Modellierung des Zeit- und Fehlerverhaltens industrieller Funklösungen zur Bewertung der Koexistenz

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieurin / Doktoringenieur
(Dr.-Ing.)

von Dipl.-Ing. André Schimschar

geb. am 21.01.1983 in Schönebeck/Elbe

genehmigt durch die Fakultät Elektrotechnik und Informationstechnik der Otto-von-Guericke-Universität Magdeburg

Gutachter:

Prof. Dr.-Ing. Christian Diedrich

Prof. Dr.-Ing. habil. Helmut Beikirch

Promotionskolloquium am 11.09.2013

Vorwort

Die vorliegende Dissertation entstand am Institut für Automation und Kommunikation e.V. (ifak) Magdeburg im Schwerpunkt „Drahtlose Industrielle Kommunikation“. Eine Betreuung von Seiten der Otto-von-Guericke Universität Magdeburg erfolgte am Institut für Automatisierungstechnik durch den Lehrstuhl „Integrierte Automation“ unter der Leitung von Prof. Dr.-Ing. Christian Diedrich.

An dieser Stelle möchte ich mich bei allen bedanken, die zum erfolgreichen Abschluss meiner Dissertation beigetragen haben.

Insbesondere möchte ich der Phoenix-Contact-Stiftung meinen Dank für die dreijährige finanzielle Unterstützung in Form eines Stipendiums aussprechen.

Der größte Dank gehört allerdings meinen Eltern, die mich während des Studiums immer wieder motiviert, aber niemals unter Druck gesetzt haben.

Meine Kollegen Dr. Lutz Rauchhaupt, André Gnad, Marko Krätzig und René Hensel haben mich fachlich unterstützt, standen für tiefergehende Diskussionen bereit und gaben mir immer wieder neue Denkansätze und Hinweise. Unsere tägliche Zusammenarbeit war für mich eine Riesenfreude.

Einen großen Respekt zolle ich meinem Doktorvater Christian Diedrich, der mich sehr engagiert betreut hat, mir mit vielen Ratschlägen und Anregungen zur Seite stand und damit maßgeblich zur Realisierung dieser Arbeit beigetragen hat.

Neben Prof. Dr.-Ing. Christian Diedrich möchte ich Prof. Dr.-Ing. habil. Helmut Beikirch für die Übernahme der Begutachtung der vorliegenden Arbeit danken.

Kurzfassung

Die Vorteile des Einsatzes von drahtlosen Kommunikationssystemen in der industriellen Automation sind unumstritten. Allerdings verlangt die Vielfalt an Anwendungsmöglichkeiten wie Kabelersatz, fahrerlose Transportsysteme oder die Übermittlung von Sensordaten von rotierenden Geräten sowie die Überbrückung großer Entfernungen verschiedene Funklösungen, welche die Anforderungen der jeweiligen Applikation erfüllen können. Diese Lösungen basieren hauptsächlich auf den Spezifikationen IEEE 802.11b/g, IEEE 802.15.1 oder IEEE 802.15.4. Allerdings existiert noch eine Vielzahl anderer drahtloser Lösungen, die das lizenzfreie 2,4 GHz ISM Band nutzen. Es ist offensichtlich, dass die Vorteile der Funklösungen für die industrielle Automation zu einer erweiterten Nutzung dieses begrenzten Spektrums führen.

Das wesentliche Problem beim Einsatz von Funksystemen in der industriellen Automation ist die gegenseitige Beeinflussung der Funksysteme und die damit verbundene Verschlechterung des Zeit- und Fehlerverhaltens durch beispielsweise verlängerte Wartezeiten beim Medienzugriff oder durch Fehlversuche bei der Übertragung. Im Extremfall können die Anforderungen der Applikation nicht mehr erfüllt werden. Es drohen Produktionsausfälle und damit erhebliche Kosten.

Obwohl mittlerweile die drahtlose Funkkommunikation fester Bestandteil der industriellen Automation ist und viele Automatisierungsanwendungen auf Funk zurückgreifen, werden die Anwender von Funksystemen bei der Frage der Koexistenz oft allein gelassen. Verschiedene Untersuchungen zur Analyse der gegenseitigen Beeinflussung von drahtlosen Lösungen derselben oder unterschiedlicher Technologien sind veröffentlicht worden. Maßnahmen zur Erreichung der Koexistenz wie ein geeignetes Koexistenzmanagement oder kognitive Verfahren existieren ebenfalls. Des Weiteren kommen auch verschiedene Simulationswerkzeuge zum Einsatz, um das Koexistenzverhalten unterschiedlicher Funksysteme untersuchen zu können. Es ist allerdings kein Simulationstool verfügbar, welches das Koexistenzverhalten aus Sicht der industriellen Automation betrachtet. Dieses Tool sollte ebenfalls die Grenzen der drahtlosen Lösungen herausfinden und freie Ressourcen identifizieren können.

In dieser Arbeit wird ein Modellansatz zur Simulation drahtloser Koexistenzszenarien präsentiert, welcher auf hierarchischen und zeitbehafteten kolorierten Petri-Netzen basiert. Dadurch ist die Analyse des Einflusses von Interferenzen auf drahtlose Geräte und Netzwerke möglich. Außerdem kann eine Leistungsbewertung mit Kriterien aus Sicht der industriellen Automation erfolgen. Der Modellansatz wird zunächst auf die Verfahren der ETSI EN 300 328, welche den Zugriff auf das 2,4 GHz ISM Band reglementieren, angewendet. Am Beispiel von WLAN und WSAN-FA werden mit dem vorgestellten Ansatz auch reale Systeme modelliert. Ein entsprechendes Kanalmodell und dessen Implementierung ist ebenfalls Bestandteil dieser Arbeit.

Des Weiteren werden neue Lösungen für Medienzugriffsverfahren der industriellen Automation vorgestellt, modelliert und simuliert. Dazu wird zunächst MS-Aloha, eine Funktechnologie aus der Car-to-Car-Kommunikation, untersucht und dessen Schwächen offengelegt. Außerdem wird ein eigener Ansatz ausgehend von DECT hergeleitet und modelliert. Die dazugehörigen Simulationsergebnisse zeigen, dass dieser Ansatz für den Einsatz in der industriellen Automation zu empfehlen ist.

Abstract

The benefits of using wireless communication systems in industrial automation are undisputed. However, the variety of application possibilities such as cable substitution, autonomous guided vehicles, transmission of sensor data of rotating devices or bridging of large distances demands various wireless solutions in order to fulfill the requirements of the particular application. These solutions are mandatory based on the specifications IEEE 802.11b/g, IEEE 802.15.1 or IEEE 802.15.4. However, there are a multitude of other wireless solutions using the licence-free 2.4 GHz ISM band. It is evident that the benefits of wireless solutions for industrial automation lead to an extended use of the limited spectrum.

The significant problem when using wireless systems in industrial automation is the mutual influence of wireless systems and the associated degradation of time and error behaviour by, for instance, longer waiting times during media access or failed attempts at transmission. In the worst case, the requirements of application can no longer be fulfilled. There is a risk of production losses which could lead to substantial costs.

Although wireless communication is meanwhile a fix part of industrial automation and many automation applications resort to radio communication, the user is often left alone with the question of coexistence. Various investigations into analysing the mutual influence between wireless solutions of the same or different technology have been published. Measurements for reaching coexistence such as a suitable coexistence management or cognitive mechanisms also exist. In addition, various simulation tools can be used to investigate the coexistence behaviour of different wireless systems. However, there is no simulation tool available which considers the coexistence behaviour from the point of view of the industrial automation and which finds the limits of a wireless solution and is able to identify free resources.

In this thesis a model approach for simulating wireless coexistence scenarios is presented, which is based on hierarchical and timed coloured Petri nets. Thereby, an analysis of the influence of interferences on wireless devices and networks becomes possible. Moreover a performance evaluation with criteria from the point of view of the industrial automation can occur. First of all the model approach is applied to methods of ETSI EN 300 328, which regularise access to the 2.4 GHz ISM band. Using the example of WLAN and WSAN-FA also real systems are modelled with the introduced approach. An appropriate channel model and its implementation are also part of this thesis.

Furthermore, new solutions of media access mechanisms for industrial automation are introduced, modelled and simulated. Therefore at first MS-Aloha, which is a radio technology for car-to-car-communication, is investigated and its failings are revealed. Moreover an own approach based on DECT is derived and modelled. The associated simulation results show that this approach can be recommended for the use in industrial automation.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ABB	Asea Brown Boveri
ACK	Acknowledgement
AFH	Adaptive Frequency Hopping
AP	Access Point
APPL	Application Layer
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
BS	Basisstation
CCA	Clear Channel Assessment
CDMA	Code Division Multiple Access
CENELEC	European Committee for Electrotechnical Standardization
CEPT	Conférence Européenne des Administrations des Postes et des Télécommunications
CINR	Carrier-to-Interference-Noise-Ratio
CIR	Carrier-to-Interference-Ratio
CLDPS	Connectionless DECT Packet Service
CNR	Carrier-to-Noise-Ratio
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
CSV	Comma-Separated Values
DAA	Detect-And-Avoid
DCF	Distributed Coordination Function
DECT	Digital Enhanced Cordless Telecommunications
DFS	Dynamic Frequency Selction
DIFS	Distributed Inter Frame Space
DLL	Data Link Layer
DUT	Device-Under-Test
ECCA	Extended Clear Channel Assessment
EMC	ElectroMagnetic Compatibility
EN	Europäische Norm
ERM	ETSI Committee on EMC and Radio Spectrum Matters
ETSI	European Telecommunications Standards Institute
FB	Frame Based
FDD	Frequency Division Duplex
FDMA	Frequency Division Multiple Access
FFP	Fixed Frame Period
FH	Frequency Hopping
FHSS	Frequency Hopping Spread Spectrum

FI ..... Frame Information Structure
FIFO ..... First-In-First-Out
FN ..... Framenummer
FT ..... Fixed Terminal
FTT ..... Funk-Transfer-Tester
GPRS ..... General Packet Radio Service
GPS ..... Global Positioning System
GSM ..... Global System for Mobile Communications
HART ..... Highway Addressable Remote Transducer
IA ..... Industrielle Automation
ID ..... Identifikationsnummer
IEC ..... International Electrotechnical Commission
IEEE ..... Institute of Electrical and Electronics Engineers
IFS ..... Inter Frame Space
IMA ..... I'm Alive
ISA ..... International Society of Automation
ISM ..... Industrial, Scientific, and Medical
ISO ..... International Organization for Standardization
ITU ..... International Telecommunication Union
LAT ..... Listen-After-Talk
LB ..... Load Based
LBT ..... Listen-Before-Talk
LLC ..... Logical Link Control
LOS ..... Line-Of-Sight
LTE ..... Long Term Evolution
MAC Layer ..... Media Access Control Layer
MS-Aloha ..... Mobile Slotted Aloha
MU ..... Medium Utilization
NA ..... Non-Adaptive
NFH-FB ..... Non-FH systems frame based devices using LBT based DAA
NFH-LB ..... Non-FH systems load based devices using LBT based DAA
NFH-NA ..... Non-Frequency Hopping non-adaptive systems
NFH-NLBT ..... Non-Frequency Hopping systems using non-LBT based DAA
OFDM ..... Orthogonal Frequency Division Multiplexing
OSI ..... Open Systems Interconnections
PCF ..... Point Coordination Function
PER ..... Packet Error Rate
PHY ..... Physical Layer
PIFS ..... Point Coordination Inter Frame Space
PLR ..... Packet Loss Rate

PNO ..... PROFIBUS Nutzerorganisation
PT ..... Portable Terminal
QoS ..... Quality of Service
R&TTE ..... Radio and Telecommunications Terminal Equipment
RF ..... Radio Frequency
SA ..... Sensor/Aktor
SCC ..... Strongly Connected Component
SDR ..... Software Defined Radio
SIFS ..... Short Inter Frame Space
SNR ..... Signal-to-Noise-Ratio
SRD ..... Short Range Device
SSID ..... Service Set Identifier
TDD ..... Time Division Duplex
TDMA ..... Time Division Multiple Access
TG ..... Technical Group
UMTS ..... Universal Mobile Telecommunications System
VANET ..... Vehicular Ad Hoc Network
WAP ..... Wireless Application Protocol
WIA-PA ..... Wireless networks for Industrial Automation - Process Automation
WISA ..... Wireless Interface for Sensors and Actuators
WLAN ..... Wireless Local Area Network
WSAN-FA ..... Wireless Sensor Actor Network - Factory Automation

1 Einleitung

In den letzten 20 Jahren hat die drahtlose Kommunikation mehr und mehr den Weg in das tägliche Leben gefunden. In den Anfängen spielte vor allem die Einführung und die Verbreitung des Handys eine wichtige Rolle. Die Möglichkeit, ständig und überall erreichbar zu sein, führte sogar dazu, dass in Deutschland mehr Handyverträge als Einwohner existierten. Nachdem das Handy Bestandteil des Alltags wurde, hielten auch im Bereich der Heim- und Bürokommunikation verschiedene Funktechnologien Einzug. Dabei wurde das Wireless Local Area Network (WLAN, [9]) zur Anbindung an ein bestehendes Netzwerk oder zur direkten Ankopplung an einen Internetrouter entwickelt. Bluetooth ([10]) dagegen fand Anwendung bei kabellosen Headsets, bei der Handy-zu-Handy-Kommunikation, aber auch bei der drahtlosen Verbindung einer Tastatur oder Maus mit dem PC. Vermehrt werden heute schon drahtlose Sensoren für die Gebäudeautomatisierung eingesetzt. Diese dienen zur Messung der Temperatur, der Luftfeuchtigkeit, des Lichts oder des CO_2-Gehaltes. Diese Sensoren werden entweder durch Batterien mit Energie versorgt oder sie nutzen Energy Harvesting. Für die energiesparende Kommunikation dieser drahtlosen Sensoren wurde die Spezifikation IEEE 802.15.4 ([12], [14]) erarbeitet. Weiterhin bietet die Firma EnOcean Module an, welche das Energy Harvesting und die Funkkommunikation vereinen, jedoch nicht der IEEE 802.15.4 entsprechen ([29]).

Wie im Heim- und Bürobereich wurden die Vorteile der drahtlosen Kommunikation auch von der industriellen Automatisierungstechnik aufgegriffen. Dabei ist das ausschlaggebendste Argument die Erhöhung der Mobilität und Beweglichkeit. Dies ist besonders bei rotierenden Applikationen oder bei fahrerlosen Transportsystemen wichtig. Hinzu kommt die Vereinfachung der Installation von Geräten. Der Einsatz von drahtlosen Kommunikationssystemen als reiner Kabelersatz bietet sich ebenfalls an. Damit wird zum Beispiel die Anbindung schwer zugänglicher Bereiche ermöglicht. Jedoch sollen drahtgebundene Kommunikationssysteme auf keinen Fall ersetzt werden. Stattdessen sollen mit ihnen neue Anwendungsbereiche erschlossen werden. So ist beispielsweise die Erweiterung von Prozessanlagen um zusätzliche Sensoren möglich. Die zu den bisher überwachten Signalen hinzukommenden Daten können dann dabei helfen, ein bestehendes Prozesssystem weiter zu optimieren. Neben dem Monitoring kann mit Funksystemen eine Lokalisierung einzelner Teilnehmer vorgenommen werden. Innerhalb einer Industriehalle ist damit beispielsweise der Weg von Gabelstaplern nachverfolgbar. Die durch den Einsatz drahtloser Kommunikationssysteme hinzugewonnene Flexibilität sollte schon während der Erstellung des Automatisierungskonzeptes einer Anlage Berücksichtigung finden.

Die genannten Anwendungen stellen unterschiedliche Anforderungen an das drahtlose Kommunikationssystem. Dabei ist eine Funktechnologie für eine Anwendung mehr oder weniger gut geeignet. Bei der Einführung von Funk in industriellen Anwendungen wurden

zunächst die bestehenden Technologien aus dem Heim- und Bürobereich übernommen: WLAN, Bluetooth und IEEE 802.15.4. Diese zeigen jedoch durch ihre Funktionsprinzipien nicht nur vorteilhafte Eigenschaften für die Automatisierungstechnik, welche besonders die Echtzeitfähigkeit und Zuverlässigkeit der Datenübertragung fordert. Beispielsweise hat WLAN Nachteile im Zeitverhalten, da CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance) als Medienzugriffsverfahren verwendet wird. Vor der Übertragung eines Paketes wird vom Kommunikationsteilnehmer eine zufällige Wartezeit generiert. Außerdem ist die Überprüfung auf ein unbelegtes Medium, welche während der zufälligen Wartezeit durchgeführt wird, leicht störbar. Eine fehlerfreie Übertragung innerhalb eines definierten Zeitrahmens ist mit WLAN aus diesen Gründen nicht garantierbar.

Um die für die Heim- und Bürokommunikation optimierten Systeme industrietauglicher zu machen, wurden an den bestehenden Technologien Anpassungen vorgenommen. So optimierten die Hersteller ihre WLAN-Systeme auf einen schnelleren Verbindungsaufbau beziehungsweise Zellenwechsel. Die Zuverlässigkeit konnte durch die Verwendung geringer Datenübertragungsraten erhöht werden. Ein anderer Ansatz sieht die Integrierung der Point Coordination Function (PCF) anstelle der bei WLAN gebräuchlichen Distributed Coordination Function (DCF) vor. Bei PCF bekommt jeder Client vom Access Point (AP) einen bestimmten Zeitschlitz zugewiesen, in dem er auch dann sendet, wenn das Medium scheinbar belegt ist. Doch nicht nur an WLAN, sondern auch an Bluetooth wurden Änderungen vorgenommen. Der Bluetooth-Stack wurde so optimiert, dass geringere Übertragungszeiten erreicht werden können. Dazu sind ungenutzte Profile per Konfiguration abschaltbar. Notwendige Profile, beispielsweise für die serielle Übertragung, wurden verbessert. Von Bluetooth wurde WISA (Wireless Interface for Sensors and Actuators) abgeleitet. Es ist für den Einsatz in der Fertigungsindustrie ausgelegt und basiert wie Bluetooth auf der Spezifikation IEEE 802.15.1. Aus WISA wurde mit geringen Veränderungen WSAN-FA (Wireless Sensor Actor Network - Factory Automation) entwickelt und in der PROFIBUS Nutzerorganisation (PNO) spezifiziert.

