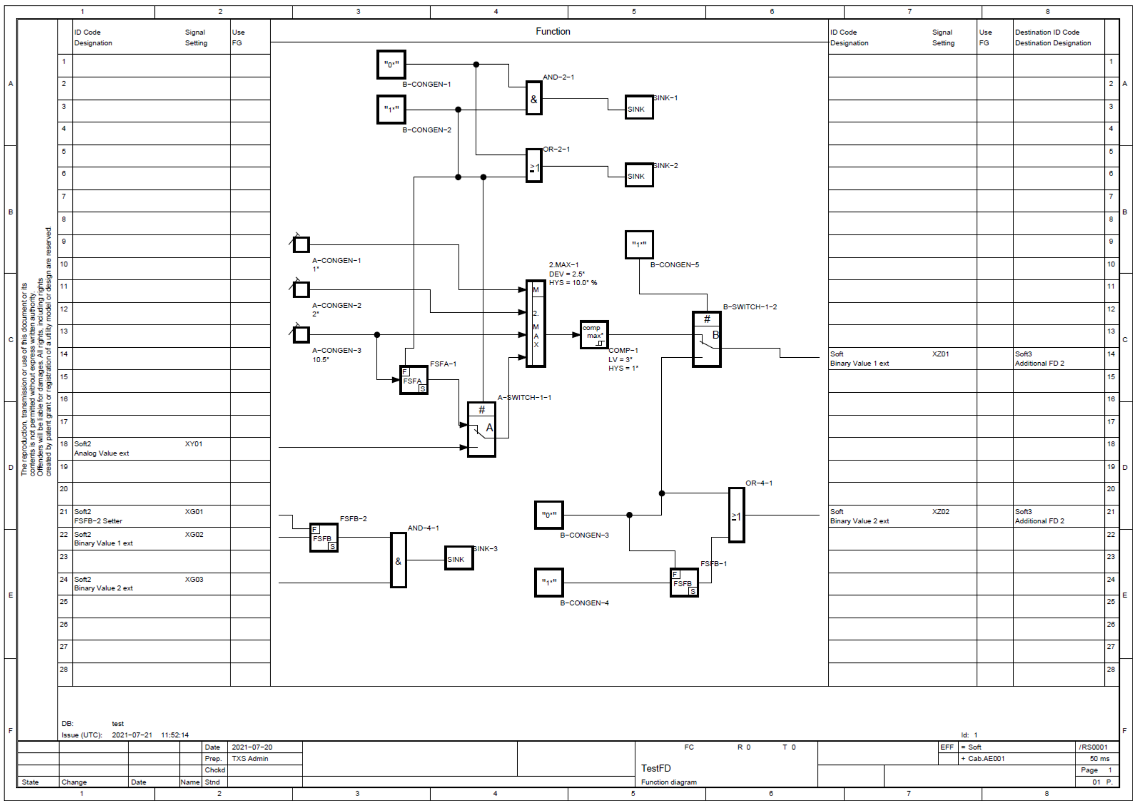


Figure A 4 TXS sample function diagram

If, for example, the TXS function diagram shown in Figure A 4 is converted into a Matlab/Simulink model in this way, the result is the (Matlab/Simulink) function diagram for the simulation shown in Figure A 5. The function blocks contained in this function diagram (FB_1, FB_2, ...) themselves contain further logic that simulates the behavior of the corresponding TXS function blocks (see also /MCH 21/).

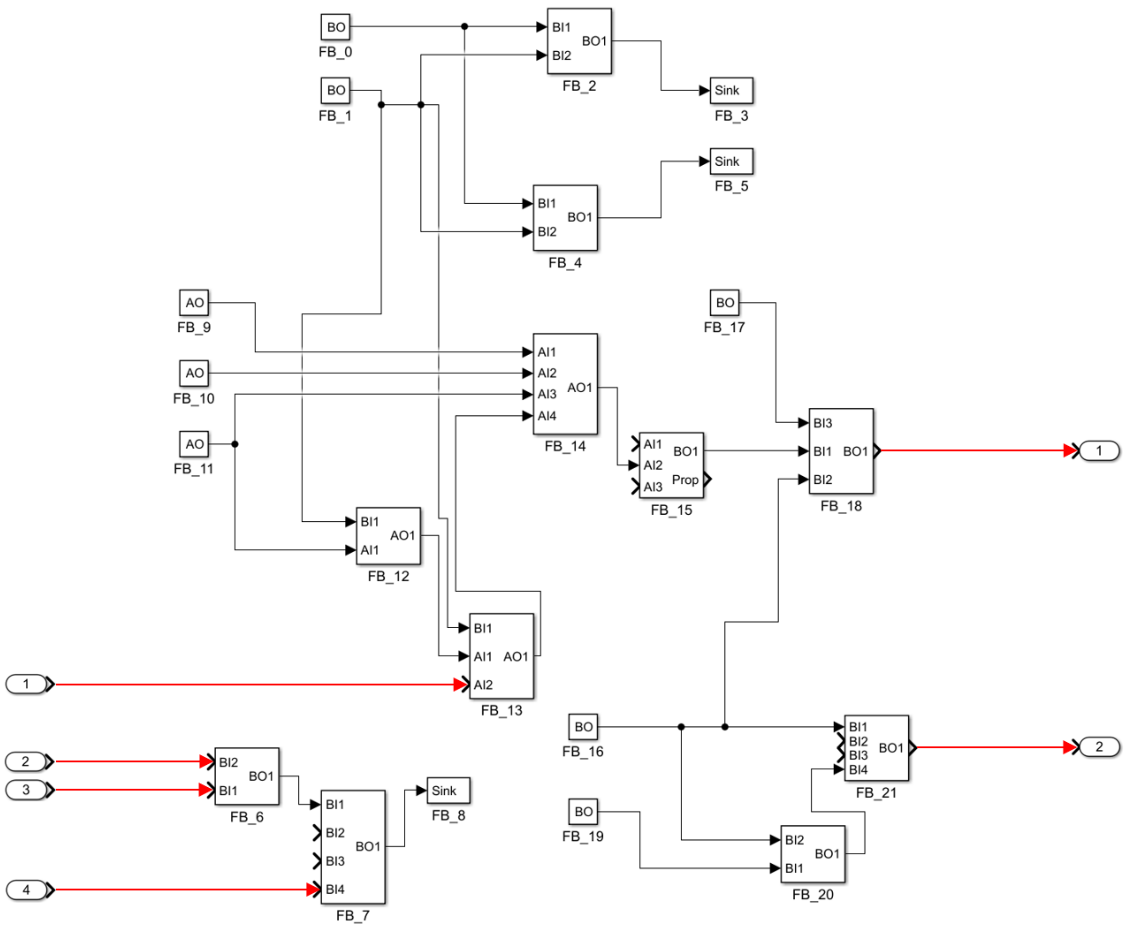


Figure A 5 Simulink model generated by TXS2Simulink for the example function diagram in Fig. A 4, the arrows marked in red are input or output signals