WSAN-FA gilt damit für die PNO, neben den für die drahtlose Übertragung von PROFINET-Telegrammen zugelassenen WLAN und Bluetooth, als dritte Funkspezifikation für die Fertigungsindustrie. Das Ziel von WSAN-FA ist die drahtlose Anbindung von Sensoren und Aktoren an eine Maschinensteuerung. Da die Sensorsignale in der Regel binär sind, besteht für die Nutzdaten eine Begrenzung von 2 Byte. Als ein Beispiel sei die drahtlose Ankopplung von Näherungssensoren genannt. Dabei erlaubt WSAN-FA eine echtzeitfähige und robuste Übertragung, wie sie in einer Roboterzelle von Nöten ist. Die Topologie ist sternförmig.

Im europäischen Raum hat sich WirelessHART als Funkstandard, welcher in der IEC 62591 ([6]) normiert ist, für die Prozessindustrie durchgesetzt. WirelessHART ist für das Asset-

Management von Geräten, das Sammeln analoger Messwerte und das drahtlose Ansteuern von Stellgeräten konzipiert worden. Aber auch die Erfassung digitaler Signale ist möglich. Die Teilnehmer, welche über die Prozessanlage verteilt sind, bauen ein vermaschtes Sensornetzwerk auf. Die aufgenommenen Werte werden über die einzelnen Geräte bis zu einer Datensenke, dem Gateway, weitergeleitet. Das Gateway dient zur Anbindung an den firmeninternen Feldbus, über den die notwendigen Informationen abgegriffen und aufgezeichnet werden können. Die Übertragung ist durch das Weiterreichen von Teilnehmer zu Teilnehmer zeitaufwendig. Der chinesische Standard WIA-PA (Wireless networks for Industrial Automation - Process Automation), welcher in der IEC 62601 ([7]) standardisiert ist, und ISA 100.11a ([1]), welches den Ursprung in den Vereinigten Staaten hat, sind WirelessHART relativ ähnlich. Das Vorhandensein von drei Standards für die Prozessindustrie hemmt die Integration von Funk in diesem Zweig der Automatisierungsindustrie.

Die meisten industriellen Funklösungen benutzen für die Funkkommunikation die weltweit frei verfügbaren ISM-Bänder (Industrial, Scientific, and Medical Band). Gemäß Abschnitt 5.150 der ITU Radio Regulations ([81]) sind ISM-Bänder für industrielle, wissenschaftliche und medizinische Frequenznutzungen vorgesehen. Funkkommunikationsdienste, welche diese Bänder nutzen, müssen störende Beeinflussungen durch andere Frequenznutzer hinnehmen. Da mit einer Änderung an den weltweit frei verfügbaren ISM-Bändern in den nächsten Jahren nicht zu rechnen ist, müssen die vorhandenen ISM-Bänder effizient genutzt werden. Das heißt, durch sorgfältige Planung und Konfiguration der Funklösungen und durch den umsichtigen Betrieb sollte von der kostbaren Ressource „Frequenzspektrum“ nur so viel in Anspruch genommen werden, wie für den zuverlässigen Betrieb erforderlich ist.

Die Echtzeitfähigkeit und Zuverlässigkeit der Datenübertragung des Kommunikationssystems, welches eines dieser ISM-Bänder nutzt, wird sehr stark von verschiedenen Einflussgrößen bestimmt. Die wichtigsten Einflussgrößen sind nicht nur die Entfernung oder die in den Funkgeräten eingestellten Parameter, wie z. B. die Bitrate oder die verwendete Modulation, sondern auch die Struktur der Umgebung oder das Vorhandensein anderer Funksysteme. Hindernisse in der Nähe der Funkkomponenten, wie beispielsweise Wände, Maschinen oder Menschen, beeinträchtigen nachweislich das Ausbreitungsverhalten der für die Übertragung genutzten elektromagnetischen Welle. So entsteht eine Mehrwegeausbreitung des Signals, sodass Effekte wie der Delay Spread oder das Fading auftreten. Bei der Anwesenheit anderer Funksysteme im gleichen Frequenzbereich werden die Funkgeräte auf zwei Arten beeinflusst. Betreibt das getestete System Listen-Before-Talk (LBT), so wird das Medium vor dem Versenden der Daten auf Freiheit überprüft. Im interessierenden Frequenzbereich erkennt der Sender das Medium als belegt, wenn ein definierter Parameterwert, beispielsweise der des Energielevels, überschritten wird. Ist dies der Fall, wird das Senden des zu übertragenden Datenpaketes erst gar nicht vorgenommen. Eine

andere Form der Beeinflussung durch andere Systeme findet beim Empfänger statt. Wird bei diesem das ankommende Signal durch ein anderes überlagert, können die für die Übertragung modulierten Daten nicht wieder demoduliert werden.

Der Handlungs- und Forschungsbedarf zur Untersuchung der gegenseitigen Beeinflussung von industriellen Funksystemen wird signifikant zunehmen. Ein Grund ist der stark wachsende Einsatz von Funksystemen in Fertigungs- beziehungsweise Prozessanlagen. Jedoch eignen sich die einzelnen Funktechnologien für die verschiedenen Anwendungen unterschiedlich gut, sodass jede Funktechnologie seine Daseinsberechtigung besitzt. Auch führen neue Anwendungsmöglichkeiten zu immer neuen Funktechnologien. Die existierenden Funktechnologien unterscheiden sich beispielsweise in der Modulationsart, der Sendeleistung und der Bandbreite. Die kostbare und begrenzte Funkressource wird gleichzeitig von einer Vielzahl von Geräten und Systemen, die unterschiedliche Funktechnologien verwenden, genutzt. Eine effiziente Nutzung der knappen Funkressource ist dadurch zwingend notwendig. Ebenfalls ist die gegenseitige Beeinflussung der Funksysteme zu minimieren, sodass die hohen Anforderungen der industriellen Automatisierungsanwendung erfüllt werden.

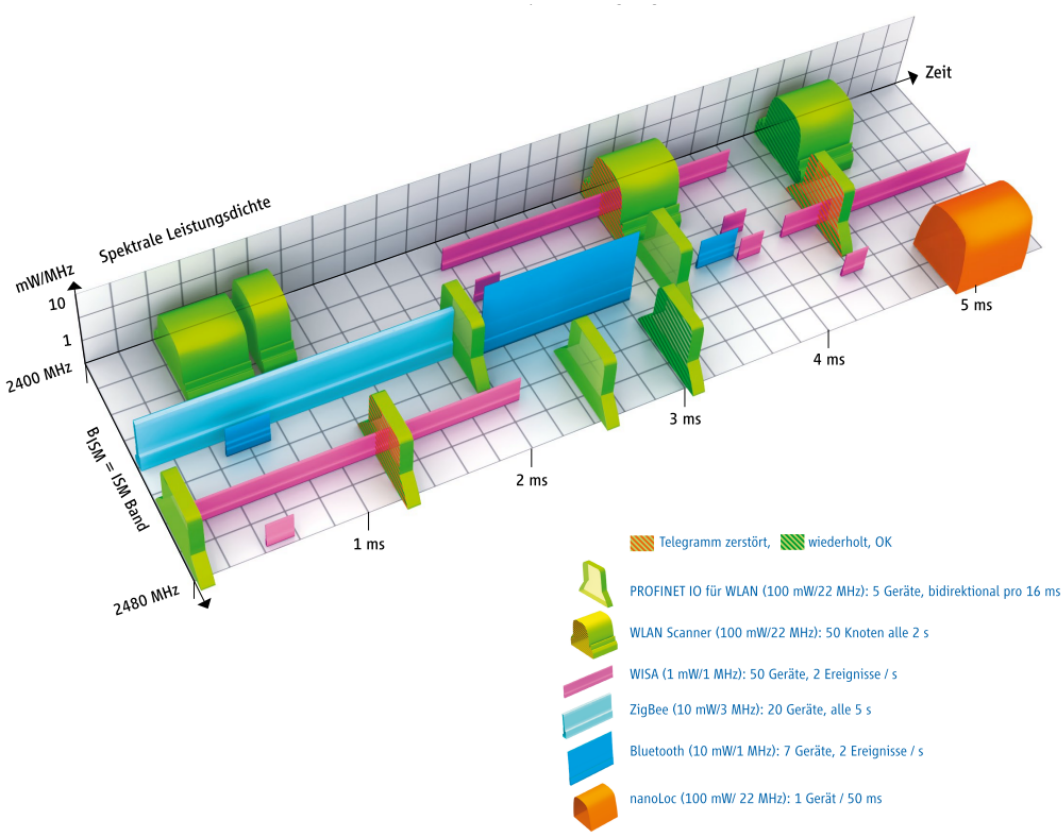

Abb. 1: *Gleichzeitige Belegung des 2,4 GHz-ISM-Bandes durch unterschiedliche Funktechnologien ([96])*

Abbildung 1 zeigt beispielhaft die Belegung des 2,4 GHz-ISM-Bandes für eine bestimmte Zeitdauer in Form eines dreidimensionalen Spektrogramms. Neben der Frequenz und Zeit

ist als dritte Dimension die spektrale Leistungsdichte aufgeführt. In der Abbildung greifen die Technologien WLAN, WISA, ZigBee, Bluetooth und nanoLoc mit ihren unterschiedlichen Medienzugriffsmechanismen, Bandbreiten und Spektralmasken auf das genannte Frequenzband zu. Dabei entstehen Paketkollisionen (orange-hellgrün schraffiert), welche Übertragungswiederholungen (hellgrün-dunkelgrün schraffiert) nach sich ziehen.

Die formale Modellierung des Zeit- und Fehlerverhaltens von industriellen Funklösungen ist das wesentliche Ziel dieser Arbeit. Sie bildet die Grundlage für verschiedene weiterführende Arbeiten. So werden durch die Modellierung und die durchzuführenden Simulationen neue Erkenntnisse gewonnen. Diese können den Ausgangspunkt für die Formulierung einfacher Regeln bilden. Die Regeln helfen dann dem Endanwender von Funksystemen bei der Sicherstellung ihres parallelen und fehlerfreien Betriebs. Weiterhin kann das erarbeitete Wissen beispielsweise in Planungs-, Diagnose- oder Analysetools Anwendung finden. Mit Hilfe der Modellierung wäre nicht nur die Analyse der auf dem Markt befindlichen industriellen Funksysteme möglich, sondern auch eine Simulation denkbar, welche die Systembetrachtung vor der eigentlichen Realisierung erlaubt. Die Modellierung unterstützt ebenfalls die Entwicklung und Überprüfung aufkommender Standards, beziehungsweise Spezifikationen, die sich in der Überarbeitung befinden. Des Weiteren ist dieser Ansatz der Modellierung generell auch zur Untersuchung von neuen Medienzugriffsmechanismen, deren Einsatzfeld besonders im Bereich der industriellen Automation liegt, geeignet.

2 Stand der Forschung

2.1 Koexistenz

2.1.1 Definition

Der Einsatz von Funk ermöglicht eine Vielfalt an Anwendungen in der industriellen Automation. Jede Anwendung hat unterschiedliche Anforderungen an die genutzte Funkkommunikation. Da sich die verschiedenen Funklösungen zur Erfüllung der Anforderungen mehr oder weniger gut eignen, hat jede einzelne der sich auf dem Markt befindlichen Funktechnologien ihre Daseinsberechtigung. Zum Beispiel gilt Bluetooth als besonders robust. WLAN ist dagegen für die Übertragung großer Datenmengen, die beispielsweise bei der Videoüberwachung fahrerloser Transportsysteme anfallen, geeignet. Die in der industriellen Automation eingesetzten Funktechnologien unterscheiden sich unter anderem in der Modulation, der benötigten Bandbreite, der maximalen Sendeleistung und im verwendeten Medienzugriffsmechanismus. Allerdings müssen sich alle Technologien dasselbe Frequenzband teilen. Die gemeinsame Nutzung der kostbaren Spektrumressource durch mehrere parallele Funkapplikationen führt zu einer gegenseitigen Beeinflussung, welche eine Verschlechterung des Zeit- und Fehlerverhaltens nach sich zieht. Es entsteht eine Forderung nach Koexistenz, um eine Veränderung des Übertragungsverhaltens zu vermeiden oder zu reduzieren. Doch was verbirgt sich hinter dem Begriff „Koexistenz“?

Eine allgemeine Festlegung trifft [62]:

> „This term refers to the capability of a wireless network to operate properly in an environment in which noise and interference are present. For reliable operation, a node in a wireless network should be able to
>
> - Coexist with thermal noise
> - Coexist with incoming multipath components
> - Coexist with other nodes within its own network
> - Coexist with independent wireless networks using the same protocols
> - Coexist with independent wireless networks using different protocols
> - Coexist with other services“

In [96] ist Koexistenz in Bezug auf die industrielle Automation wie folgt definiert:

> „Koexistenz ist ein Zustand, in dem unterschiedliche Funksysteme bezogen auf ihre Anwendungen, deren Anforderungen und Randbedingungen (z. B. Umgebung) jeweils ihre bestimmungsgemäße Funktion erfüllen, trotz Vorhandenseins anderer Funksysteme.“

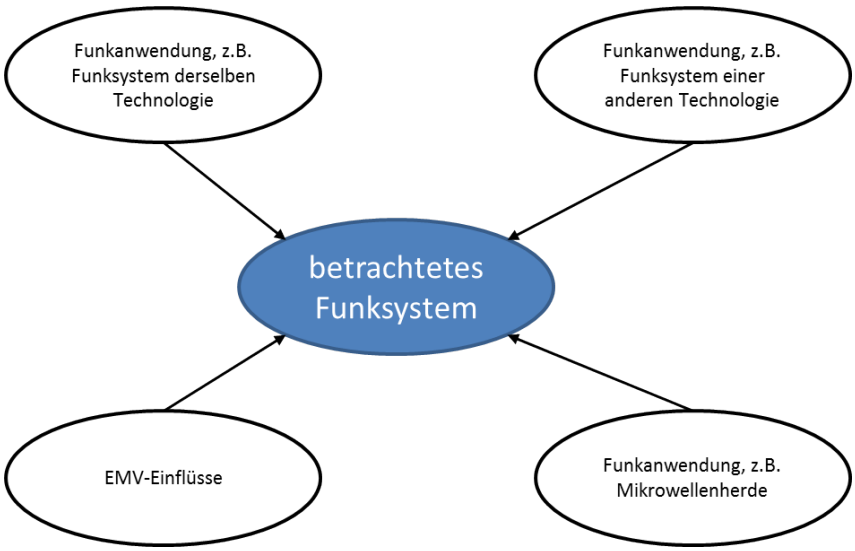

Abb. 2: *Koexistenzrelevante Störeinflüsse auf industrielle Funksysteme ([4])*

Industrielle Funksysteme sind koexistenzrelevanten Störeinflüssen ausgesetzt, die bestimmten Klassen zugeordnet werden können (siehe Abbildung 2). Neben parallelen Funkapplikationen, die entweder dieselbe oder eine andere Technologie verwenden, sind auch EMV-Einflüsse sowie andere Frequenznutzer (z. B. Mikrowellensysteme) zu nennen. Zum Erreichen des Zustandes der Koexistenz wird die Umsetzung eines Frequenz- bzw. Koexistenzmanagements empfohlen ([8], [4]). Dabei wird versucht, die einzelnen Funkanwendungen zeitlich, räumlich und im Frequenzbereich zu entkoppeln, um so die gegenseitige Beeinflussung zu reduzieren. Die strikte Durchführung des Koexistenzmanagements ist in vielen Phasen des Lebenszyklus einer Anlage notwendig, da die funkbasierte Kommunikation ständigen Änderungen unterliegt.

2.1.2 Dimensionen der Koexistenz

Neue Spektrenaufteilungstechniken von Funksystemen erlauben die gemeinsame Nutzung von orthogonalen elektromagnetischen Übertragungsressourcen. Dabei sind beispielsweise die Zeit, die Frequenz, der Code, die Polarisation, die Ausrichtung und die Position zu nennen. Diese multidimensionale Ressource kann als n-Tupel ausgedrückt werden und wird als „Übertragungshyperspace“ oder „Electrospace“ bezeichnet ([57], [27]). Viele der aufgezählten Parameter können zur Orthogonalität unter den Nutzern führen. Andere sind dagegen nicht zwingend orthogonal (z. B. Code oder Leistung), können aber trotzdem dabei helfen, Signale zu unterscheiden. In diesem Fall geschieht das mit einer statistischen Wahrscheinlichkeit ([17]). Tabelle 1 listet die Dimensionen des Hyperspace auf und ordnet ihnen die genannten Parameter zu.

2.1.3 Ansätze für Koexistenzbetrachtungen

Die Simulation und Modellierung der Koexistenz von Funksystemen sollte in zwei Schritten erfolgen. Als Erstes sollte ein rückwirkungsfreier Ansatz gewählt werden (siehe Abb. 3).

Tab. 1: *Dimensionen der Koexistenz*

Dimension	**Parameter**	**Auswirkung**
Protokoll	Zeit	Ein Frequenzkanal wird zu unterschiedlichen Zeiten genutzt und bewertet.
	Trägerfrequenz des Kanals	Für jede Funktechnologie sind eine Reihe von Kanälen definiert, die zur Übertragung ausgewählt und verwendet werden können.
Signal	Leistung	Die Leistung bestimmt das Empfangsverhalten und die Störfestigkeit.
	Signalfrequenz	Diese Frequenz beschreibt das auf die Trägerfrequenz aufmodulierte Signal.
	Code	Zur Unterscheidung können Nutzdatenströme mit speziellen Spreizcodes codiert werden.
	Polarisation	Die Polarisation beschreibt die Richtung der Schwingung des elektrischen Feldes (vertikal/horizontal).
	Modulation	Die Modulation hat einen Einfluss darauf, wie robust das Signal gegenüber Störungen ist.
Raum	Position	Das Empfangsverhalten wird maßgeblich durch die Positionen von Sender, Empfänger und Störer und die daraus resultierenden Abstände bestimmt.
	Ausrichtung	Mit Hilfe beispielsweise der Antennencharakteristik kann das Signal in bestimmte Richtungen gebündelt werden.