However, it should be noted that although each TXS function diagram also has its own C-file, this does not constitute fully-fledged software on its own. When compiling the C-files, not only all function diagrams, but also many other C-files (e.g., for hardware configuration, etc.) are compiled together to form executable software. One of the consequences of this is that when connecting the input and output signals (red arrows in Figure A 5) to function blocks in the respective C-file, it is not clear to which port of the function block they are connected. Often the assignment is nevertheless clear if, for example, only a single port comes into question (for example, only one output port is available to output signal "2" in Figure A 5 on function block FB_21), but even when using TXS2Simulink, manual checking and, if necessary, post-processing is still necessary.

A.2 Analyses with the network manipulators

Based on the findings of the experiments with Wireshark, software was developed that allows data packets to be read and, in the sense of fault injection, modified (Figure A 7).[26] This software runs on network manipulators specially designed for this purpose (based on Raspberry Pi 4 microcomputers, Figure A 6). A more detailed description of the software can be found in section A.2.2.

Each network manipulator has three Ethernet connections. One is used exclusively to control the network manipulators from the outside, while the other two are used to forward all network traffic in both directions, either unaffected or manipulated.

Figure A 8 shows a typical system setup when using a network manipulator. In this example setup, the entire data traffic from and to redundancy 1 (“Red. 1”) can be read and, if necessary, changed.

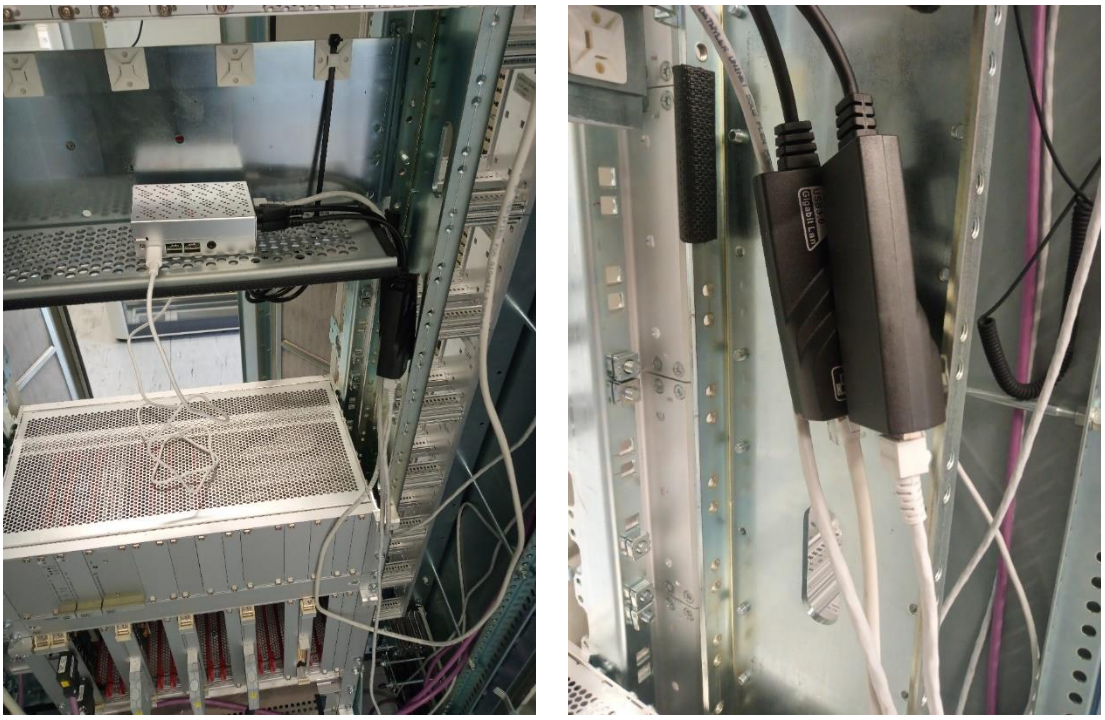
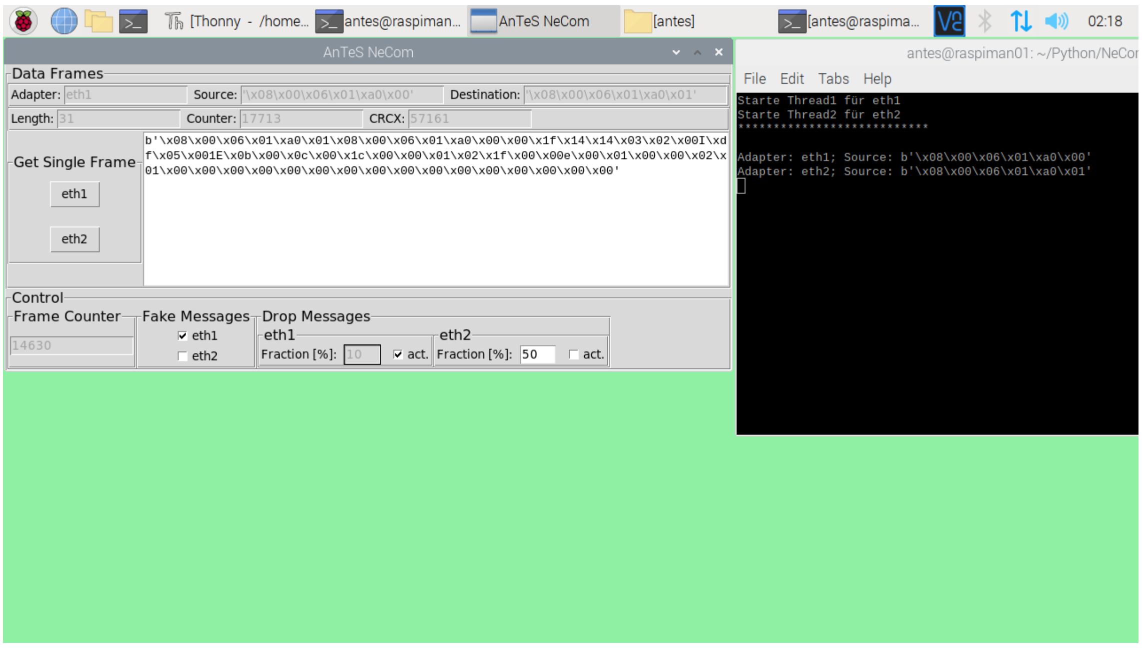


Figure A 6 A network manipulator in action

[26] In addition, the software can also carry out targeted manipulation of the data packets in the sense of cyberattacks.