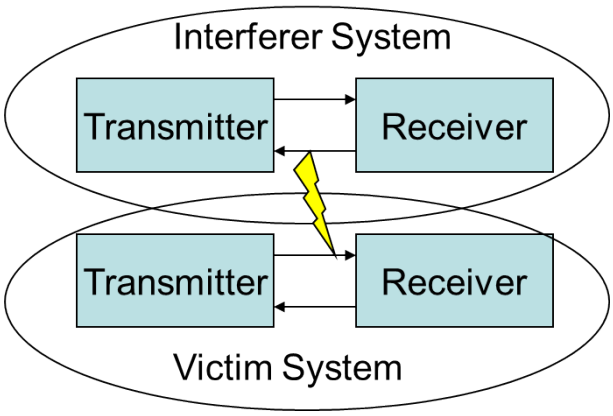

Abb. 3: *Rückwirkungsfreie Beeinflussung eines Funksystems*

Dabei wird nur das Opfersystem von dem Interferersystem gestört. Das Opfersystem beeinflusst dagegen das Interferersystem in keinster Weise. Der rückwirkungsfreie Ansatz eignet sich hervorragend zur Untersuchung, inwieweit sich ein störendes Funksystem auf den Clear Channel Assessment- (CCA-) bzw. CSMA/CA-Mechanismus des betrachteten Systems auswirkt. Beim CCA wird anhand des Energieniveaus überprüft, ob ein zur Übertragung vorgesehener Frequenzkanal frei ist oder bereits von einem anderen Teilnehmer verwendet wird. Das Zeitverhalten des Opfersystems kann sich durch CCA und CSMA/CA dramatisch verschlechtern. Im zweiten Schritt wird ein Ansatz mit Wechselwirkung genutzt. Dabei stören sich das Opfer- und das Interferersystem gegenseitig ([32]).

2.2 Kenngrößen

2.2.1 Kenngrößen der Nachrichtentechnik

Die Bewertung der Koexistenz erfolgt meist mit Kenngrößen der Nachrichtentechnik und für die Anforderungen der Heim- und Bürokommunikation. So werden die Koexistenzeinflüsse anhand der Kenngrößen Paketfehlerrate (Packet Error Rate, PER), Bitfehlerrate (Bit Error Rate, BER) und Datendurchsatz bewertet ([12], [76], [21], [11]).

Zur Betrachtung von Funklösungen auf der Ebene des Physical Layers werden unabhängig vom Vorhandensein anderer Systeme das Level der Signalleistung S und das Rauschniveau N (Signal-to-Noise-Ratio, SNR) beurteilt. Bei der Übertragung im Hochfrequenzbereich und unter Verwendung der Phasen- oder Frequenzmodulation sind Signal- und Trägerleistung nicht voneinander zu unterscheiden. Aus diesem Grund existiert auch die Kenngröße des Träger-Rausch-Verhältnisses (Carrier-to-Noise-Ratio, CNR). CNR wird zur Bewertung des Nutzsignals auf dem Funkmedium verwendet, also zwischen den Antennen von Sender und Empfänger. SNR dagegen beschreibt das Signal im Basisband, d. h. vor der Modulation oder nach der Demodulation. Da nicht nur das Rausch-, sondern auch das Interferenzniveau, welches beispielsweise durch parallele Funksysteme angehoben wird, einen Einfluss auf die Qualität des Nutzsignals hat, wird häufig auf das

Träger-Interferenz-Verhältnis zurückgegriffen (Carrier-to-Interference-Ratio, CIR). Auch der Bezug der Trägerleistung C sowohl zum Interferenz- I als auch zum Rauschniveau N ist möglich (Carrier-to-Interference-Noise-Ratio, CINR). Es gilt:

$$CINR = \frac{C}{I+N} \tag{1}$$

In der digitalen Kommunikation wird mit E_b/N_0 eine normierte Form des Signal-Rausch-Verhältnisses verwendet. Dabei meint E_b die Bitenergie, welche der Signalleistung S bezogen auf die Dauer eines Bits T_b entspricht. N_0 beschreibt die spektrale Rauschleistungsdichte. Sie kann auch mit Hilfe des Quotienten aus Rauschleistung N und Bandbreite W ausgedrückt werden. Wenn R_b als Bitrate eingeführt wird, so ist T_b identisch zu $1/R_b$. Es gilt:

$$\frac{E_b}{N_0} = \frac{S \cdot T_b}{N/W} = \frac{S/R_b}{N/W} \tag{2}$$

Nach Umstellen ergibt sich:

$$\frac{E_b}{N_0} = \frac{S}{N}\left(\frac{W}{R_b}\right) \tag{3}$$

Zur Charakterisierung von Funksystemen wird in der Regel eine monoton fallende Kurve genutzt, welche die BER in Abhängigkeit von E_b/N_0 abbildet (siehe Abb. 4). Auf der Ebene des Physical Layers ist dies die wichtigste Performancemetrik. Die Form der Kurve ist abhängig von der ausgewählten Modulation und dem Empfängerdesign. An dieser Stelle sei angemerkt, dass es sich bei der BER um einen theoretischen Messwert handelt. In einem realen System ist die Messung der BER unmöglich, da der Receiver das gesendete Signal nur schätzen kann. Für eine BER-Messung würde er noch die Kopie des tatsächlich gesendeten Signals benötigen. Dagegen ist innerhalb einer Simulation der Vergleich zwischen der Kopie des gesendeten Signals und dem empfangenen Signal einfach möglich ([32]).

In der Literatur wird vorwiegend die gegenseitige Beeinträchtigung zwischen den Systemen WLAN, Bluetooth und IEEE 802.15.4 untersucht. Dabei nehmen die Empfänger Störer nur durch ein erhöhtes Energielevel wahr. Auf ein IEEE 802.15.4 Netzwerk wirkt ein WLAN-System wie ein breitbandiger Additive White Gaussian Noise (AWGN)-Störer, der das Rauschniveau am Empfänger erhöht. Ein Netzwerk, welches Bluetooth als Funktechnologie verwendet, wirkt dagegen als schmalbandiger Störer. Im Endeffekt stellen beide Störarten eine Form von Interferenz dar. Die durch diese Interferenz erhöhte Bitfehlerrate ergibt sich dann aus dem Signal-Interferenz-Verhältnis (SIR) und dem Abstand der Mittenfrequenzen ([76], [78]).
Die BER ist nur im Physical Layer von Interesse. Sie dient als Bewertungskriterium, um die Funktionsfähigkeit eines Modulationsverfahrens unter konstanten und festgelegten Einflüssen beurteilen zu können. Da bei dieser Kenngröße der Nachrichtentechnik das Augenmerk auf die niedrigste Schicht des ISO/OSI-Referenzmodells gelegt wird, sollte in Modellen zur Koexistenzbetrachtung von industriellen Funksystemen statt der BER die

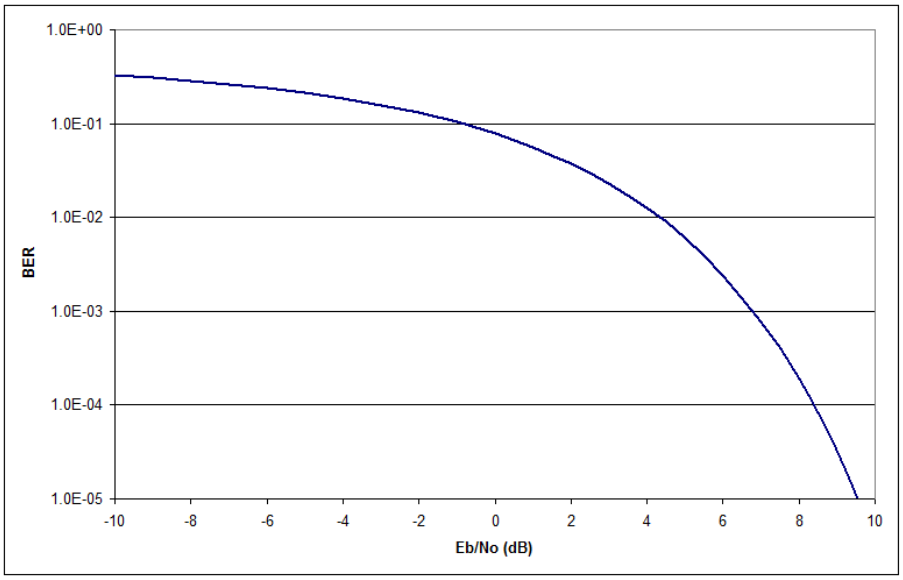

Abb. 4: *BER in Abhängigkeit von der normalisierten SNR am Beispiel von BPSK*

PER Berücksichtigung finden. Die PER beschreibt die Wahrscheinlichkeit, mit der ein Paket bei der Übertragung über das Funkmedium verloren geht. Diese Angabe, ob ein Paket erfolgreich übertragen wurde oder eine Wiederholung notwendig wird, ist für die Modellierung des Zeit- und Fehlerverhaltens ausreichend. Die Schätzung der PER anhand der BER ist mit folgender Formel möglich:

$$PER = 1 - (1 - BER)^N \tag{4}$$

wobei N die Länge eines Paketes in Bits kennzeichnet. Bei der Formel handelt es sich deshalb um eine Schätzung, da von einer Gleichverteilung der Bitfehler ausgegangen wird. Die Schätzung der PER anhand der BER mittels Formel 4 ist nicht zulässig, wenn eine Forward Error Correction eingesetzt wird. In diesem Fall sollte die PER separat und unabhängig von der BER beschrieben werden, beispielsweise mit einem der folgenden Ansätze.

Als einfachste Methode zur Modellierung der PER wird angenommen, dass es sich bei der PER um eine Zufallsvariable handelt, die sowohl unabhängig als auch identisch verteilt ist. Das heißt, dass zwei aufeinanderfolgende Pakete unabhängig voneinander fehlerbehaftet sein können und dass für jedes der Pakete die gleiche Verteilung gilt. Es wird also davon ausgegangen, dass die PER einer Bernoulli- beziehungsweise einer geometrischen Verteilung genügt ([73]). Die Übertragung eines Paketes entspricht mit dieser Annahme dem Wurf einer Münze, nur dass die Wahrscheinlichkeit für Kopf oder Zahl nicht gleich groß ist. Es wird also so lange geworfen, bis das erste Mal Kopf auftaucht. Im Falle der Paketübertragung bedeutet das, dass so lange ein Paket wiederholt wird, bis der Receiver es korrekt empfangen hat.

In einem anderen Ansatz wird das Fehlerverhalten mit einem Markov-Modell, wie zum Beispiel dem allgemein bekannten Gilbert-Elliot-Modell, beschrieben. Beim Gilbert-Elliot-

Modell handelt es sich um eine diskrete Markov-Kette, die aus zwei Zuständen besteht: einen Good- und einen Bad-State. Die Zustandsübergänge erfolgen zufällig mit einer unterschiedlich hohen Wahrscheinlichkeit. Innerhalb der Zustände sind unterschiedliche Werte für die PER angegeben ([28], [37]). In Abb. 5 ist das allgemeine Gilbert-Elliot-Modell dargestellt. Auf der rechten Seite der Abbildung ist das Modell beispielhaft mit Parametern für WLAN belegt. Beim Gilbert-Elliot-Modell ist es wichtig, das Zeitraster zu definieren, wann auf einen möglichen Zustandsübergang geprüft wird. In der Regel werden entweder Zeitslots definiert oder es wird die Symbol- bzw. Bitdauer als Zeitbasis verwendet.

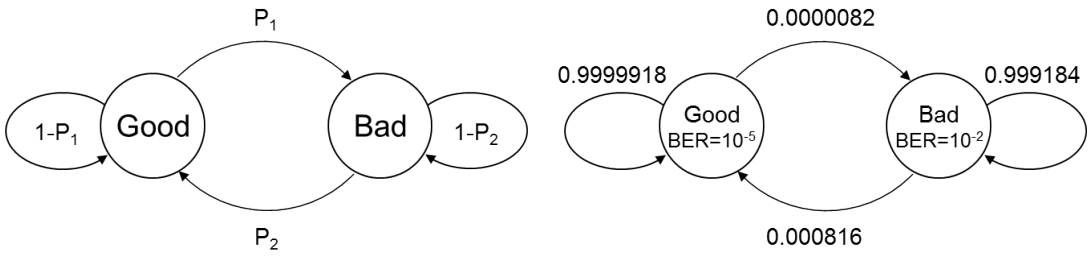

Abb. 5: *Allgemeines Gilbert-Elliot-Modell (links) und in parametrisierter Form für WLAN (rechts) nach [28]*

Eine andere Art der Modellierung basiert auf der Verwendung einer empirischen Verteilung. Eine weitaus komplexere, aber auch genauere Methode ist dagegen die Nutzung eines Error Exponenten ([47]).

2.2.2 Kenngrößen der industriellen Automation

Im Heim- und Bürobereich spielt vor allem der Datendurchsatz eine wichtige Rolle. Dieser gibt an, wie viele Nutzdatenbytes oder Nutzdatenbits pro Zeiteinheit übertragen werden können ([4]). Der Anwender fordert, dass eine Datei, zum Beispiel ein Foto, möglichst schnell von einem auf ein anderes Gerät übertragen wird. Da allerdings in der Automatisierungstechnik gegenüber dem Bild- und Videotransfer die Übertragung von Sensor- und Aktorwerten, die selten die Marke von vier Oktetten überschreiten, dominiert, spielt der Datendurchsatz keine wesentliche Rolle. Vielmehr wird Wert darauf gelegt, wie sicher und zuverlässig sowie in welcher Zeit die Daten übertragen werden. Dazu wurden in der VDI/VDE-Richtline 2185 ([4]) die Kenngrößen Übertragungszeit, Aktualisierungszeit, Antwortzeit und Paketverlustrate definiert, welche angeben, wie gut die Anforderungen der industriellen Automatisierungstechnik von einem Kommunikationssystem erfüllt werden können. Die Kommunikationsschnittstelle (Communication Interface), welche ebenfalls dem Endanwender in einem konkreten Einsatzfall zur Verfügung stehen würde, kann keiner Schicht des ISO/OSI-Referenzmodells zugeordnet werden. Die Ursache liegt in der Vielfältigkeit dieser Schnittstelle begründet. So kann es sich um eine protokollbehaftete oder protokollfreie Schnittstelle handeln. Weiterhin ist es möglich, dass die Schnittstelle softwarebasiert ist oder sich auf ein digitales Signal beschränkt.

- **Übertragungszeit (Transmission Time):** Die Übertragungszeit (siehe Abb. 6) wird nach dem Producer-Consumer-Model ermittelt und ist der Zeitabschnitt vom Beginn der Übergabe des ersten Nutzdatenbytes eines Pakets an die Kommunikationsschnittstelle eines Producers bis zur Übergabe des letzten Nutzdatenbytes desselben Pakets von der Kommunikationsschnittstelle eines Consumers. Zur Übertragung des Paketes kann es erforderlich sein, mehrere Telegramme zwischen den Kommunikationsmodulen von Producer und Consumer zu übertragen. Die Kommunikation zwischen Producer und Consumer kann direkt, über eine Infrastrukturkomponente (z. B. Basisstation) oder über weitere Netzwerkknoten (z. B. bei Sensornetzwerken) erfolgen.

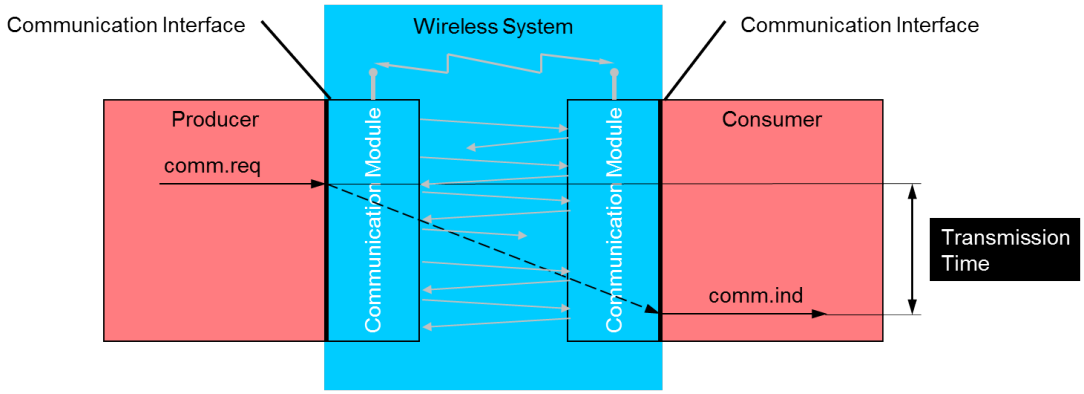

Abb. 6: *Kenngröße „Übertragungszeit“*

- **Aktualisierungszeit (Update Time):** Die Aktualisierungszeit (siehe Abb. 7) wird nach dem Producer-Consumer-Model ermittelt. Sie ist der Zeitabschnitt von der Übergabe des letzten Nutzdatenbyte des Pakets eines Producers von der Kommunikationsschnittstelle eines Consumers an die Anwendung bis zur Übergabe des letzten Nutzdatenbyte des folgenden Pakets vom gleichen Producer.