AnTeS NeCom

Data Frames

Adapter: eth1 Source: '\x08\x00\x06\x01\xa0\x00' Destination: '\x08\x00\x06\x01\xa0\x01'

Length: 31 Counter: 17713 CRCX: 57161

Get Single Frame: eth1 | eth2

```
b'\x08\x00\x06\x01\xa0\x01\x08\x00\x06\x01\xa0\x00\x00\x1f\x14\x14\x03\x02\x00I\xd
f\x05\x001E\x0b\x00\x0c\x00\x1c\x00\x00\x01\x02\x1f\x00\x00e\x00\x01\x00\x00\x02\x
01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
```

Control

Frame Counter: 14630

Fake Messages: [✓ eth1] eth2

Drop Messages: eth1 Fraction [%]: 10 [✓ act.] eth2 Fraction [%]: 50 act.

antes@raspiman01: ~/Python/NeCom

File Edit Tabs Help

```
Starte Thread1 für eth1
Starte Thread2 für eth2
******************************

Adapter: eth1; Source: b'\x08\x00\x06\x01\xa0\x00'
Adapter: eth2; Source: b'\x08\x00\x06\x01\xa0\x01'
```

Figure A 7 Software of the network manipulators, screenshot

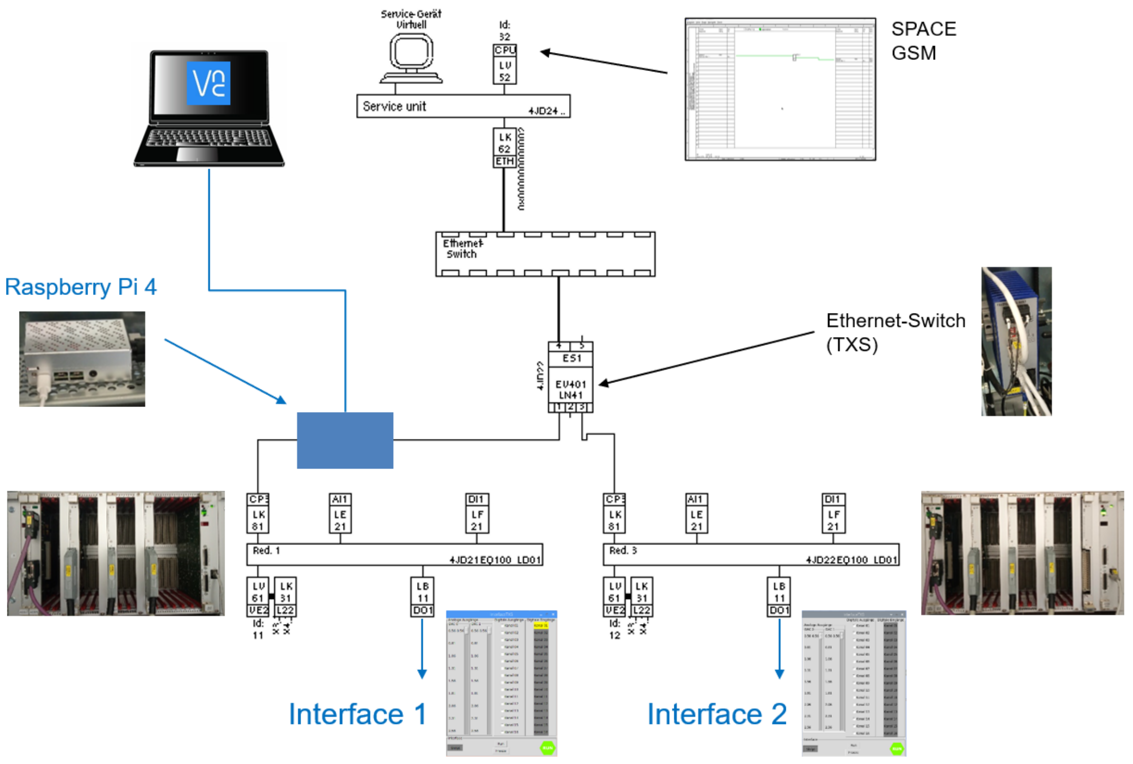


Figure A 8 Example of the integration and use of a network manipulator ("Raspberry Pi 4" in the picture)

On the one hand, the network manipulators were used to determine the basic behavior of network communication in the event of failures (i.e., their relevant failure types). On the other hand, the behavior of certain model systems (by fault injection) could also be investigated with real systems (AnTeS-SILT-real) (see e.g., section 3.2).

More complex model systems with more than five network connections, on the other hand, were investigated exclusively on the basis of the previously determined relevant failure modes using simulated systems (see also, for example, Section 3.2).

Since the analyses of the (overall) model systems are described in detail in Sections 3.2 and 3.3, only a more detailed description of the determination of the relevant failure modes of network communication follows here.

A.2.1 Determination of the relevant failure modes

With the help of the knowledge gained so far and, in particular, the network manipulators developed on this basis, the relevant failure modes of network communication were determined. The test system used for this corresponds to the example shown in Figure A 8.

Initial tests concentrated on the fundamental effects of changing individual bits of a data packet on how and whether such a message is still processed by the recipient. To this end, systematic changes were made to bits in each individual position of a data packet.

Such changes of a single bit correspond either to so-called single event upsets (SEU) /WIK 23a/, as they can occur, for example, in semiconductor components during the passage of high-energy ionizing particles (e.g., heavy ions, protons), or so-called single event latch-ups (SEL) /WIK 23b/, as they can occur, for example, due to local short circuits, whereby the state of individual bits is changed ("bit flip") either once (i.e., without permanent damage - SEU) or permanently (SEL) in a component.

It was shown that in all tests, data packets are reliably recognized as faulty on the receiver side as soon as even a single bit is changed at any point.[27] In particular, the change of a single bit only in a single data packet has practically no effect at all (see also

[27] Theoretically, in individual cases, changing a single bit could also result in data packets that are not recognized as faulty. However, their random occurrence is very unlikely (see explanations in section A.2.1).

explanations on various loss rates of data packets below)[28]. If individual bits in a series of data packets are changed, the corresponding network communication is recognized as faulty by the receiver (and marked with red crosses in the Graphical Service Monitor (GSM), for example (see Figure A 10) - these failures are therefore self-reporting or at least (easily) detectable).