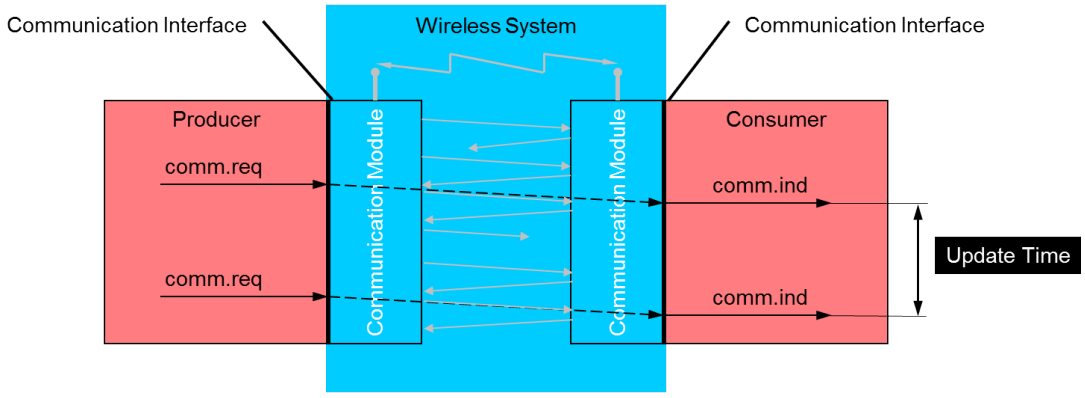

Abb. 7: *Kenngröße „Aktualisierungszeit“*

- **Antwortzeit (Response Time):** Die Anwortzeit wird nach dem Client-Server-Model ermittelt und ist der Zeitabschnitt vom Beginn der Initiierung eines Requests

an der Kommunikationsschnittstelle eines Clients bis zur Identifizierung der Confirmation an der Kommunikationsschnittstelle desselben Clients, die dem Request zugeordnet werden kann. Das heißt, die Antwortzeit setzt sich aus mindestens einer Übertragungszeit zwischen Client und Server und einer Übertragungszeit zwischen Server und Client zusammen. Hinzu kommt die Bearbeitungszeit im Server. Die Kommunikation zwischen Client und Server kann direkt, über eine Infrastrukturkomponente (z. B. Basisstation) oder über weitere Netzwerkknoten (z. B. bei Sensornetzwerken) erfolgen.

- **Paketverlustrate (Packet Loss Rate):** Die Paketverlustrate (R_{PL}) wird nach dem Producer-Consumer-Model ermittelt (siehe Abb. 8). Sie drückt aus, wie viele der von der Anwendung im Producer an die Kommunikationsschnittstelle übergebenen Pakete im Consumer von der Kommunikationsschnittstelle an die Anwendung übergeben werden. Die Paketverlustrate wird wie folgt ermittelt:

$$R_{PL} = \frac{P_{tx} - P_{rx}}{P_{tx}} \tag{5}$$

Dabei ist P_{tx} die Anzahl der gesendeten Pakete und P_{rx} die Anzahl der empfangenen Pakete.

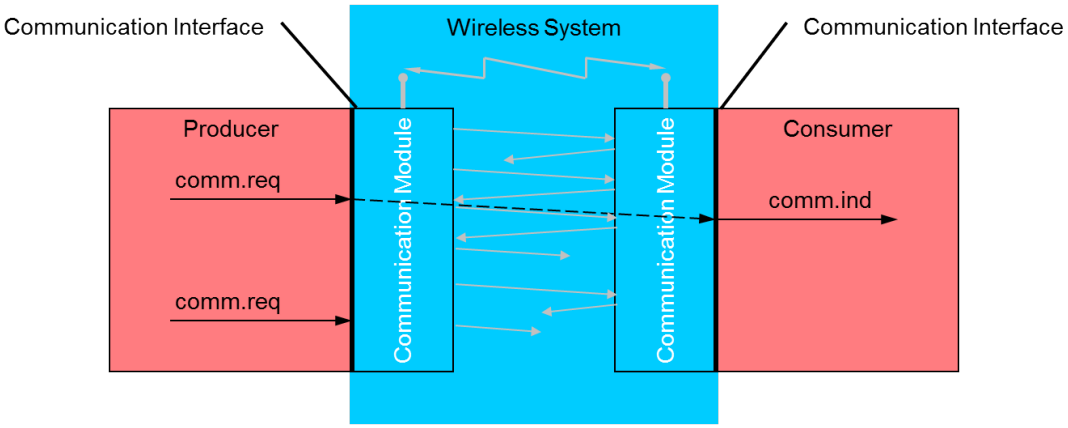

Abb. 8: *Kenngröße „Paketverlustrate“*

Die in [105] vorgestellte Aufteilung der Gesamtübertragungszeit in einzelne Zeitkomponenten wurde so formuliert, dass sie Gültigkeit für sämtliche Funktechnologien besitzt. Dazu sind in Tabelle 2 die relevanten Parameter aufgelistet, die den Wert einer Übertragungszeit mitbestimmen.

Tab. 2: *Parameter, von denen der Übertragungszeitwert abhängt*

Parameter	**Abhängigkeit**		**Anmerkung**
Laufzeit Anwendungsschnittstelle	Funkmodul n	T_{ai}	Die Übertragung zwischen Anwendungsmodul und Kommunikationsmodul kann den Übertragungszeitwert bedeutend beeinflussen. Der Maximalwert hängt von der Art der Schnittstelle, deren Einstellung und der Implementierung ab.
Laufzeit der Funkmodulimplementierung	Funkmodul n	T_i	Die Implementierung des Funkmoduls kann den Übertragungszeitwert bedeutend beeinflussen. Der Maximalwert hängt von der Technologie und deren Einstellungen sowie von der Implementierung ab.
Nutzdatenlänge	Verbindung i	L_{ud}	Die Nutzdatenlänge ist auf die Daten bezogen, die von der Automatisierungsanwendung generiert bzw. konsumiert werden. Ein Worst-Case-Szenarium berücksichtigt das längste Nutzdatenpaket, im vorliegenden Fall bezogen auf die zyklisch übertragenden Prozessdaten.
Datenübertragungsrate	Verbindung i	Bd_{ud}	Die Datenübertragungsrate bezieht sich auf die Übertragung der Nutzdaten bei der Funkübertragung, die z. B. bei WLAN einen anderen Wert haben kann als der Telegramm-Header. In einigen Fällen wird eine Symbolrate angegeben. In diesen Fällen kann ein Symbol aus mehreren Bits bestehen.

Fortsetzung auf der nächsten Seite

Tab. 2: *Parameter, von denen der Übertragungszeitwert abhängt – Fortsetzung*

Parameter	**Abhängigkeit**		**Anmerkung**
Technologie-konstante	Übertragung j	T_{tc}	Die Technologiekonstante beinhaltet alle technologieabhängigen Protokoll-Overheads einer Übertragung, wie feste Pausen- oder Wartezeiten oder die Zeiten zur Übertragung von Telegrammrahmen.
Technologie-variable	Übertragung j	T_{tv}	Die Technologievariable beinhaltet alle technologieabhängigen Protokoll-Overheads, die bei jeder Übertragung einen anderen Wert haben können. Dazu gehören die Wartezeit auf einen freien Kanal oder die Back-off-Zeit. Bei einigen Technologien ist ein Acknowledgement erforderlich, um die Übertragung abzuschließen.
Anzahl der Wiederholungen	Übertragung j, Protokollschicht	N_r	Wenn eine Übertragung gestört wurde, wird sie üblicherweise wiederholt. Das kann in verschiedenen Protokollschichten unterschiedlich häufig geschehen.
Übertragungs-Deadline	Verbindung i	DL	In einigen Fällen wird eine Übertragung abgebrochen, wenn eine Frist überschritten ist.
Zeitfenster für andere Verbindungen	Netzwerk k	T_{ac}	Wenn mehrere Verbindungen im gleichen Übertragungssystem bestehen, ist für diese ein Zeitfenster vorzusehen.
Globaler Zeitrahmen	Netzwerk k	T_{GTS}	In Systemen mit Time Division Multiple Access (TDMA) kann die maximale Übertragungszeit mithilfe des globalen Zeitrahmens ermittelt werden.

Unter Berücksichtigung der in Tabelle 2 aufgeführten Parameter kann die Übertragungszeit je nach Technologie auf verschiedenen Wegen ermittelt werden. In Abbildung 9 sind

die einzelnen Zeitsegmente der Übertragungszeit entsprechend den in Formel (6) beschriebenen Abhängigkeiten dargestellt. Den Maximalwert erhält man, indem für jedes Zeitsegment der Maximalwert eingesetzt wird.

$$T_{TT} = f(T_{ai}(prod), T_i(prod), T_{ac}, N_r, T_{tc}, T_{tv}, L_{ud}, Bd_{ud}, T_i(con), T_{ai}(con)) \tag{6}$$

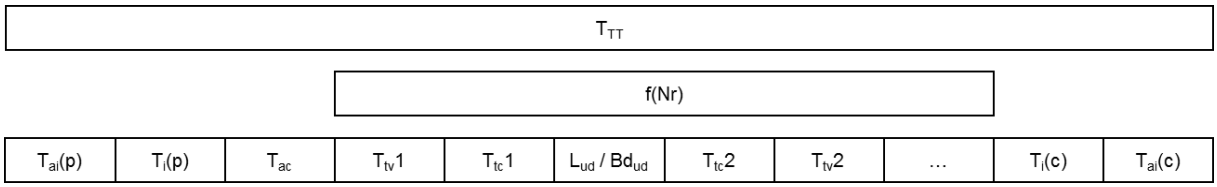

Abb. 9: *Zeitsegmente der Übertragungszeit*

Bei Funklösungen, bei denen eine Übertragungs-Deadline konfiguriert werden kann, vereinfacht sich die Bestimmung des Maximalwertes der Übertragungszeit, wie in Formel (7) und Abbildung 10 dargestellt. Spätestens nach Ablauf der Deadline liegen die Daten vor oder das Paket zählt als verloren.

$$T_{TTmax} = T_{ai}(prod) + T_i(prod) + DL + T_i(con) + T_{ai}(con) \tag{7}$$

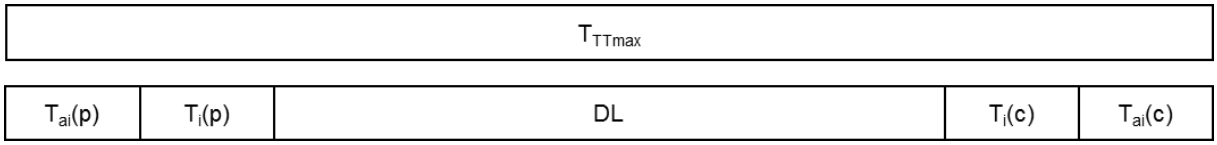

Abb. 10: *Vereinfachte Ermittlung der maximalen Übertragungszeit*

Eine ähnliche Herangehensweise ist bei TDMA-Systemen möglich, bei denen jeder Verbindung ein Zeitschlitz zugeteilt wird. Es ist zu berücksichtigen, dass innerhalb eines Zeitschlitzes mehrere Wiederholungen möglich sein können. Der Maximalwert der Übertragungszeit wird im Wesentlichen durch die Anzahl an Wiederholungen und die dafür genutzten Zeitschlitze bestimmt (siehe Formel (8) und Abbildung 11).

$$T_{TTmax} = f(T_{ai}(prod), T_i(prod), N_r, T_{GTS}, T_i(con), T_{ai}(con)) \tag{8}$$

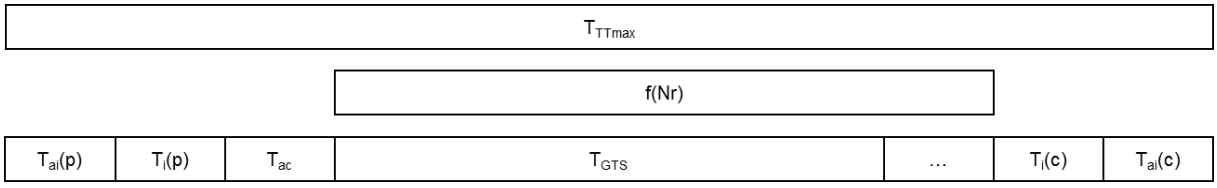

Abb. 11: *Ermittlung der maximalen Übertragungszeit bei TDMA-Systemen*

Bis auf die Datenübertragungsrate, die Übertragungs-Deadline und den globalen Zeitrahmen handelt es sich bei allen anderen Parametern um Zufallsvariablen, für welche die Maximalwerte für den störungsfreien Betrieb zu ermitteln bzw. festzulegen sind.

2.2.3 Ansatz zur Bestimmung der Kenngrößen der industriellen Automation

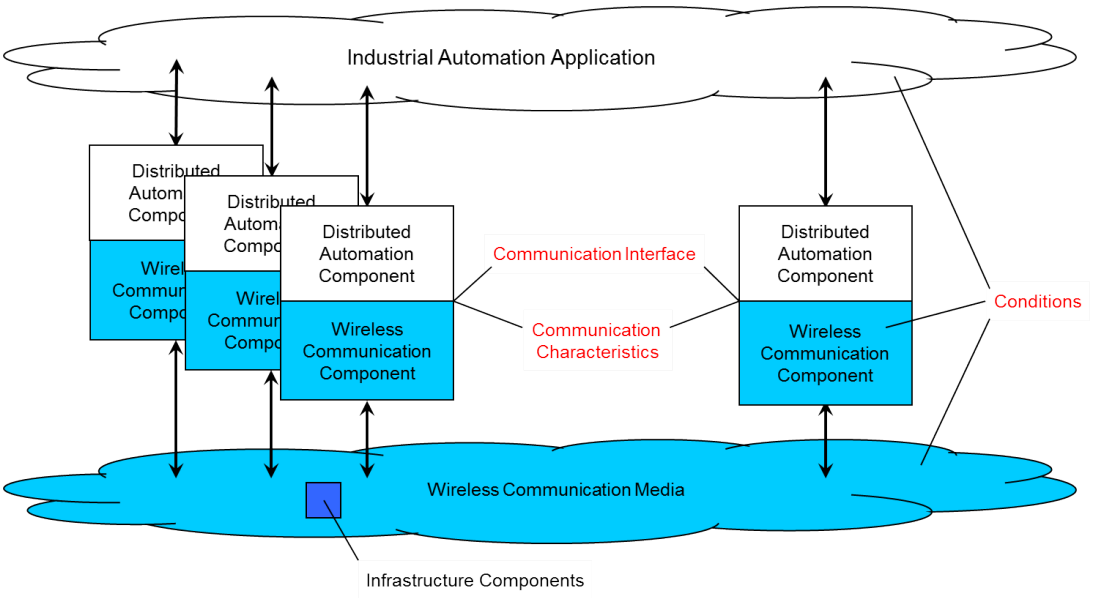

Abb. 12: *Architekturmodell des Betrachtungsraums*

Die am Institut für Automation und Kommunikation e. V. Magdeburg (ifak) entwickelte Methode zur einheitlichen Beurteilung und Bewertung, ob sich eine Funkkomponente (Wireless Communication Component) für einen bestimmten Anwendungsfall eignet ([68], [67]), stützt sich auf die Einführung eines abstrakten Architekturmodells (siehe Abb. 12). Dieses besteht aus einer automatisierungstechnischen Anlage (Industrial Automation Application) und den zur Umsetzung der Anwendung notwendigen verteilten Automatisierungskomponenten (Distributed Automation Components). In der Regel bestimmt die Automatisierungsanwendung, also eine Fertigungsstrecke oder ein verfahrenstechnischer Prozess, die Anforderungen an das Kommunikationssystem (siehe Abschnitt 2.2.4). Automatisierungskomponenten sind Sensoren und Aktoren, die Prozessdaten zur Übertragung generieren beziehungsweise empfangen. Die Übertragung wird von Funkkomponenten (Wireless Communication Component) realisiert, welche direkt mit den Automatisierungskomponenten verbunden sind. Die Funkkomponenten verwenden ein und dasselbe drahtlose Medium (Wireless Communication Media) zur Kommunikation. Zur drahtlosen Übertragung können Infrastrukturkomponenten (Infrastructure Components) wie Access Points oder Basisstationen von Nöten sein.
In der zu Grunde liegenden Abstraktionsebene sind drei wesentliche Bestandteile entscheidend: die verwendete Kommunikationsschnittstelle (Communication Interface) zwischen Automatisierungs- und Funkkomponente, die herrschenden Einflussgrößen (Conditions) und die zu ermittelnden Kenngrößen (Communication Characteristics).
Die verwendeten Kenngrößen wurden bereits in Abschnitt 2.2.2 eingeführt. Die Kommunikationsschnittstelle zwischen Funk- und Automatisierungskomponente ist drahtgebunden und verwendet Standards wie beispielsweise Ethernet, RS232, RS485, UART oder SPI. Einflussgrößen haben eine unmittelbare Wirkung auf die Kenngrößen. Diese können in

anwendungsbezogene, geräte- und systemspezifische sowie umgebungsbezogene Einflussgrößen unterschieden werden.
Anwendungsbezogene Einflussgrößen werden durch die Kommunikationsanforderungen aus Anwendungssicht bestimmt. Dazu gehören die zeitlichen Randbedingungen der Übertragung und die Empfangsbedingungen durch Übertragungsentfernung, Knotendichte und Mediennutzung. Geräte- und systemspezifische Einflussgrößen werden durch die Implementierung der Funksysteme festgelegt. Dazu gehören auch konfigurierbare Parameter wie die Bitrate. Auch die Umgebungsbedingungen beeinflussen die Kenngrößen einer Funklösung. Diese sind kaum oder gar nicht planbar. Ein Beispiel sind die Ausbreitungsbedingungen der Funkwellen.

Zur Ermittlung der Kenngrößen und der damit verbundenen Festellung, ob ein Funkkommunikationssytem bestimmte Anwendungsanforderungen erfüllt und sich dementsprechend eignet, müssen die Automatisierungskomponenten in Abbildung 12 durch Testkomponenten ausgetauscht werden. In [31] wurde als Testkomponente ein Multifunktionsinterface (Multiface) entwickelt, welches die kabelgebundene Anbindung an das zu testende Gerät (Device-Under-Test, DUT) übernimmt. Ein Multiface generiert die zu übertragenden Testpakete und misst deren Kenngrößen. Zur Erfassung zeitbehafteter Kenngrößen ist eine Synchronisierung der Multifaces notwendig. Diese wird mit Hilfe von Lichtwellenleiterverbindungen zwischen den Multifaces realisiert. Die beschriebene Messmethodik eines Funk-Transfer-Testers (FTT) für industrielle Funklösungen wurde in [103] erarbeitet.

2.2.4 Anforderungen der Anwendung

Jede Automatisierungsanwendung stellt an ein Kommunikationssystem unterschiedliche Anforderungen, die für einen fehlerfreien Betrieb eingehalten werden müssen. Dabei ist es von der einzelnen Anwendung abhängig, um welche Anforderungen es sich handelt und in welchem Umfang diese zu erfüllen sind. Im Folgenden sollen verschiedene Anwendungsanforderungen definiert und erläutert werden.

- **Echtzeitfähigkeit:** Nach [106] ist ein System echtzeitfähig, wenn es „unter allen Betriebsbedingungen rechtzeitig auf alle auftretenden Ereignisse reagieren“ kann. In diesem Zusammenhang meint „rechtzeitig“, dass das Abtasttheorem nach Shannon nicht verletzt werden darf. Für ein Kommunikationssystem heißt das, dass „die qualitativen und zeitlichen Forderungen an den Datenaustausch der Komponenten einer konkreten Anwendung“ erfüllt werden. „Dies kann bedeuten, dass für die Zeit zwischen zwei Ereignissen eine bestimmte Maximalzeit, die Deadline, nicht überschritten werden darf [...]“.

- **Verfügbarkeit:** Die Verfügbarkeit ist in [5] definiert als „Fähigkeit einer Einheit, zu einem gegebenen Zeitpunkt oder während eines gegebenen Zeitintervalls eine

geforderte Funktion unter gegebenen Bedingungen erfüllen zu können, vorausgesetzt, dass die erforderlichen äußeren Hilfsmittel bereitgestellt sind.“ Angewandt auf die Funkkommunikation ist es die Fähigkeit einer Funklösung, Nutzdaten von einem Producer zu einem Consumer zu übertragen. Zur geforderten Funktion gehört auch, dass die Daten durch die Funkübertragung nicht verändert werden, also fehlerfrei den Consumer erreichen. Und schließlich gehört zur geforderten Funktion, dass die Daten innerhalb einer bestimmten Zeit oder zu einem geforderten Zeitpunkt den Consumer erreichen.

- **Zuverlässigkeit:** In [5] ist die Zuverlässigkeit definiert als „zusammenfassender Ausdruck zur Beschreibung der Verfügbarkeit und ihrer Einflussfaktoren Funktionsfähigkeit, Instandhaltbarkeit und Instandhaltungsbereitschaft.“ Die Zuverlässigkeit ist also selbst keine Kenngröße, die durch Messung ermittelt werden kann.

- **Robustheit:** Robustheit ist die Wahrscheinlichkeit, mit der ein System auch unter ungünstigen Störeinflüssen im Rahmen der spezifizierten Eigenschaften funktioniert. Diese wahrscheinlichkeitsbasierte Definition spiegelt die begriffliche Nähe von Robustheit und Zuverlässigkeit wider. Die Robustheit setzt erst bei Überschreitung des Rahmens des normalen Betriebs ein. Zudem erlaubt es der Robustheitsbegriff, die betrachteten Störeinflüsse und Eigenschaften explizit anzugeben, wodurch festgelegt wird, wogegen und bezüglich welcher Eigenschaften ein System robust sein soll ([66]).

- **Determinismus:** „Determinismus“ hat seinen Ursprung im lateinischen Wort *determinare*, welches „abgrenzen“ oder „bestimmen“ bedeutet. Nach [49] ist das Verhalten eines deterministischen Systems eindeutig durch die Systemgleichungen bestimmt und bei unendlicher Genauigkeit theoretisch exakt vorhersagbar (Laplace'scher Dämon). Damit gilt für ein drahtloses Kommunikationssystem, dass alle Zustände und Kenngrößen der Datenübertragung im Voraus berechenbar sind ([106]).

2.3 Formale Modellansätze für Koexistenzuntersuchungen

2.3.1 Ansatz eines analytischen Modells

In Bezug auf Koexistenzbetrachtungen besteht die Möglichkeit, ein analytisches Modell einzusetzen. Dabei wird die PER in Abhängigkeit von der BER und der Kollisionszeit berechnet. Es fehlt allerdings die Berücksichtigung der Forward Error Correction ([76], [77], [75]). Abb. 13 zeigt beispielhaft die Kollisionen zwischen Bluetooth und IEEE 802.15.4 Paketen. Bluetooth nutzt TDMA. Bei diesem Zugriffsverfahren bekommt jeder Teilnehmer Zeitschlitze (Slots) zugewiesen, in denen dieser den Kanal ohne vorherige Überprüfung zur Übertragung von Paketen belegen darf. Weiterhin verwendet Bluetooth ein Frequenzsprungverfahren. Nach jedem gesendeten Paket wechselt das System den zur Übertragung verwendeten Frequenzbereich. Bei IEEE 802.15.4 kommt CSMA/CA zum Einsatz. Daten

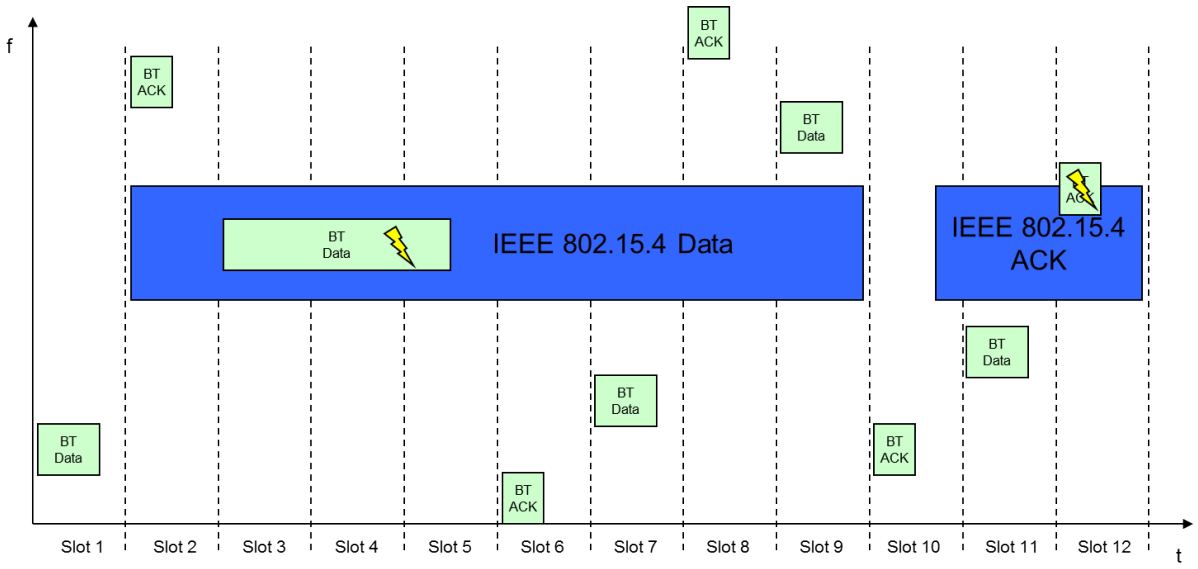

Abb. 13: *Koexistenzbetrachtung zwischen Bluetooth (BT) und IEEE 802.15.4*

werden bei diesem Zugriffsverfahren erst übertragen, nachdem das Medium als unbelegt erkannt wurde. Kollisionen treten zwischen beiden Systemen bei gleichzeitiger Nutzung desselben Frequenzbereiches auf.

2.3.2 Modellierung mit Petri-Netzen

Petri-Netze werden im Bereich der Kommunikation auf zwei Anwendungsgebieten erfolgreich eingesetzt: der Performanceevaluierung und der Validierung von Übertragungsprotokollen ([63]). [93] führt die Überprüfung auf Fehlerfreiheit bei einem existierenden Routingprotokoll für mobile Ad-hoc-Netzwerke ein. Ein weiteres Beispiel liefert [34], welches die Spezifikation vom WAP Class 2 Wireless Transaction Protocol mit Petri-Netzen beschreibt. Performanceanalysen werden in [99] für drahtlose Ad-hoc-Netzwerke, in [45] und [94] für IEEE 802.11 DCF und in [18] für drahtlose Internetzugangssysteme basierend auf der GSM/GPRS-Technologie durchgeführt. [56] diskutiert drei Verfahren für eine zusammengesetzte Performance- und Verfügbarkeitsanalyse von Warteschlangensystemen in drahtlosen Kommunikationsnetzwerken. Petri-Netze werden außerdem für die Modellierung von Wiederherstellungsstrategien bei der Verwendung von RF-Kanälen ([55]) und bei der Beschreibung von Controllerkonzepten für Cognitive Radios ([89]) verwendet. Sie haben sich weiterhin bei der Modellierung von Herstellungsprozessen, in denen unabhängig arbeitende Maschinen gegenseitig miteinander verknüpft werden, bewährt. Verschiedene Elemente der industriellen Automation sind ebenfalls Gegenstand der Modellierung mit Petri-Netzen, beispielsweise Feldbussysteme ([48], [52]).

[30] und [38] haben mit WLAN und Bluetooth nachgewiesen, dass sich Petri-Netze auch für die Modellierung von Funksystemen eignen. Im Detail wird das Verhalten eines Bluetooth-Scatternets und eines WLAN Ad-Hoc-Netzes nachgebildet und untersucht. Zur Modellierung des Übertragungsmediums wurde entweder ein idealer Kanal verwendet oder es

wurde in Abhängigkeit von der Entfernung und Interferenz eine Packet Loss Probability genutzt. Diese Wahrscheinlichkeit gibt an, ob ein Paket empfangen wird. Auf ankommende Pakete haben die Autoren zusätzlich noch die BER angewendet. In den genannten Quellen waren nur der Throughput und die Wartezeit bis zum Medienzugriff von Interesse. Die Gesamtübertragungszeit spielte keine Rolle.

2.3.3 Warteschlangenmodelle

Wie Petri-Netze werden auch Warteschlangenmodelle zur Performancebewertung von drahtlosen Netzwerken eingesetzt. Ein Beispiel ist die Analyse eines drahtlosen Multihop Ad-hoc-Netzwerks, bei dem die Knoten einen zufälligen Medienzugriff verwenden ([24]). Als Kenngrößen dienen die Ende-zu-Ende-Verzögerung und der maximal erreichbare Durchsatz. Als Ende-zu-Ende-Verzögerung wird dabei die Zeitdauer verstanden, die von der Generierung eines Paketes bis zu dessen Eintreffen am Zielknoten vergeht. Weiterhin werden Warteschlangenmodelle zur Performanceevaluierung der IEEE 802.11 MAC eingesetzt ([60], [64], [97]). Der Vorteil dieser Modellierungsart ist, dass anhand einer Analyse der Netzwerkwarteschlangen analytische Ergebnisse gewonnen werden können. Dadurch werden Simulationen überflüssig.
Warteschlangenmodelle wurden bereits für homogene Koexistenzuntersuchungen eingesetzt ([69]). Dabei wurde beispielsweise die Kollisionswahrscheinlichkeit bei vorhandenen Hidden Nodes ermittelt. Koexistenzuntersuchungen mit unterschiedlichen Funktechnologien sind nicht bekannt.

2.4 Medienzugriffsmechanismen

2.4.1 Einordnung ins ISO/OSI-Referenzmodell

Oft ist der Aufbau von Kommunikationsprotokollen sehr komplex. Bei deren Entwicklung müssen eine Reihe von Anforderungen Berücksichtigung finden, sodass für eine gleichzeitige Lösung der vorliegenden Herausforderungen eine strukturierte Aufgabenteilung notwendig ist. Diese Gliederung führt zu einem hierarchischen Schichtenmodell, welches den Aufbau der Kommunikation zwischen zwei Teilnehmern beschreibt. In vielen Fällen wird das „Open Systems Interconnections“ (OSI) - Referenzmodell verwendet, welches durch das ISO Standardisierungsgremium (ISO/OSI - Referenzmodell) spezifiziert wurde und aus sieben Schichten besteht (siehe Abbildung 14). Jede Schicht übernimmt bestimmte Aufgaben und Funktionen. Eine höhere Schicht kann nur auf die Funktionen der unmittelbar darunterliegenden Schicht zugreifen. Der Austausch einzelner Schichten ist ohne Beeinflussung der anderen Schichten möglich. Bei der Entwicklung eines Protokollstacks ist die Umsetzung aller sieben Schichten nicht zwingend erforderlich. Mit dem standardisierten, geschichteten Ansatz des ISO/OSI-Referenzmodells kann die Komplexität beim Entwurf eines digitalen Kommunikationssystems reduziert werden.
Die erste und niedrigste Schicht ist der Physical Layer. Der Physical Layer ist für die

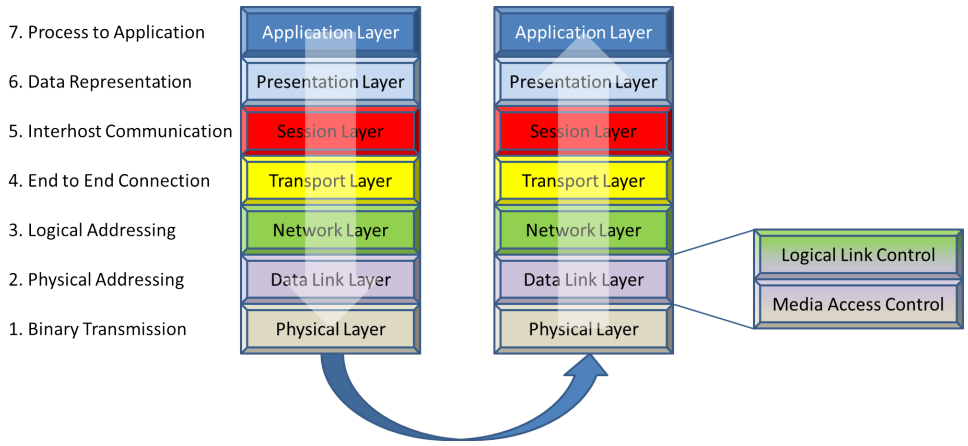

Abb. 14: *ISO/OSI-Referenzmodell*

Umsetzung der Bitfolge, die aus den Nutzinformationen resultiert, in das physikalische Signal verantwortlich und definiert all die Bestandteile, die für die Übertragung über das verwendete Medium notwendig sind. Beispielhafte Aufgaben sind: Aktivierung und Deaktivierung des Funkmoduls, Erfassung der Verbindungsqualität, die Kanalauswahl sowie das physische Senden und Empfangen von Telegrammen. Der Physical Layer ist direkt mit dem Data Link Layer verknüpft, welcher aus MAC Layer und Logical Link Control (LLC) Layer besteht. Dabei ist der MAC Layer der untere Sublayer der beiden. Dieser Sublayer ist für die Koordinierung und Kombination von Datenkanälen unterschiedlicher Herkunft und Frequenz verantwortlich. Weiterhin steuert er die geordnete und kollisionsfreie Nutzung des von mehreren Teilnehmern gemeinsam genutzten Teilnehmers mit Hilfe eines entsprechenden Medienzugriffsmechanismus. Der LLC Layer ist verantwortlich für die Fluss- und Fehlerkontrolle. Im Gegensatz zu den beiden unteren Schichten stellt der Network Layer das erste Mal den Netzwerkaspekt in den Vordergrund. Er ist verantwortlich für das Routing und den Aufbau des Netzwerks. In diesem Teil des ISO/OSI - Referenzmodells wird die Übertragung über mehrere Knoten bis zum eigentlichen Empfänger sichergestellt. Die vierte Schicht ist der Transport Layer, welcher Optimierungsroutinen und andere Quality of Service (QoS)-Methoden für einen effizienten Datenaustausch beinhaltet. Der Session Layer realisiert die Dialogkontrolle zwischen den Anwendungen an beiden Enden der Verbindung. Der Presentation Layer löst Unterschiede zwischen den Darstellungsweisen von Formaten und Daten auf. Zu dieser Schicht gehört auch die mögliche Datenkomprimierung und Verschlüsselung. Der Application Layer ist die höchste Schicht des ISO/OSI - Referenzmodells und definiert die Anwendungsdienste ([79]).

Zur Untersuchung des Koexistenzverhaltens von Funksystemen ist es am sinnvollsten, den Fokus auf den Physical und MAC Layer des ISO/OSI-Referenzmodells zu legen ([62]). Die Modulationsart entscheidet beispielsweise, wie robust das Empfangsverhalten gegenüber Interferenzen ist. Der MAC Layer bestimmt die Regeln, wann eine Übertragung über das geteilte Medium stattfinden kann und beeinflusst dadurch die Kenngrößen wie die Übertragungszeit und den Durchsatz. Auch die Kanalauswahl erfolgt im MAC Layer. Zusätzlich

wird der Application Layer benötigt, welcher die Anforderungen und das Verhalten der Automatisierungsanwendung beschreibt.

2.4.2 Klassifizierung von Medienzugriffsmechanismen

Es existieren eine Reihe von Medienzugriffsmechansismen, die ein unterschiedliches Verhalten in Bezug auf die Koexistenz aufweisen. Sie können in nicht-adaptive, adaptive und kognitive unterteilt werden ([25]).
Nicht-adaptive Medienzugriffsmechanismen sind zum Beispiel Zeit-, Frequenz- und Codemultiplexverfahren (TDMA, FDMA, CDMA). Sie enthalten keinen Mechanismus zur Vermeidung von Interferenzen und stören dadurch andere Medienzugriffsverfahren. Durch ihre synchrone Struktur weisen sie allerdings ein deterministisches Verhalten auf.
Adaptive Mechanismen vermeiden Interferenzen und stellen dadurch eine weitestgehend fehlerfreie Übertragung her. Dies wird erreicht, indem auf den Zustand der Funkumgebung reagiert wird. Ein nichtdeterministisches Verhalten ist die Folge. Adaptive Mechanismen unterscheiden sich in dem Zeitpunkt, in welchem der Zustand des Mediums berücksichtigt wird. Listen-Before-Talk (LBT) tastet das Medium vor der Übertragung ab. Dagegen bewertet Listen-After-Talk (LAT) den Status vorheriger Übertragungen. Adaptivität existiert auch für Frequenzhopper (z. B. AFH). So wird bei der Erkennung von Paketverlusten der dazugehörige Kanal für zukünftige Übertragungen gesperrt.
Kognitive Medienzugriffsmechanismen passen sich der Funkumgebung an, in dem das Medium beobachtet wird und die zukünftige Belegung abgeschätzt wird. Sie können sich über mehrere Parameter flexibel anpassen (z. B. Zeit, Frequenz und Leistung). Kognitive Verfahren sollen nicht tiefergehend betrachtet werden.
Tabelle 3 zeigt eine Auflistung von Medienzugriffsmechanismen und deren Zuordnung zu adaptiven und nicht-adaptiven Verfahren. In Klammern sind zugehörige Beispiele angegeben.