Several data packets must be affected for failures in network communication with significant effects to occur at all. Therefore, in the following tests, the proportion of faulty data packets (by error injection with the network manipulators) was successively increased (0 %, 10 %, 20 %, ...) and the respective effects on the overall system were recorded and documented.

Here, network communication in the test system proved to be extremely robust. Up to around 50 % of the data packets can be faulty or completely lost without any significant effects on the receiver side (Figure A 9).

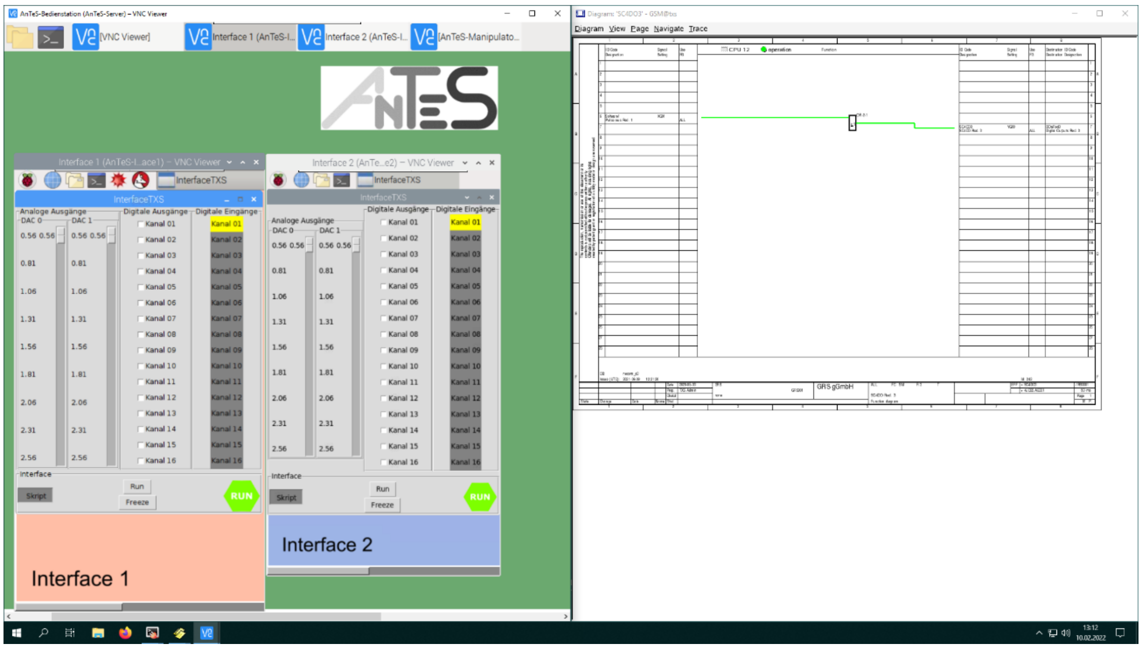


Figure A 9 Behavior of the system in Figure A 8 when less than 50 % of the data packets are affected (snapshot)

[28] However, in the case of a single "defective" data packet, actuations (of, e.g., safety functions) may be delayed by the cycle time set in the I&C system (typically ~ 50 ms). This can and should be considered when selecting the cycle time used in the I&C system.

The left side shows the outputs of the two interfaces, the right side shows the signal received by redundancy 3 (“Red. 3”) in GSM.

It did not matter whether, for example, 1/3 (~ 33 %) of the data packets were changed randomly or whether every third data packet was actually affected.

If more than 70% of the data packets are faulty or lost, the network communication on the receiver side is assessed as completely failed and the corresponding signals in the GSM are marked with red crosses (Figure A 10).[29]

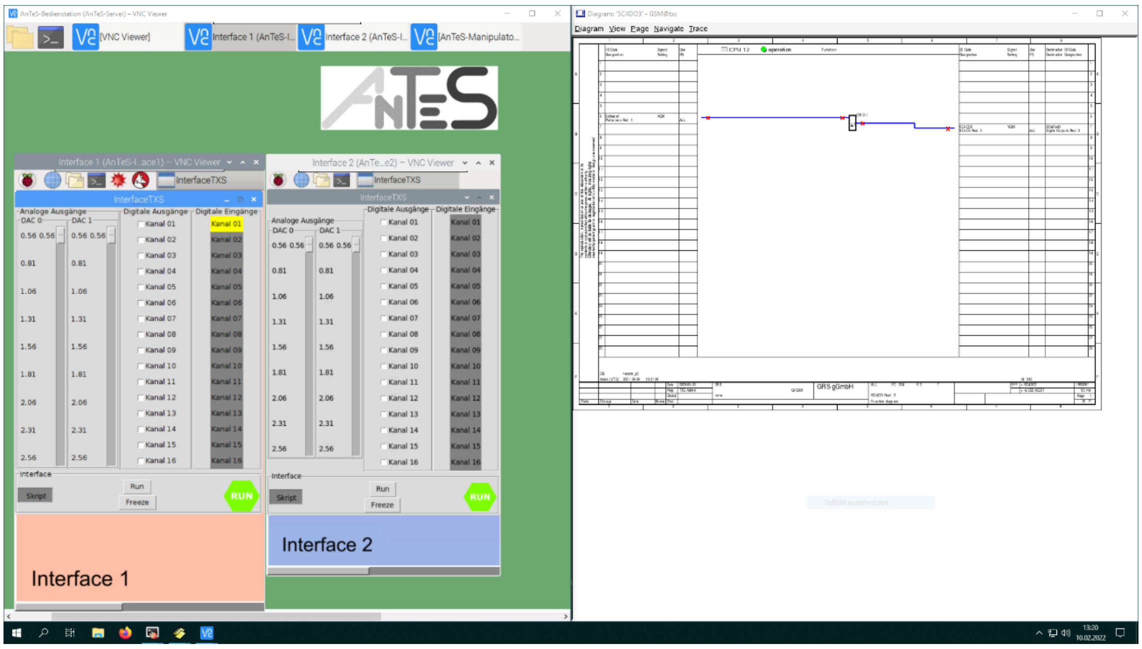


Figure A 10 Behavior of the system in Figure A 8 when more than 70 % of the data packets are affected (snapshot)

No signal output can be seen in interface 2 (of the receiver). In GSM, communication is marked as faulty/failed by red crosses.