Tab. 3: *Übersicht zu Medienzugriffsmechanismen*

	Nicht-Adaptive	**Adaptive**
Code	CDMA (UMTS, GPS)	
Zeit	TDMA (GSM), TDD (DECT)	ALOHA, CSMA/CA (WLAN)
Frequenz	FHSS (Bluetooth), FDMA (OFDM), FDD (C-Netz)	AFH (Bluetooth V1.2)

2.4.3 Besonderheiten von CSMA/CA-Systemen

Ein weiterer, meist unberücksichtigter Punkt bei Koexistenzuntersuchungen ist der Kanalzugriff von CSMA/CA-Systemen. Darunter fallen Technologien wie WLAN oder IEEE 802.15.4, nicht jedoch Bluetooth. Bei CSMA/CA-Systemen wird vor der Übertragung ei-

nes Paketes mit CCA geprüft, ob der Kanal frei ist. Zur Erfassung anderer Systeme wird dazu das Energielevel im betreffenden Frequenzbereich ermittelt und anschließend beurteilt, ob es sich unter einer festgelegten Grenze befindet. Bei WLAN befindet sich diese bei -76 dB ([19]).

IEEE 802.15.4 und WLAN unterscheiden sich in ihrem CSMA/CA-Verhalten. Während WLAN auch innerhalb der Wartezeit den Zustand des Kanals überprüft, führt IEEE 802.15.4 CCA nur nach Ablauf der zufälligen Verweilzeit aus. Wird in dieser kurzen Phase das Medium als belegt erkannt, generiert der IEEE 802.15.4-Teilnehmer erneut eine zufällige Wartezeit, in der weder ein Paket versendet noch CCA betrieben wird.

In [23] wird der Einfluss auf das CSMA/CA eines gestressten IEEE 802.15.4-Systems durch ein WLAN-, Bluetooth- und ein anderes IEEE 802.15.4-System untersucht. Die darin vorgestellten Ergebnisse zeigen, dass die Mehrzahl der Paketverluste durch das eingeschaltete CCA verursacht wurde. Die Autoren geben daraufhin die Empfehlung, CCA generell zu deaktivieren, was allerdings in verschiedenen Ländern gegen relevante Regulierungen verstoßen kann.

Bei der Untersuchung der Koexistenz zwischen WLAN und IEEE 802.15.4 bietet sich die Unterteilung in verschiedene Koexistenzgebiete an. Im Nahbereich erkennen sich beide Systeme gegenseitig. Im mittleren Bereich nimmt nur das IEEE 802.15.4-System das WLAN wahr. Dies resultiert aus der wesentlich größeren Sendeleistung des WLAN-Systems. Im Fernbereich ist eine gegenseitige Beeinflussung zwischen beiden Systemen ausgeschlossen ([95]).

Ein weiteres Gebiet bei der Koexistenz von Systemen, welche dieselbe Funktechnologie verwenden, ist die Adjacent-Channel-Interferenz. Dabei wird, zum Beispiel in [78] und [20], die gegenseitige Beeinflussung von Systemen untersucht, welche in benachbarten Kanälen arbeiten.

2.4.4 Anforderungen an Medienzugriffsmechanismen aus Gerätesicht

Für Medienzugriffsmechanismen gelten eine Reihe von Anforderungen, die anhand von Parametern beschrieben werden. Dabei hängen die Anforderungen von dem verwendeten Frequenzband ab. Für Geräte mit einer Sendeleistung bis 100 mW, die auf das 2,4 GHz-ISM-Band zugreifen, gilt die EN 300 328 ([85]). Die genauen Werte der Anforderungsparameter haben Einfluss auf das Koexistenzverhalten (siehe [8]). Eine Auswahl an Parametern zeigt die folgende Auflistung.

- **Kanalbelegungszeit (Channel Occupancy Time):** Die Gesamtzeit, die ein Gerät für Übertragungen auf einem gegebenen Kanal zugreifen kann, ohne erneut die

Verfügbarkeit dieses Kanals bewerten zu müssen, wird Kanalbelegungszeit genannt ([85]). Die Kanalbelegungszeit wird auch als Sequenzzeit bezeichnet.

- **Transmission Gap (Sendepause):** Die minimale Transmission Gap ist die Zeit zwischen zwei durch einen Transmitter vorgenommenen Kanalbelegungen ([8]).
- **Duty Cycle:** Der Duty Cycle ist das Verhältnis aus Sendesequenz und einem gegebenen Beobachtungszeitraum für einen verwendeten Funkkanal. Die Weise, wie der Beobachtungszeitraum gewählt wird, beeinflusst den Wert des Duty Cycle ([8]).
- **Maximum Dwell Time:** Die Maximum Dwell Time entspricht der Zeitperiode, in der ein System zu einem bestimmten Kanal zugeordnet ist. Dieser Parameter ist nur für Frequenzhoppingsysteme geeignet ([8]).
- **CCA Time:** Die minimale CCA Time bestimmt, wie lange ein Kanal mindestens auf eine mögliche Belegung überprüft werden muss.
- **Bandbreite:** Die Bandbreite ist der Frequenzbereich, der 99 % der Leistung eines modulierten Trägersignals beinhaltet.
- **Kanalsperrzeit (Unavailable Time):** Die Kanalsperrzeit ist die Zeit, die ein Kanal nicht für Übertragungen genutzt werden darf, nachdem Interferenzen erkannt wurden.

In der Auflistung befinden sich auch Parameter, die nur für spezifische Medienzugriffsmechanismen von Bedeutung sind. Beispiele dafür sind die CCA Time oder Kanalsperrzeit. Es existieren noch weitere solcher Parameter. Diese werden wenn nötig an entsprechender Stelle gesondert erläutert.
Die genannten Anforderungen stellen Kenngrößen dar, die das Verhalten eines Funkgerätes auf dem Medium charakterisieren. Aus Sicht der industriellen Automation wirken diese Anforderungen als gerätebezogene Einflussgrößen (siehe Abschnitt 2.2.3). Gleichzeitig werden durch die EN 300 328 die Grenzen dieser Einflussgrößen vorgegeben und damit das Zeit- und Koexistenzverhalten einer Funklösung bestimmt.

2.5 Bekannte Ergebnisse von Koexistenzuntersuchungen

Nach dem aktuellen Stand der Technik gibt es erste Ansätze einer pragmatischen Frequenzplanung (siehe [98], [53], [43]). Allerdings fehlen noch, gestützt auf systematische Untersuchungen, Informationen über die Charakteristik der involvierten Funklösungen und deren Auswirkungen auf die Koexistenz. Dies gilt insbesondere, wenn Wechselwirkungen zwischen vielen unterschiedlichen Systemen auftreten. Die Literatur liefert mit den Ergebnissen aus Experimenten und Simulationen erste Eindrücke, wann eine gegenseitige Beeinflussung der Systeme auf physikalischer Ebene stattfindet und wann nicht:

- Eine Beeinflussung von Bluetooth durch WLAN findet nicht mehr bei einem Mittenfrequenzoffset größer 11 MHz oder einem SIR größer 6 dB statt ([78]).
- Adjacent-Channel-Interferenzen treten bei WLAN erst bei SIR-Werten kleiner -14 dB auf. Dabei sind die Interferenzen auf Grund des Barker-Codes bei einem Mittenfrequenzoffset von 1 MHz am größten und bei 8 MHz am kleinsten ([78]).
- Bei der Verwendung von Bluetooth können Co-Channel-Interferenzen bei SIR-Werten größer 20 dB ausgeschlossen werden ([78]).
- Die Wirkung von Bluetooth-Interferenzen auf WLAN ist bei SIR-Werten größer 2 dB vernachlässigbar ([78]).
- Eine Beeinflussung von IEEE 802.15.4 auf WLAN ist nur bei einem Mittenfrequenzabstand kleiner 12 MHz feststellbar ([41]).
- Eine Veränderung der BER bei IEEE 802.15.4 durch WLAN ist bei einem Mittenfrequenzoffset größer 7 MHz oder einer Entfernung von 8 m nicht nachweisbar ([76], [65])
- Wenn zwei WLAN-Komponenten weniger als 3 m voneinander entfernt sind, können sie nicht von einem IEEE 802.15.4-Netzwerk beeinflusst werden ([77]).
- Um bei IEEE 802.15.4 eine PER von 10^{-5} zu gewährleisten, ist eine Distanz von 8,65 m zu WLAN und 5,7 m zu Bluetooth notwendig ([77]).

In [102] wurden Untersuchungen zum Thema Koexistenz von drahtlosen Kommunikationssystemen im industriellen Umfeld durchgeführt. Diese bestanden aus zwei Teilen. Im ersten Teil wurden Bluetooth, WLAN, WISA als Vorgänger von WSAN-FA, und ZigBee als gestörte Systeme betrachtet. Ausgewertet wurden dabei der Einfluss der Umgebung, der Einfluss von Störsystemen (Bluetooth, nanoNet, WISA, WLAN, ZigBee), der Einfluss des Abstandes zwischen Störsystem und gestörtem System (1 m, 3 m, 10 m) sowie der Einfluss der Fahrt eines Hallenkrans auf das Zeit- und Fehlerverhalten des untersuchten Funksystems. Im zweiten Teil des Projektes wurden Tests an Bluetooth, WISA, WLAN, MeshScape von Millennial Net und Smart Mesh von Dust Networks durchgeführt. Die zuletzt genannten Systeme sind drahtlose Sensornetzwerke, welche auf der Spezifikation IEEE 802.15.4 basieren. Bei diesem Projektteil wurde der Einfluss der Knotendichte und der Einfluss anderer Frequenznutzersysteme (Bluetooth, WLAN, WISA, IEEE 802.15.4) ausgewertet. Weiterhin wurde der Einfluss des Signal-Stör-Verhältnisses auf das Zeit- und Fehlerverhalten des gestörten Systems untersucht. Dabei wurde festgestellt, dass eine Störung tatsächlich vorliegt, wenn die Störleistung mindestens 10 dB größer als die Sendeleistung des Producers am Ort des Consumers ist.
Die oben aufgeführten Mess- und Simulationsergebnisse könnten den Eindruck entstehen

lassen, dass sich einfache Regeln in Bezug auf die Koexistenz von industriellen Funksystemen ableiten lassen. Die Messungen in [102] haben allerdings bewiesen, dass eine Bildung solcher „Faustformeln“ nicht möglich ist. Aus diesem Grund sind Messungen und Simulationen zur Untersuchung des Koexistenzverhaltens im industriellen Umfeld zwingend erforderlich.

2.6 Tools

Zur Untersuchung von Funkkommunikation und der damit verbundenen Technologien existieren eine Reihe von Tools:

- In der Forschung werden vor allem Netzwerksimulatoren wie OMNeT++, OPNET oder ns2 eingesetzt ([33], [92]). Die ursprünglich zur Untersuchung von Ethernet geschaffenen Simulatoren wurden in den letzten Jahren um Funktechnologien und verschiedene Kanalmodelle erweitert. Für Koexistenzanalysen müssen jedoch zusätzliche Frameworks eingebunden oder erstellt werden. Bei OMNeT++ gibt es beispielsweise das INET- und das Mixim-Framework. Das INET-Framework wurde für die Simulation von TCP/IP und anderen Protokollen des Internets entwickelt. Bei der Untersuchung von drahtlosen Anwendungen lassen die Kanalmodelle bisher nur Funklösungen der gleichen Technologie zu. Die gegenseitige Beeinflussung unterschiedlicher Technologien kann nicht untersucht werden. Dagegen bietet das Mixim-Framework ein Reihe von PHY- und MAC-Layer verschiedener Funktechnologien. Zudem stehen eine Reihe von Kanalmodellen zur Verfügung, sodass Koexistenzbetrachtungen detailliert möglich sind. Die Anbindung höherer Schichten muss durch den Nutzer selbst erfolgen.

- Auch MATLAB/Simulink kann mit der „Signal Processing Toolbox“ zur Simulation der Funkkommunikation verwendet werden ([58]). Dabei erfolgt die Untersuchung aus nachrichtentechnischer Sicht. Der Fokus liegt auf dem Physical Layer. Als Kenngrößen dienen sowohl die PER als auch die BER. Der Übertragungskanal wird in der Regel als AWGN-Kanal modelliert. Ein entscheidender Vorteil von MATLAB/Simulink ist die Erweiterbarkeit. Auch sind bereits viele Modelle verschiedener Funkspezifikationen verfügbar.

- Vom National Institute of Standards and Technology, genauer von der Wireless Communication Technologies Group, ist ein Open Source Tool veröffentlicht worden, welches die Koexistenz zwischen Bluetooth und IEEE 802.11b simulieren kann. Dies geschieht auf der Ebene des Physical Layers. Der User gibt dabei lediglich die Carrier-to-Interference-Ratio und die Carrier-to-Noise-Ratio für den AWGN-Kanal ein ([35]). In den Spezifikationen [12], [14] und [11] der IEEE wird die Koexistenz zwischen verschiedenen Funkstandards ebenfalls anhand von Simulationen, die mit diesem Werkzeug durchgeführt wurden, bewertet. Die Simulationsergebnisse geben

dabei einen ersten Hinweis auf die wechselseitige Beeinflussung von verschiedenen Funkübertragungsverfahren. Als Kenngröße zur Bewertung der Koexistenz wird die PER verwendet. Da sich die in den IEEE-Spezifikationen enthaltenen Simulationsergebnisse lediglich auf die physikalische Schicht der Funkübertragung beziehen, werden nur die Einflüsse, welche bei Paketkollisionen auftreten, betrachtet. Die meisten Funkspezifikationen enthalten jedoch Funktionen zur Vermeidung von Paketkollisionen, welche in den Medienzugriffsverfahren (MAC) (siehe [12], [9]) integriert sind. Diese werden bei diesem Simulationswerkzeug bisher nicht betrachtet, haben in der Realität aber einen großen Einfluss auf das Verhalten der Funksysteme. Dennoch können die physikalischen Modelle der einzelnen Funkspezifikationen für weiterführende Untersuchungen verwendet werden.

- WinProp ([26]) von AWE Communications ist ein Tool zur Funknetzwerkplanung, aber auch zur Modellierung der Funkwellenausbreitung. Zum Zeitpunkt der Recherche wurden die Technologien GSM, GPRS, EDGE, UMTS, WLAN, WiMAX, DVB-H und DVB-SH unterstützt. Zur Untersuchung der Wellenausbreitung sind verschiedene Szenarien wählbar: Suburban, Urban, Indoor und Tunnel. Ein weiteres Tool zum Propagation Modelling ist WiSE ([91]). Jedoch reicht dieses Programm mit seinem Funktionsumfang nicht an WinProp heran.

- Factory Line WST von Phoenix Contact ist ein Tool zur Simulation der Ausbreitung von Bluetooth und WLAN-Funkfeldern. Die Struktur der Umgebung kann ohne viel Aufwand in das Tool überführt werden. Anschließend können die Bluetooth und WLAN-Komponenten des Herstellers Phoenix Contact in diesem Grundriss positioniert werden. Mit Hilfe der Simulation kann dann die Ausbreitung des Funkfeldes sichtbar gemacht werden. Das Tool soll den Anwender bei der Installation von Funklösungen unterstützen. Überflüssige Access Points können vermieden werden.

- Einen anderen Ansatz zur rechnergestützten Koexistenzbetrachtung bietet SEAMCAT (siehe Abb. 15). Das Softwaretool SEAMCAT ([70]) wurde für allgemeine Untersuchungen zur Koexistenz verschiedener Funkkommunikationssysteme, die in denselben oder in anliegenden Frequenzbändern arbeiten, entwickelt. SEAMCAT ist nicht zum Zweck der Systemplanung entwickelt worden.
  Der Simulationsalgorithmus basiert auf der Monte-Carlo-Methode, eine numerische Methode zur Lösung mathematischer Probleme mit Hilfe der Modellierung von Zufallsgrößen. Bei dieser Methode wird zur Untersuchung der Interferenzwahrscheinlichkeit das Simulationsszenario viele Male neu ausgewertet. Dabei werden jedes Mal gewisse Parameter, wie die Position der Geräte in einem vorgegebenen Raum oder die Frequenz in einem vorgegebenen Frequenzbereich, neu festgelegt. Das bedeutet, dass eine Vielzahl zufälliger Konstellationen dieser Parameter betrachtet wird. Allerdings findet keine Berücksichtigung des Zeitverhaltens der einzelnen Funkspezifikationen statt.

Abb. 15: *SEAMCAT-Oberfläche*

Daher ist in SEAMCAT die Simulation von genauen Sende- und Empfangsvorgängen der verschiedenen Funkspezifikationen kaum möglich. Bis auf wenige zusätzlich zum Standardsimulationsschema implementierte Verfahren ist dies bisher auch nicht vorgesehen. Auch die Möglichkeit, Plug-Ins selber zu schreiben und diese mit einzubinden, bietet nur wenig Spielraum, um den Simulationsalgorithmus anzupassen. Zwar lassen sich in jedem Simulationsdurchgang die Ergebnisse für die an den Empfängern aufgenommene Sendeleistung beliebig beeinflussen, aber zum Beispiel ist eine Wahl der Sendefrequenz für den aktuellen oder den nächsten Schritt aufgrund vorliegender Bedingungen, so wie es unter anderem zur Nachbildung eines FHSS-Schemas nötig wäre, nicht möglich. Wie bereits erwähnt, wird diese, sofern sie nicht konstant festgelegt wurde, zufällig bestimmt. Außerdem ist es nicht vorgesehen, dass die Geräte ihre Sende- bzw. Empfangsfunktion tauschen, was allerdings in der Realität häufig vorkommt.

Von großem Interesse bei der Betrachtung der Funkstandards ist auch der zeitliche Aspekt des Ablaufs. Um das Übertragungsmedium auch zeitlich optimal auszunutzen, ist es nötig, die spezifischen Zeitparameter im Sende- bzw. Empfangszyklus geeignet zu wählen. Allerdings lässt sich hierzu mit SEAMCAT keine Untersuchung durchführen, da eine zeitliche Betrachtung bisher keine Rolle gespielt hat.

Die Tabellen 4 und 5 vergleichen in einer Übersicht die vorgestellten Simulationsmethoden. Bei dem Vergleich werden auch die Einsatzmöglichkeiten für die industrielle Automation (IA) berücksichtigt.

Tab. 4: *Übersicht zu verschiedenen Simulationswerkzeugen und -ansätzen (Teil 1)*

Werkzeug	**OPNET**	**SEAMCAT**	**WinProp**	**OMNeT++**
Methode	Netzwerk- (Warteschlangen)/ Protokollsimulation	Stochastische Simulation	Funkwellenausbreitungssimulation	Netzwerk- (Warteschlangen)/ Protokollsimulation
Verhaltensmodell	Ereignisdiskret	Monte-Carlo	Analytisch	Ereignisdiskret
Technologien	WLAN, Bluetooth, ZigBee, UMTS, LTE	WLAN, Bluetooth, IEEE 802.15.4, GSM	WLAN, GSM, UMTS, LTE	WLAN, Bluetooth, IEEE 802.15.4, GSM
Typische Kenngrößen	Durchsatz	Interferenzwahrscheinlichkeit	Signalstärke	Durchsatz, Verzögerung
Kanalmodelle	Verschiedene Pfadverlustmodelle	Verschiedene Pfadverlustmodelle	- Ray Tracing - Dominant Path	Verschiedene Pfadverlustmodelle, Fading-Modelle
Koexistenzaussagen	Keine Angaben	ja	nein	ja
Vorteile	- Hunderte Protokolle (drahtgebunden u. drahtlos) - Viele Pfadverlustmodelle	- Aussage über die gegenseitige Beeinflussung von Funktechnologien möglich	- Ermittlung der exakten Empfangs- und Störleistung möglich	- Viele Protokolle - Viele Kanalmodelle - Koexistenzbetrachtung möglich
Nachteile	- Sehr teuer - Koexistenz? - Keine Kenngrößen der IA - Kein formales Modell	- Kein Zeit- und Fehlerverhalten - Kein Medienzugriff - Keine Kenngrößen der IA	- Kein Zeit- und Fehlerverhalten - Keine Koexistenz - Keine Kenngrößen der IA	- Keine Kenngrößen der IA - Kein formales Modell - Validierung?

Tab. 5: *Übersicht zu verschiedenen Simulationswerkzeugen und -ansätzen (Teil 2)*

Werkzeug	**NIST Coexistence Simulator**	**MATLAB/Simulink**	**Factory Line WST**
Methode	Signalflusssimulation	Signalflusssimulation	Funkwellenausbreitungssimulation
Verhaltensmodell	Keine Angabe	Zeitdiskret u. kontinuierlich	Analytisch
Technologien	IEEE 802.11b, Bluetooth	WLAN, Bluetooth, ZigBee, UMTS, LTE, GSM	WLAN, Bluetooth
Typische Kenngrößen	PER	BER, PER	Signalstärke
Kanalmodelle	AWGN-Kanal	AWGN-Kanal	Dominant Path
Koexistenzaussagen	ja	ja	nein
Vorteile	- Betrachtung der Koexistenz aus nachrichtentechnischer Sicht möglich	- Betrachtung der Koexistenz aus nachrichtentechnischer Sicht möglich	- Ermittlung der exakten Empfangs- und Störleistung möglich
Nachteile	- Hauptsächlich Betrachtung des PHY - Keine Kenngrößen der IA	- Hauptsächlich Betrachtung des PHY - Eingeschränktes Zeitverhalten - Keine Kenngrößen der IA	- Kein Zeit- und Fehlerverhalten - Keine Koexistenz - Keine Kenngrößen der IA

2.7 Koexistenz in existierenden Funkspezifikationen

Die erste Spezifikation, welche sich mit der Koexistenz in der drahtlosen Kommunikation befasste, ist die IEEE 802.16.2-2001 ([15]). Diese beschränkt sich auf den Hochfrequenzbereich von 10 GHz bis 66 GHz. Ein besonderer Fokus liegt auf dem Abschnitt zwischen 23,5 GHz und 43,5 GHz. Sie empfiehlt anhand von Leitfäden spezifische Aufstellungsmethoden zur Minimierung der Interferenzen zwischen festen Breitbandfunksystemen.

Die IEEE 802.16.2-2001 wurde von der IEEE 802.16.2-2004 ([16]) und der IEEE 802.15.2-2003 ([11]) abgelöst. Diese betrachten neben festen Breitbandfunksystemen auch Personal, Local und Metropolitan Area Networks. Der Fokus liegt dabei auf lizenzfreien Frequenzbändern. Auch diese beiden Spezifikationen analysieren entsprechende Koexistenzszenarien und liefern eine Beratung zu Systemdesign, Aufstellung, Koordination und Frequenznutzung.

Es folgte die IEEE 802.15.4-2003 ([13]), die im Anhang (Annex E) Einflussfaktoren der Koexistenz andeutet.

Die IEEE 1900.2-2008 ([3]) empfiehlt Analysekriterien zur Messung von Interferenzen zwischen Funksystemen. Sie enthält eine vollständige Liste von Koexistenzfaktoren sowohl des Physical als auch des MAC Layers. Außerdem empfiehlt diese Spezifikation eine Struktur für einen Koexistenzbericht.
Die IEEE 1900.2-2008 ist ein Ergebnis der „IEEE 1900.2 Working Group on Recommended Practice for Interference and Coexistenz Analysis“, die sich nicht vordergründig mit Störbeeinflussungen, sondern mit der Analyse von Möglichkeiten zur Koexistenz zwischen Funksystemen im gleichen Frequenzband oder in benachbarten Frequenzbändern befasst hat. Derzeit untersucht diese Arbeitsgruppe den Einsatz und das Verhalten von kognitiven Medienzugriffsverfahren.

Weitere Aktivitäten gibt es in der „IEEE 802.19 Coexistence Technical Advisory Group“. Das Hauptaugenmerk liegt auf IEEE 802-Spezifikationen, die lizenzfreie Frequenzbänder verwenden. Aktuell konzentriert sich die Gruppe auf Anwendungsmethoden zur Bewertung der Koexistenz von Funknetzwerken. Außerdem werden in dieser Gruppe Methoden zur Vorhersage von Interferenzen zwischen den verschiedenen Funktechnologien und damit die Harmonisierung der verschiedenen IEEE-Standards angestrebt. Die Ergebnisdokumente zu den Untersuchungen der Koexistenz zweier Funktechnologien heißen „Coexistence Assurance“.

Der ISA Standardisierungsausschuss ISA-SP100 „Wireless Systems for Automation“ hat im Entwurf des ISA100.11a Standards ([1]) die Koexistenz von Funksystemen im 2,4 GHz ISM-Band und Strategien zur Verbesserung der Koexistenz allgemein beschrieben. Me-

thoden zur Ermittlung und Bewertung der Koexistenz von Funksystemen sind nicht in diesem Entwurf des Standards enthalten.

Die IEC 62657-2 ([8]) spezifiziert das Koexistenzmanagement für die industrielle Kommunikation. Dazu werden grundlegende Voraussetzungen, Begriffe, Parameter und Verfahren für die Koexistenz der Funkkommunikation festgelegt. Weiterhin enthält sie Leitfäden, Anforderungen und bewährte Methoden für die Verfügbarkeit und Funktionsfähigkeit der Funkkommunikation. Dabei wird die Koexistenz der Funkkommunikation über den gesamten Lebenszyklus als Arbeitshilfe für alle Personen mit relevanten Zuständigkeiten abgedeckt, damit diese mit den kritischen Aspekten in jeder Phase des Koexistenzmanagements in einer Automatisierungsanlage zurechtkommen können. Die IEC 62657-2 stellt als eine einheitliche Richtlinie einen allgemeinen Bezugspunkt für die Koexistenz der Funkkommunikation in industriellen Automatisierungsanlagen bereit, um die Anwender bei der Bewertung und Einschätzung des Aufwands in der Anlage zu unterstützen. Zu den Aspekten des Lebenszyklus gehören Planung, Entwicklung/Aufbau, Installation, Implementierung, Betrieb, Instandhaltung, Verwaltung und Schulung.

Die beschriebenen Spezifikationen bilden Leitfäden zur Minimierung der Interferenzen, nennen Koexistenzfaktoren und empfehlen Messmethoden zur Erfassung der gegenseitigen Störbeeinflussung. Sie betrachten die Koexistenz aus nachrichtentechnischer Sicht und sind daher für diese Arbeit nur wenig relevant. Eine Ausnahme bildet die IEC 62657-2. Diese Spezifikation empfiehlt den Einsatz eines Koexistenzmanagements in industriellen Automatisierungsanlagen. Dabei werden auch koexistenzrelevante Parameter aufgelistet und definiert, welche teilweise in einem Modell zur Untersuchung der Koexistenz berücksichtigt werden müssen. Neben Werkzeugen zur Planung der Signalausbreitung wird auch der Einsatz von Simulationen vorgeschlagen. Die Art der Simulation und deren Durchführung ist allerdings nicht festgelegt, sodass von einem Bedarf und einer Nachfrage für eine solche Simulationsumgebung ausgegangen werden kann.

3 Problemstellung

In den vergangenen Jahren hat sich die Funkkommunikation in der industriellen Automation etabliert. Die Ursache liegt in den verschiedenen Vorteilen, die bei der Verwendung der Funkkommunikation entstehen. Ein Beispiel ist die Einsparung von oft komplexen und teuren Kabeln, Kabelsicherungen und Steckern. Aber auch die Erhöhung der Mobilität und Flexibilität von Sensoren, Aktoren und ganzen Werkzeugmaschinen muss als Benefit genannt werden. Die aufgezählten Vorteile stechen in konkreten Anwendungsfällen hervor. Beispiele für Anwendungen der Funkkommunikation in industriellen Automatisierungsanlagen sind:

- Monitoring;
- Kommunikation mit mobilen Fachkräften;
- Drahtlose Sensoren und Aktoren an bewegten Teilen;
- Installationen, die Flexibilität bezüglich Werkzeug- und Maschinenrekonfiguration benötigen.

Jede Automatisierungsanwendung stellt unterschiedliche Anforderungen an das Zeit- und Fehlerverhalten der Funkkommunikation. Diese Anforderungen betreffen beispielsweise die Echtzeitfähigkeit, Robustheit und Zuverlässigkeit (siehe Abschnitt 2.2.4). Für Anwendungen mit Sensoren und Aktoren ist die Hauptanforderung das Echtzeitverhalten. Dies gilt sowohl für die Prozess- als auch für die Fertigungsautomation. In der Fertigungsautomation sind zusätzlich nur geringe Zeitverzögerungen durch die Übertragung zulässig. In der ETSI TR 102 889-2 ([84]) sind die wesentlichen Anforderungen der industriellen Automation an die Funkkommunikation zusammengefasst.
Um Aussagen über die Einhaltung von Anwendungsanforderungen treffen zu können, werden Kenngrößen wie die Übertragungszeit, Anwortzeit und PLR (siehe Abschnitt 2.2.2) verwendet.
Eine Nichterfüllung der Anwendungsanforderungen durch ein Funksystem führt in der Regel zu einer Unterbrechung des Herstellungsprozesses. Dies hat Produktionsausfälle und damit erhebliche Kosten zur Folge.

Die unterschiedlichen Anforderungen der verschiedenen Automatisierungsanwendungen bilden die Daseinsberechtigung für eine Vielzahl an Funktechnologien. Diese Funktechnologien unterscheiden sich beispielsweise im verwendeten Medienzugriffsverfahren (siehe Abschnitt 2.4.2), der Bandbreite und Modulation des übertragenen Signals sowie der Kodierung. Eine Automatisierungslösung verlangt sehr oft die gleichzeitige Verwendung von mehr als nur einer Funktechnologie innerhalb desselben Bereiches. An dieser Stelle kommt es zur Frage der Koexistenz (siehe Abschnitt 2.1.1) zwischen Funksystemen mit gleichem oder unterschiedlichem Medienzugriffsmechanismus, da von sämtlichen Teilnehmern bzw.

Systemen ein gemeinsames Medium zur Kommunikation verwendet wird. Bei einer friedlichen Koexistenz sind keine oder nur geringe Auswirkungen auf das Übertragungs- und damit Zeitverhalten festzustellen und die bereits genannten Anforderungen können erfüllt werden. Andererseits können sich die auf das gemeinsame Medium zugreifenden Systeme gegenseitig auch so beeinflussen und stören, dass es zu einer Verschlechterung der Kenngrößen der Automation kommt und damit die für die Anwendung erforderlichen Anforderungen nicht mehr eingehalten werden können. Dies hätte verheerende Auswirkungen für den zu regelnden und zu steuernden Prozess.

Eine friedliche Koexistenz kann mit Hilfe einer Reihe von koexistenzrelevanten Parametern erreicht und hergestellt werden (siehe Abschnitt 2.1.2). Diese Parameter werden gemäß Abschnitt 2.4.1 vom Physical und MAC Layer bestimmt. Neben der Herstellung der friedlichen Koexistenz soll auch das Spektrum als knappe Ressource effizient genutzt werden. Die Medienzugriffsmechanismen bieten Reserven und Potentiale, um friedliche Koexistenz und die damit verbundene Einhaltung von Anwendungsanforderungen erreichen und gewährleisten zu können. Auch können die Art und das Verhalten der Medienzugriffsmechanismen zur effizienten Nutzung des Spektrums beitragen. Aus den genannten Gründen soll der Fokus der Arbeit auf die Untersuchung von Medienzugriffsmechanismen aus Sicht der industriellen Automation gelegt werden.

Zur Verbesserung des Koexistenzverhaltens existieren auch im Bereich der Hochfrequenztechnik Innovationen. In diesem Zusammenhang spielen Software Defined Radios und kognitive Systeme eine Rolle. Diese sollen allerdings nicht Bestandteil dieser Arbeit sein.

In Abschnitt 2.6 wurden verschiedene Simulationsmethoden und -werkzeuge verglichen, die den Fokus auf die Analyse und Untersuchung der Funkkommunikation und des Koexistenzverhaltens legen. Diese Tools betrachten jedes für sich unterschiedliche Aspekte. Beispielsweise wird die Signalausbreitung oder die Interferenzwahrscheinlichkeit simuliert. Bisher existiert allerdings kein Simulationsansatz, welcher die Funkkommunikation aus Sicht der industriellen Automation betrachtet. So fehlt zum Beispiel die Berücksichtigung der Anwendungsanforderungen oder der automatisierungsspezifischen Kenngrößen. Auch sollte die Simulation auf einem formalen Modell basieren. Für einen solchen Modellansatz existieren eine Reihe von Anforderungen:

1. Das Modell sollte ereignisdiskret arbeiten, weil sämtliche Kommunikationsprotokolle als ereignisdiskrete Systeme beschrieben werden.

2. Zeitbedingungen müssen modellierbar sein, da der Medienzugriff als auch die Kenngrößen der industriellen Automation zeitbehaftet sind.

3. Parameterbehaftete Ereignisse müssen im Modell Berücksichtigung finden, da die Medienzugriffsverfahren und das Übertragungsverhalten von Parametern abhängig sind.

4. Das Modell muss Nebenläufigkeiten realisieren können, da sich mehrere Teilnehmer gleichzeitig dasselbe Medium teilen.

Bisherige Modellansätze für Koexistenzuntersuchungen (siehe Abschnitt 2.3 und 2.6) erfüllen nicht diese aufgelisteten Anforderungen (1-4). Des Weiteren würde ein geschlossenes analytisches Modell viel zu komplex sein, da eine Vielzahl von Einflussgrößen auf das Koexistenzverhalten und auf die Kenngrößen der Funklösungen einwirken. Simulationen, z. B. basierend auf Petri-Netzen, bilden eine geeignete Alternative zum analytischen Ansatz. Mit Petri-Netz-Modellen wurden bisher nur einzelne Funktechnologien nachgebildet und untersucht (siehe Abschnitt 2.3.2). Petri-Netze stellen eine formale Modellbeschreibung dar und können die genannten Anforderungen (1-4) erfüllen. Auch sind Koexistenzuntersuchungen mit diesem Ansatz möglich.

Die Modellierung und Simulation des Koexistenzverhaltens ist in der industriellen Automation für das Koexistenzmanagement von entscheidender Bedeutung. Allerdings wird der Anwender neben dem Koexistenzmanagement auch bei der Auswahl, Planung und Diagnose von Funklösungen allein gelassen. Simulationen können ihn bei diesen Problemstellungen unterstützen und zusätzlich funkrelevante Parameter während des Engineerings optimieren. Die in Abschnitt 2.6 vorgestellten Tools sind dazu, wie bereits erwähnt, ungeeignet. Es wird daher eine Simulationsumgebung benötigt, die in den Phasen der Planung, des Engineerings, der Diagnose und vor allem beim Koexistenzmanagement den Anwender unterstützt und auf mögliche Kollisionen zwischen verschiedenen Funksystemen und die daraus resultierenden Paketverzögerungen und -fehler aufmerksam macht, um einen fehlerfreien Betrieb der Funkkommunikation in automatisierungstechnischen Anlagen gewährleisten zu können ([74]).