Between typically 50 % and 70 % of faulty or lost data packets, a "flickering" occurs. In this range, the observed behavior alternates (several times per second) between the two states shown in Figure A 9 and Figure A 10. In this range, actuations (e.g., of safety functions) would still occur (albeit possibly with a slight delay), but the faulty behavior would also be detected at the same time.

[29] Note: Such failures can be made fail-safe by selecting suitable default values for function blocks or evaluating these errors in the software.

Note: The values given are only to be understood as approximate reference values. In repeated tests, the actual numerical values (in the percentage point range) were often slightly different. This is presumably due to slightly varying timing in each test - the two redundancies of the test system and the network manipulator used all work cyclically, but completely asynchronously to each other.

Overall, the tests resulted in the behavior summarized in Table A 1 with different proportions of modified data packets.

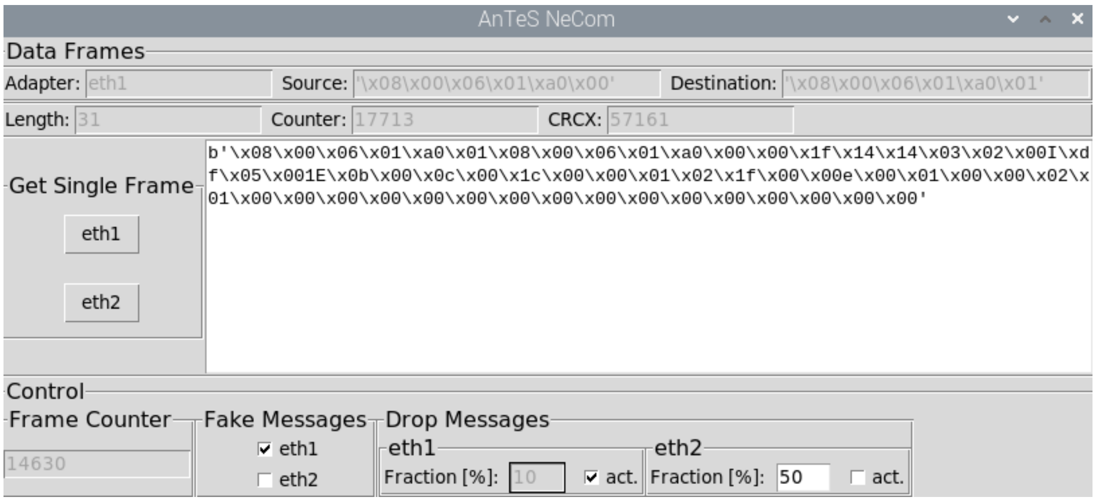
Table A 1 Influence of the proportion of disturbed data packets on the observed behavior in experiments carried out

Experiment	**Observed behavior**
Undisturbed	ok
10 % of the data packet disturbed	ok
20 % of the data packet disturbed	ok
30 % of the data packet disturbed	ok
40 % of the data packet disturbed	ok
50 % of the data packet disturbed	flickering*)
60 % of the data packet disturbed	flickering*)
70 % of the data packet disturbed	SF
80 % of the data packet disturbed	SF
90 % of the data packet disturbed	SF
100 % of the data packet disturbed	SF

*) the system switches back and forth several times per second between the two states shown in Fig. A 9 and Fig. A 10
SF – self-reporting failure

A.2.2 Network manipulation software

In this section, the software running in the network manipulators (Figure A 11) is presented in more detail.


AnTeS NeCom

Data Frames

Adapter: eth1 Source: '\x08\x00\x06\x01\xa0\x00' Destination: '\x08\x00\x06\x01\xa0\x01'

Length: 31 Counter: 17713 CRCX: 57161

Get Single Frame: eth1 eth2

```
b'\x08\x00\x06\x01\xa0\x01\x08\x00\x06\x01\xa0\x00\x00\x1f\x14\x14\x03\x02\x00I\xd
f\x05\x001E\x0b\x00\x0c\x00\x1c\x00\x00\x01\x02\x1f\x00\x00e\x00\x01\x00\x00\x02\x
01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
```

Control

Frame Counter: 14630

Fake Messages: [✓ eth1] eth2

Drop Messages: eth1 Fraction [%]: 10 [✓ act.] eth2 Fraction [%]: 50 act.

Figure A 11 Network manipulation software, screenshot

After starting the software, no information is initially displayed in the "Data Frames" area (see screenshot). In the lower area ("Control"), only a continuous counter ("Frame Counter") runs up, which displays the number of data packets transmitted (and possibly changed) since the software was started (in both directions).

By pressing the "eth1" and "eth2" buttons (under "Get Single Frame"), a single data packet can be displayed at any time. By pressing "eth1", the data packet currently being received by the network manipulator at Ethernet port 1 and forwarded via Ethernet port 2 is displayed. The same applies to "eth2".

By selecting "eth1" or "eth2" under "Fake Messages", the message currently recorded and displayed in the data area can be sent again and again instead of the currently received data packets (although the counter in the TXS payload and the TXS checksum are adjusted when sending - see also section 2.3 on potential cyberattacks).[30]

Under "Drop Messages", the proportion of data packets that are not forwarded to the other Ethernet connection can be set for both Ethernet connections (the specific selection is made statistically by random selection).

[30] These manipulations are irrelevant for the simulation of random failures and only play a role in (simulated) cyberattacks. As cyber security was not the subject of this project, the functionality here is still quite simple. However, it would be possible, for example, to extend the software in such a way that a whole series of data packets are recorded and then sent to the recipients later (e.g., in a loop). This would completely conceal manipulation beyond the network manipulators from the recipients.

In addition to this software with a (simple) graphical interface, a number of other Python scripts were created that were necessary for further experiments (for example, not forwarding every n^{th} data packet).

As the tests showed that the network communication only knows two relevant states (either the communication is still working or it has failed self-reporting), in addition to using the network manipulators, it is also possible to implement network communication failures using TXS replacement circuits (in the sense of fault injection). This is explained in the following section.

Nevertheless, the future further development of network manipulators, especially for the investigation of cyberattack scenarios, is certainly indicated.

A.2.3 TXS replacement circuit for network manipulation

The approach presented in this section of using TXS replacement circuits instead of network manipulators has not yet been used for the tests carried out in this project. Rather, this approach is a result of these tests and the knowledge gained from them.

Since network communication can only have two states[31], a few TXS function blocks can be used to create simple circuits that lead to exactly the same behavior as the network manipulators.

This is illustrated (representatively) by Figure A 11.

[31] Note: The third state ("flickering"), e.g., in Table A 1, can easily be interpreted as alternating between "ok" and "SF" (self-reporting failure).

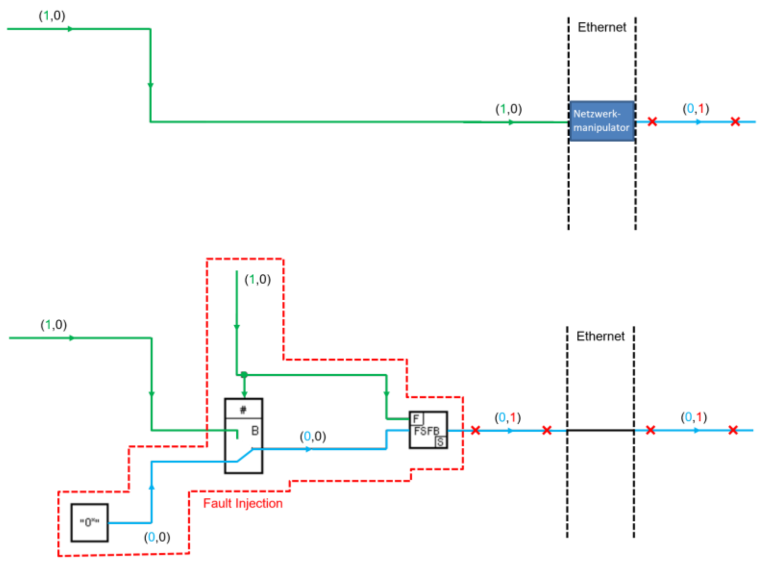


Figure A 12 TXS replacement circuit for a network manipulator ("fault injection"), representative example

In the upper part, an error-free binary 1 is transmitted via Ethernet. This means that the signal to be transmitted has the value 1, the error attribute (see also /MCH 21/) of the signal is 0. If this is changed by the network manipulator or not forwarded, this is interpreted by the receiver on the right-hand side as (default value) 0 with the error attribute 1.
The same can also be achieved with the additional TXS function blocks within "Fault Injection" in the image below. There, the same effect can be achieved with an additional signal (which is specified via the corresponding interface, for example), using a switch and a so-called FSFB[32] function block.

A.3 Model systems used

Detailed descriptions of most model systems (and the corresponding nomenclature) can be found in /MCH 18/ and /MCH 21/. At this point, the model systems used in this project are only presented in compact form using their representation in Matlab/Simulink. Comparatively simple model systems were also implemented for analyses using the real safety I&C system from AnTeS (AnTeS-SIC-real) (see e.g., Figure 3.2 on page 24).

[32] The name of the function block (FSFB) is not explained in the available documentation (e.g., /TXS 12/). It probably stands for fault setting function block (or similar). In any case, a logical 1 at input "F" sets the error attribute of the incoming signal at the output.

The first two model systems (A122 and A222) as Matlab/Simulink models (AnTeS-SILT-sim) are shown together in Figure A 13. The fact that two different model systems can actually be simulated together here is illustrated by the following two figures (Figure A 14 and Figure A 15).

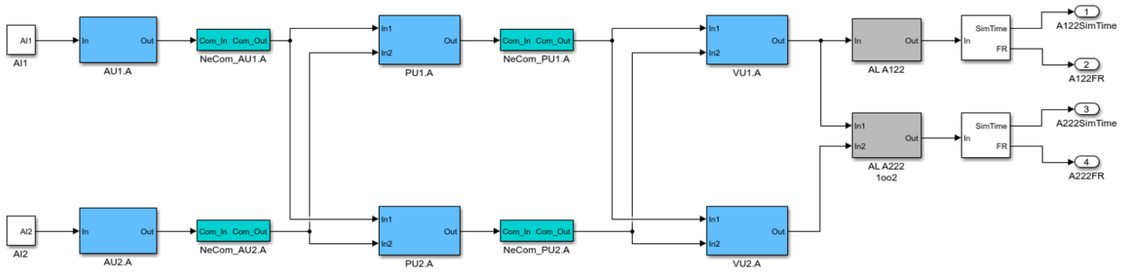


Figure A 13 Model systems A122 and A222 in Matlab/Simulink

Figure A 14 is identical to Figure A 13, only the part of the overall model representing the model system A122 has been outlined in red. For the evaluation (determination of the failure rate achieved so far during simulation) in the green-framed part of the image, only the output signal of the VU1.A (via an additional analog logic (AL), which is always present for reasons of comparability) is used. The corresponding calculated value therefore belongs to a system with 2 AUs, 2 PUs and 1 VU (of "subsystem" A) - A122.

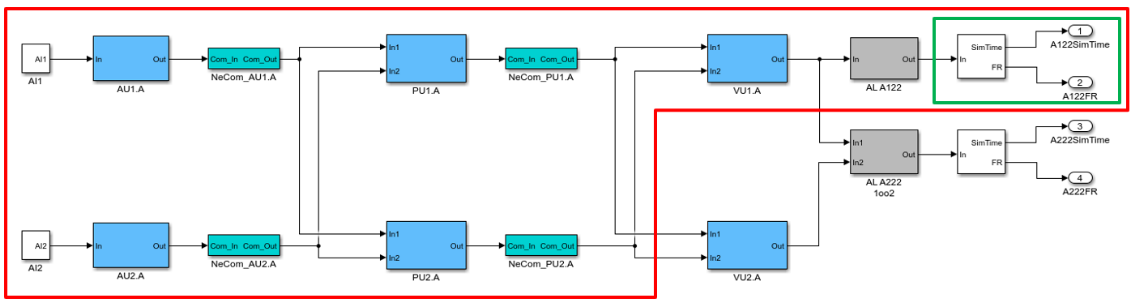


Figure A 14 Model system A122 (outlined in red)

Accordingly, the outputs of both VUs are taken into account in the red-framed subsystem in Figure A 15 - A222.

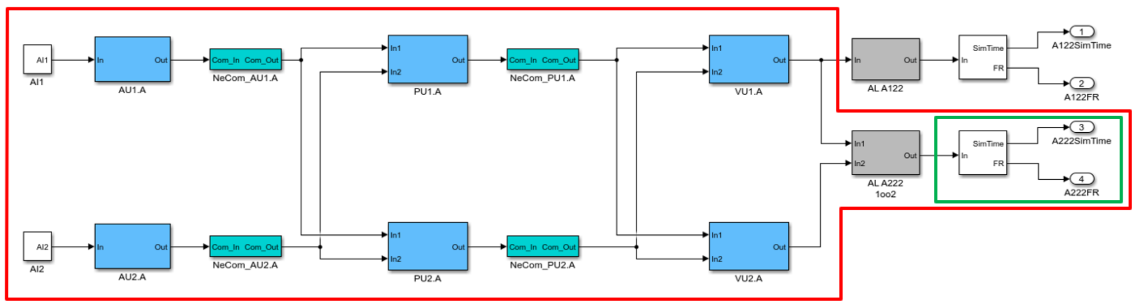


Figure A 15 Model system A222 (outlined in red)

Other model systems used are shown in the following figures.

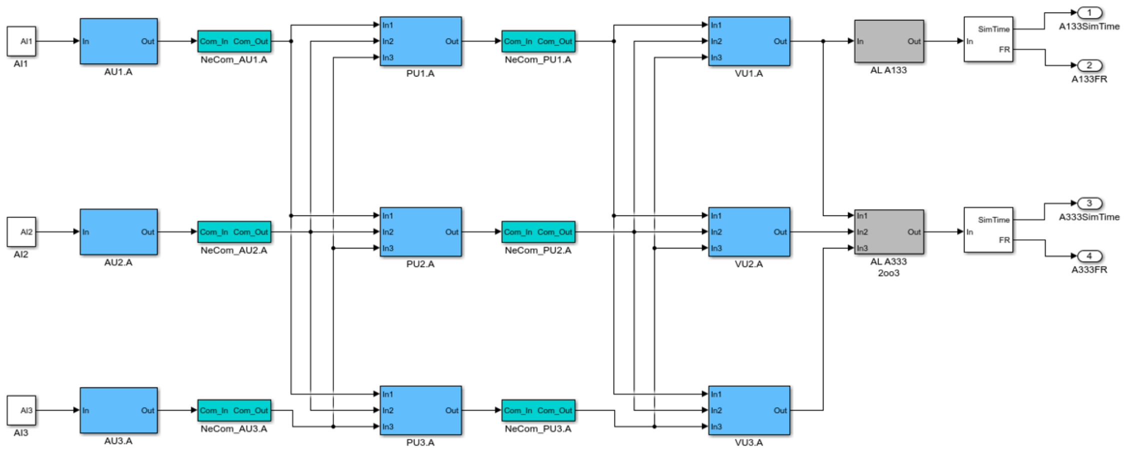


Figure A 16 Model systems A133 and A333 in Matlab/Simulink

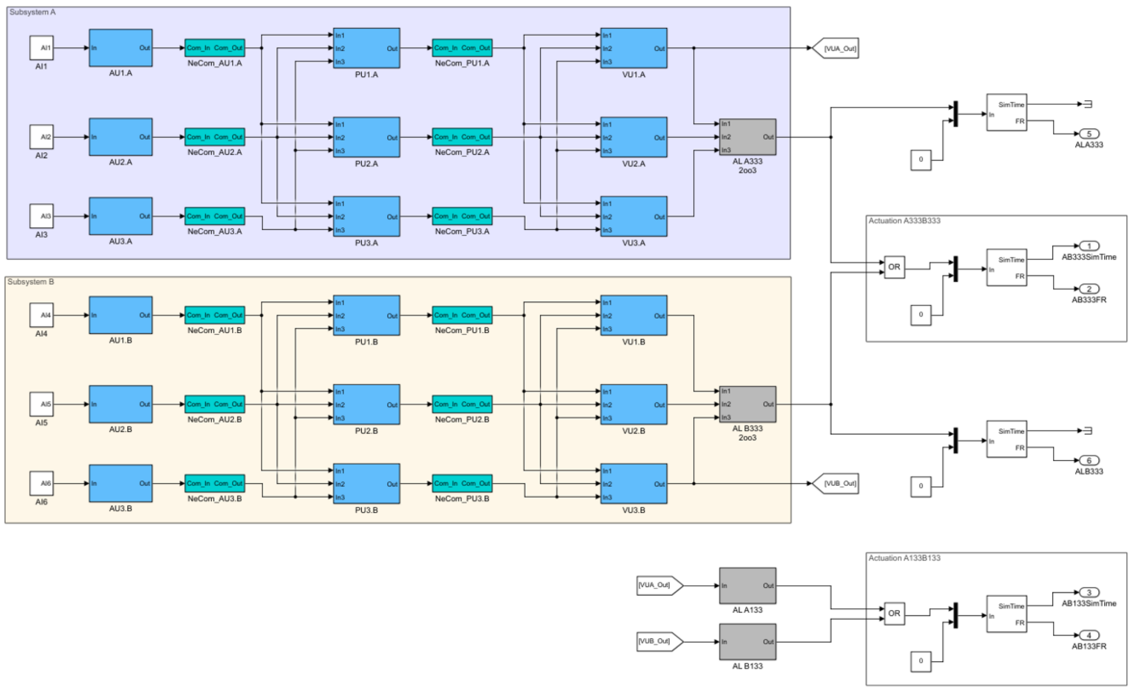


Figure A 17 Model systems A133B133, A333B333 as well as A333 and B333 in Matlab/Simulink

The results in Figure 3.8 on page 32 also show three other model systems (A222mod, A122mod and A333ext):

- A122mod:
  - Is largely identical to A122, only one function block (specifically a 2nd MAX block) has been replaced in both PUs (by a MAX block).
  - A122mod looks like A122 on the display level[33] shown in Fig. A 13.
  - Reasons and further explanations can be found in /MCH 21/.
- A222mod:
  - Is largely identical to A122, only one function block (specifically a 2nd MAX block) has been replaced in both PUs (by a MAX block).
  - A122mod looks like A122 on the display level shown in Fig. A 13.
  - Reasons and further explanations can be found in /MCH 21/.
- A333ext:
  - Is basically identical to A333.

[33] The model systems shown in the figures in this section show these on the top display level. Each "box" contains so-called Simulink subsystems, e.g., the entire function plans of the I&C functions used.

- Instead of internal random numbers, externally determined random numbers were fed into the model in each calculation step as a test.
- This was an attempt to speed up the calculations overall. However, as the Simulink models are available in the C language (compiled as fairly "fast" DLL files), no increase in speed could be achieved here.
- Apart from slightly different accuracies of the random numbers used, A333 and A333ext can therefore be regarded as identical.

B Explanations of terms

B.1 Broadcast in networks

Broadcast in networks is a mechanism that makes it possible to send data packets to all devices in a network. The purpose of broadcast is to facilitate communication between devices without the sender having to know the exact address of each recipient. For example, broadcast can be used to send ARP requests to determine the MAC address of a specific device. ARP (Address Resolution Protocol) requests are network packets that are used to determine the physical MAC address of a device on a network.

B.2 Bytes and their representation

A byte is a basic unit of digital information and consists of 8 bits. Each bit can have the value 0 or 1 and therefore represents a binary number (0 or 1). A byte is represented as a binary number by stringing together 8 bits (e.g.: 00101101).

A byte is represented as a decimal number by converting the binary number into the decimal system. Each bit has a value of 2 to the power of n, where n runs from 0 to 7. The decimal number is calculated by adding the products of the bits with their values. For example:

00101101 (binary) = $2^5 + 2^3 + 2^2 + 2^0 = 32 + 8 + 4 + 1 = 45$ (decimal).

The representation of a byte as a hexadecimal number is done by grouping every 4 bits and converting them into the corresponding hexadecimal digit. Each nibble (4 bits) has a value of 2 to the power of n, where n runs from 0 to 3. The hexadecimal numbers are represented by the symbols 0-9 and A-F. For example:

00101101 (binary) = 0010 (binary) 1101 (binary) = 2 (hexadecimal) D (hexadecimal)
= 2D (hexadecimal).

Overall, these representations (binary, decimal, and hexadecimal) are important tools for analyzing and processing digital data and offer different perspectives on the information contained in a byte. Often (as in this report), bytes are represented by hexadecimal numbers with a prefix “x”, for example, x2D.

B.3 Ethernet

Ethernet is a standard for wired data transmission in local area networks (LANs). It is a common standard for wired network technology and is most frequently used in computer networks. IEEE 802.3 describes the physical and the MAC protocol for the transmission of data packets between network devices. It defines the physical cabling, the transmission rate, the signaling, error detection and correction, as well as collision avoidance in the network.

The corresponding standard supports various transmission mediums such as coaxial cable, twisted pair cable, and fiber optic. It also defines various speeds, from 10 Mbit/s up to (currently) 400 Gbit/s.

IEEE 802.3 Ethernet is a widely used and established standard in network technology and is employed in a variety of applications such as offices, data centers, and industrial networks.

B.3.1 Ethernet hubs and switches

Ethernet hubs forward incoming data packets to all connected devices, regardless of whether they need to receive the packet or not. Ethernet switches, on the other hand, examine incoming data packets and send them only to the destination device for which they are intended. This reduces network load and increases security, as no unnecessary data packets are sent to unauthorized devices. Overall, Ethernet switches offer higher performance and functionality than Ethernet hubs and are therefore preferred in most networks. In the experiments conducted within this project, the otherwise disadvantageous behavior of Ethernet hubs, where all data packets are forwarded to all connected devices, was desirable for "sniffing" (monitoring), as this allows (e.g., using software like Wireshark) a simple way to monitor the traffic.

B.4 MAC address

A MAC address (Media Access Control Address) is a unique identifier assigned to network devices at the data link layer (Layer 2 in the OSI model). Each network interface, such as an Ethernet or Wi-Fi card, has a unique MAC address that consists of 48 bits (6 bytes) and is usually represented in hexadecimal numbers. The first 24 bits (3 bytes) of the MAC address identify the manufacturer of the network adapter, while the last 24 bits (3 bytes) represent a unique identifier of the adapter. The MAC address is used to transmit data packets directly between two network devices within a local network (LAN) by embedding it as the destination address in the packets. Therefore, the MAC address is an important element in communication within local networks. In the context of TXS, MAC addresses of the involved components are configurable. It is important to ensure that uniqueness (at least within a system) is not compromised.

B.5 OSI model/layers

The OSI (Open Systems Interconnection) layer model is a reference model for network protocols as a layered architecture /WIK 23/ /GEE 22/. It has been published as a standard by the International Telecommunication Union (ITU) since 1983 and by the International Organization for Standardization (ISO) since 1984 /ISO 94/. Each layer provides a specific function and works together with the other layers to enable communication between devices in a network.

The seven layers of the OSI model are:

1. Physical Layer: This layer describes the physical transmission of data across the network, including the type of cabling, signal strength, and transmission rate.
2. Data Link Layer: This layer ensures that data is correctly transmitted over the network by providing error detection and correction as well as addressing of network devices.
3. Network Layer: This layer is responsible for addressing and routing data packets in the network. It uses IP addresses to forward data packets to the correct destination device.
4. Transport Layer: This layer is responsible for the transmission of data between applications on different devices and ensures that data is transferred in the correct order and without errors.
5. Session Layer: This layer is responsible for managing and synchronizing sessions between applications on different devices.
6. Presentation Layer: This layer is responsible for the presentation of data by translating data formats and encoding of data packets so that they can be read and understood by different devices.

7. Application Layer: This layer provides applications that use network resources, such as email, web browsers, or file-sharing applications.

The OSI model is an important concept for network communication, as it provides a common understanding of how networks function and enables the identification and resolution of network problems at various levels.

B.6 Sniffing

"Sniffing" is a term from computer science and refers to the interception and monitoring of data packets transmitted over a network. The data packets are captured and analyzed by a special program or a so-called "sniffer" without impairing the transmission. Sniffing is often used for security and diagnostic purposes but can also be used for unauthorized eavesdropping and data misuse.

B.7 Wireshark

Wireshark /WIS 23/ is a free, open-source software for network analysis. It is used by network administrators, security analysts, and developers to monitor, analyze, and diagnose network traffic. Wireshark allows recording and displaying traffic from various network protocols, such as TCP, UDP, HTTP, DNS, and many others. It offers a graphical user interface to visualize, and filter captured data, facilitating the analysis of specific network events. Wireshark is a powerful software that can help identify and resolve network problems and enhance network security.

In this project, Wireshark was used for preliminary investigations to analyze the network traffic in the networks of TXS.

Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) gGmbH

Schwertnergasse 1
50667 Köln
Telefon +49 221 2068-0
Telefax +49 221 2068-888

Boltzmannstraße 14
85748 Garching b. München
Telefon +49 89 32004-0
Telefax +49 89 32004-300

Kurfürstendamm 200
10719 Berlin
Telefon +49 30 88589-0
Telefax +49 30 88589-111

Theodor-Heuss-Straße 4
38122 Braunschweig
Telefon +49 531 8012-0
Telefax +49 531 8012-200

www.grs.de

ISBN 978-3-910548-56-5