4 Theoretischer Ansatz

4.1 Formale Definition von kolorierten Petri-Netzen

Ein Coloured Petri Net CPN ist ein Tupel CPN=(P, T, A, S, V, C, G, E, I) mit

P	ist eine endliche Menge an Elementen, die als Plätze bezeichnet werden (Places).
T	ist eine endliche Menge an Elementen, die als Transitionen bezeichnet werden (Transitions).
A	ist eine endliche Menge an Elementen, die als Kanten bezeichnet werden (Arcs).
S	ist eine endliche Menge an nicht-leeren Elementen, die als Colour Sets bezeichnet werden.
V	ist eine endliche Menge an Elementen, die als typisierte Variablen bezeichnet werden.
C	Colour Set Funktion (weist Plätzen Colour Sets zu)
G	Guard Funktion (weist Transitionen Guards zu)
E	Kantenausdrucksfunktion (weist Kanten Kantenausdrücke zu)
I	Initialisierungsfunktion (weist Plätzen Initialisierungsmarkierungen zu)

Dabei sind Plätze, Transitionen und Kanten für die Umsetzung der Netzstruktur verantwortlich. Mit S und V werden Typen und Variablen definiert. C, G, E und I dienen der Netzbeschriftung. Plätze und Transitionen werden unter dem Begriff Knoten (Nodes) zusammengefasst. Allerdings gilt dabei:

$$P \cap T = \emptyset \tag{9}$$

Das heißt, dass ein Knoten entweder ein Platz oder eine Transition ist. Er kann nicht beides sein.

Für die Menge der gerichteten Kanten A wird gefordert:

$$A \subseteq P \times T \cup T \times P \tag{10}$$

Jede Kante beginnt in einem Platz und endet in einer Transition oder sie beginnt in einer Transition und endet in einem Platz. Dabei wird häufig auch eine Unterteilung vorgenommen. Für die Menge der gerichteten Kanten zwischen Plätzen und Transitionen (Vorbereich) gilt:

$$Pre \subseteq P \times T \tag{11}$$

Auf identische Weise gilt für die Menge der gerichteten Kanten zwischen Transitionen und Plätzen (Nachbereich):

$$Post \subseteq T \times P \tag{12}$$

Jedem Platz P wird mit Hilfe einer Colour Set Funktion C ein Colour Set S zugewiesen:

$$C : P \to S \tag{13}$$

Weiterhin gilt für typisierte Variablen der Menge V:

$$Type[v] \in S \quad \forall \quad v \in V \tag{14}$$

Der Typ einer Variable muss einem der Elemente entsprechen, die in der Menge S definiert sind.

Eine Guard Funktion weist jeder Transition einen Guard zu:

$$G : T \to EXPR_V \tag{15}$$

wobei alle Variablen des Ausdrucks der Menge V angehören müssen. Weiterhin wird ein boolscher Datentyp gefordert:

$$Type[G(t)] = Bool \quad \forall \quad t \in T \tag{16}$$

Eine Kantenausdrucksfunktion weist jeder Kante einen Kantenausdruck zu:

$$E : A \to EXPR_V \tag{17}$$

wobei alle Variablen des Ausdrucks der Menge V angehören müssen. Weiterhin wird gefordert, dass:

$$Type[E(a)] = C(p)_{MS} \quad \forall \quad a \in A \tag{18}$$

wobei p der Platz ist, der mit der Kante a verbunden ist. Der Kantenausdruck muss mit einer Multimenge (Multiset, MS) von Marken übereinstimmen, welche dem Colour Set des verbundenen Platzes entsprechen.

Eine Initialisierungsfunktion weist jedem Platz eine Initialisierungsmarkierung M_0 zu:

$$I : P \to EXPR_\emptyset \tag{19}$$

wobei der Initialisierungsausdruck keine Variablen enthalten darf. Weiterhin wird gefordert, dass:

$$Type[I(p)] = C(p)_{MS} \quad \forall \quad p \in P \tag{20}$$

Der Initialisierungsausdruck muss mit einer Multimenge von Marken übereinstimmen, welche dem Colour Set des Platzes entsprechen. Für die Initialisierungsmarkierung gilt dann:

$$M_0(p) = I(p) \quad \forall \quad p \in P \tag{21}$$

Eine Markierung ist ganz allgemein eine Funktion M, die jedem Platz eine Multimenge an Marken zuordnet:

$$M(p) = C(p)_{MS} \tag{22}$$

Alle Marken müssen zu dem Colour Set des Platzes passen.

Die Variablen einer Transition befinden sich entweder im Guard oder im Kantenausdruck einer Kante, die mit der Transition verbunden ist. Für die Menge dieser Variablen einer Transition t gilt:

$$Var(t) \subseteq V \tag{23}$$

Als Bindung einer Transition t wird eine Funktion b bezeichnet, die jeder Variable

$$v \in Var(t) \tag{24}$$

einen Wert

$$b(v) \in Type[v] \tag{25}$$

zuordnet. Die Menge aller Bindungen einer Transition t ist als B(t) gekennzeichnet. Ein Bindungselement ist ein Paar (t,b), wobei t eine Transition und b eine Bindung dieser Transition mit

$$b \in B(t) \tag{26}$$

ist. Die Menge aller Bindungselemente einer Transition t ist als BE(t) gekennzeichnet. Die Menge aller Bindungselemente in einem CPN ist als BE gekennzeichnet.

Die Regeln für die Freigabe und den Eintritt bzw. das Feuern einer Transition basieren auf der Bewertung der Guard- und Kantenausdrücke. Dabei werden folgende Schreibweisen verwendet:

G(t)<b>	Bewertung des Guardausdrucks für t in Bezug auf die Bindung b
E(a)<b>	Bewertung des Kantenausdrucks für a in Bezug auf die Bindung b
E(p,t)<b>	Bewertung des Kantenausdrucks der Kante von p nach t in Bezug auf die Bindung b. Wenn eine solche Kante nicht existiert, gilt $E(p,t) = \emptyset_{MS}$.
E(t,p)<b>	Bewertung des Kantenausdrucks der Kante von t nach p in Bezug auf die Bindung b. Wenn eine solche Kante nicht existiert, gilt $E(t,p) = \emptyset_{MS}$.

Bindungen werden in eckigen Klammern geschrieben.

Ein Bindungselement

$$(t,b) \in BE \tag{27}$$

ist in einer Markierung M freigegeben, wenn und nur wenn folgende zwei Eigenschaften erfüllt sind:

- Die Bedingung des Guards muss wahr sein.

$$G(t) < b >= true \tag{28}$$

- Die Marken, die von dem Eingangskantenausdruck benötigt werden, müssen in der aktuellen Markierung M vorliegen.

$$E(p,t) < b ><<= M(p) \quad \forall \quad p \in P \tag{29}$$

Wenn ein Bindungselement (t,b) in einer Markierung M freigegeben ist, könnte es geschehen, dass dies zu einer neuen Markierung M' führt:

$$M'(p) = (M(p) - E(p,t) < b >) + E(t,p) < b > \quad \forall \quad p \in P \tag{30}$$

Dabei werden von der alten Markierung Marken, die durch die Eingangskanten verbraucht werden, abgezogen und Marken, die von den Ausgangskanten erzeugt werden, hinzugefügt.

Ein Schritt ist eine nicht-leere und endliche Multimenge an Bindungselementen:

$$Y \subseteq BE_{MS} \tag{31}$$

Er ist in einer Markierung M freigegeben, wenn und nur wenn die folgenden zwei Eigenschaften erfüllt sind:

- Die Bedingungen aller Guards müssen wahr sein.

$$G(t) < b >= true \quad \forall \quad (t,b) \in Y \tag{32}$$

- Die Marken, die von dem Eingangskantenausdruck benötigt werden, müssen in der aktuellen Markierung M vorliegen.

$$\sum_{(t,b)\in Y} E(p,t) < b ><<= M(p) \quad \forall \quad p \in P \tag{33}$$

Wenn der Schritt in einer Markierung M freigegeben ist, könnte es geschehen, dass dies zu einer neuen Markierung M' führt:

$$M'(p) = (M(p) - \sum_{(t,b)\in Y} E(p,t) < b >) + \sum_{(t,b)\in Y} E(t,p) < b > \quad \forall \quad p \in P \tag{34}$$

Dabei werden von der alten Markierung Marken, die durch die Eingangskanten verbraucht werden, abgezogen und Marken, die von den Eingangskanten erzeugt werden, hinzugefügt.

Für das Stattfinden eines Schritts wird folgende Notation verwendet:

$$M_1 \xrightarrow{Y} M_2 \tag{35}$$

Der Schritt Y sorgt für den Übergang von der Markierung M_1 zur Markierung M_2.

Ein Hierarchical Coloured Petri Net HCPN ist ein 4-Tupel HCPN=(MOD, SM, PS, FS) mit

MOD	ist eine endliche Menge an Elementen, die als Module bezeichnet werden (Modules).
SM	ist eine Submodulfunktion, die jeder Substitutionstransition ein Submodul zuweist.
PS	ist eine Port-Socket-Beziehungsfunktion, die jeder Substitutionstransition eine Port-Socket-Beziehung zuweist.
FS	ist eine endliche, nicht-leere Menge an Elementen, die als Fusions bezeichnet werden.

Dabei ist ein Coloured Petri Net Modul ein 4-Tupel CPNMOD=(CPN, T_{sub}, P_{port}, PT) mit

CPN	ist ein nicht-hierarchisches Coloured Petri Net (P, T, A, S, V, C, G, E, I).
T_{sub}	ist eine endliche Menge an Elementen, die als Substitutionstransitionen bezeichnet werden (Substitution Transition).
P_{port}	ist eine endliche Menge an Elementen, die als Port-Plätze bezeichnet werden (Port Place).
PT	ist eine Porttypfunktion, die jedem Port-Platz einen Porttypen zuweist.

Jedes Modul hat eine möglicherweise leere Menge von Substitutionstransitionen T_{sub}, für die gilt:

$$T_{sub} \subseteq T \tag{36}$$

Allgemein kann ein Modul beides enthalten: normale Transitionen und Substitutionstransitionen. Da Substitutionstransitionen eine Teilmenge der Transitionen in einem Modul sind, haben sie Guards und die Kanten, die mit der Substitutionstransition verbunden sind, besitzen Kantenausdrücke. Allerdings können Substitutionstransitionen nicht freigegeben werden oder feuern.

Weiterhin hat jedes Modul eine möglicherweise leere Menge von Portplätzen P_{port}, für die gilt:

$$P_{port} \subseteq P \tag{37}$$

Die dazugehörige Porttypfunktion PT spezifiziert, ob der Portplatz ein Eingangs-Port (IN), ein Ausgangs-Port (OUT) oder ein Eingangs-/Ausgangs-Port (I/O) ist.

$$PT : P_{port} \rightarrow \{IN, OUT, I/O\} \tag{38}$$

Die Input-Socket-Plätze einer Substitutionstransition t sind die Menge an Eingangsplätzen einer Transition und Output-Socket-Plätze sind die Menge an Ausgangsplätzen. Input-/Output-Socket-Plätze sind letztendlich die Menge an Eingangs-/Ausgangsplätzen der Transition. Die Socket-Plätze $P_{sock}(t)$ einer Substitutionstransition t stellen die Vereinigung der Input-, Output- und Input/Output-Sockets für die Transition dar. Socket-Plätze sind nicht explizit als Komponenten eines Moduls definiert, da sie implizit über die Kanten, mit denen die Substitutionstransition verbunden ist, bestimmt werden. Für jede Substitutionstransition t wird eine Sockettypfunktion ST(t) definiert, die jeden Socket-Platz von

t auf dessen Typ abbildet:

$$ST(t)(p) = \begin{cases} IN, & \text{if } p \in p_{sock}^{in}(t) \\ OUT, & \text{if } p \in p_{sock}^{out}(t) \\ I/O, & \text{if } p \in p_{sock}^{i/o}(t) \end{cases} \tag{39}$$

Jedes Modul mod ∈ MOD ist definiert mit:

$$mod = ((P^{mod}, T^{mod}, A^{mod}, S^{mod}, V^{mod}, C^{mod}, G^{mod}, E^{mod}, I^{mod}), T_{sub}^{mod}, P_{port}^{mod}, PT^{mod}) \tag{40}$$

Es wird gefordert, dass

$$(P^{mod_1} \cup T^{mod_1}) \cap (P^{mod_2} \cup T^{mod_2}) = \emptyset \quad \forall \quad mod_1, mod_2 \in MOD \tag{41}$$

sodass

$$mod_1 \neq mod_2 \tag{42}$$

Die Plätze und Transitionen in den einzelnen Modulen sollen gegenseitig disjunkt sein.

SM ist eine Submodulfunktion, die jeder Substitutionstransition ein Submodul zuweist:

$$SM : T_{sub} \rightarrow MOD \tag{43}$$

PS ist eine Port-Socket-Beziehungsfunktion, die jeder Substitutionstransition t eine Port-Socket-Beziehung zuweist:

$$PS(t) \subseteq P_{sock}(t) \times P_{port}^{SM(t)} \tag{44}$$

Es wird gefordert, dass die Socket- und Porttypen sowie die Colour Set Funktionen und Initialisierungsfunktionen eines Socket-Platzes p und eines dazugehörigen Port-Platzes p' gleich sind.

$$ST(p) = PT(p'), C(p) = C(p'), I(p) = I(p') \quad \forall \quad (p, p') \in PS(t), t \in T_{sub} \tag{45}$$

Eine Fusionsmenge fs ∈ FS ist ein Teil der Menge von allen Untermengen von Plätzen, sodass gilt:

$$FS \subseteq 2^P \tag{46}$$

Es wird gefordert, dass alle Plätze einer Fusionsmenge fs identische Colour Sets haben und die Ausdrücke der Initialisierungsmarkierungen identischen Multimengen an Marken zugeordnet sein müssen. Das bedeutet, dass für alle Plätze p, p', die einer Fusionsmenge fs angehören, gelten muss:

$$C(p) = C(p'), I(p) = I(p') \quad \forall \quad (p, p') \in fs, fs \in FS \tag{47}$$

Ein zeitbehaftetes, nicht-hierarchisches Coloured Petri Net TCPN ist ein Tupel TCPN=(P, T, A, S, V, C, G, E, I) mit

P	ist eine endliche Menge an Elementen, die als Plätze bezeichnet werden (Places).
T	ist eine endliche Menge an Elementen, die als Transitionen bezeichnet werden (Transitions).
A	ist eine endliche Menge an Elementen, die als Kanten bezeichnet werden (Arcs).
S	ist eine endliche Menge an nicht-leeren Elementen, die als Colour Sets bezeichnet werden.
V	ist eine endliche Menge an Elementen, die als typisierte Variablen bezeichnet werden.
C	Colour Set Funktion, die Plätzen Colour Sets zuweist.
G	Guard Funktion, die Transitionen Guards zuweist.
E	Kantenausdrucksfunktion, die Kanten Kantenausdrücke zuweist.
I	Initialisierungsfunktion, die Plätzen Initialisierungsmarkierungen zuweist.

Die Definition von TCPN ist identisch zur Definition von nicht-hierarchischen CPNs. Anpassungen sind nur in den Colour Sets S, der Colour Set Funktion C, der Kantenausdrucksfunktion E und der Initialisierungsfunktion I notwendig. Die Unterschiede sind, dass jedes Colour Set in S entweder zeitbehaftet (timed) oder nicht-zeitbehaftet (untimed) ist. Ein Platz mit einem zeitbehafteten Colour Set wird zeitbehafteter Platz (timed place) und ein Platz mit nicht-zeitbehafteten Colour Set wird nicht-zeitbehafteter Platz (untimed place) genannt. Auf ähnliche Weise werden die umgebenden Kanten von zeitbehafteten Plätzen zeitbehaftete Kanten genannt und die umgebenden Kanten von nicht-zeitbehafteten Plätzen werden als nicht-zeitbehaftete Kanten bezeichnet. Es gilt:

$$\begin{aligned} Type[E(a)] &= C(p)_{MS}, \quad \text{wenn p untimed ist} \\ Type[E(a)] &= C(p)_{TMS}, \quad \text{wenn p timed ist} \end{aligned} \tag{48}$$

Für einen zeitbehafteten Platz wird gefordert, dass der Ausdruck der Initialmarkierung einer zeitbehafteten Multimenge entspricht. Bei einem nicht-zeitbehafteten Platz bezieht sich der Ausdruck der Initialmarkierung auf eine nicht-zeitbehaftete Multimenge. Es gilt:

$$\begin{aligned} Type[I(p)] &= C(p)_{MS}, \quad \text{wenn p untimed ist} \\ Type[I(p)] &= C(p)_{TMS}, \quad \text{wenn p timed ist} \end{aligned} \tag{49}$$

Die Festlegungen für Markierungen in einem TCPN sollen im Folgenden erläutert werden. Eine Markierung ist eine Funktion M, die jedem Platz p ∈ P eine Multimenge M(p) von Marken zuweist, sodass:

$$\begin{aligned} M(p) &\in C(p)_{MS}, \quad \text{wenn p untimed ist} \\ M(p) &\in C(p)_{TMS}, \quad \text{wenn p timed ist} \end{aligned} \tag{50}$$

Eine zeitbehaftete Markierung ist ein Paar (M,t*), wobei M eine Markierung ist und t* $\in$ **T** den Wert der globalen Uhr darstellt. Die zeitbehaftete Initialmarkierung ist ein Paar (M_0,0), wobei M_0 durch

$$M_0(p) = I(p) \quad \forall \quad p \in P \tag{51}$$

definiert ist.

Ein Schritt Y $\in$ BE_{MS} ist in einer zeitbehafteten Markierung (M,t*) zu einer Zeit t' freigegeben, wenn und nur wenn die folgenden Eigenschaften erfüllt sind:

- Jedes Bindungselement (t,b), welches zu dem Schritt Y gehört, muss dem Guard von t gerecht werden.

$$G(t) < b >= true \quad \forall \quad (t,b) \in Y \tag{52}$$

- Bei einem nicht-zeitbehafteten Platz p muss dieser mit einer Multimenge von Marken M(p) belegt sein, die größer oder gleich der Summe an Marken ist, die von p durch die einzelnen Bindungselemente des Schrittes Y entfernt werden.

$$\sum_{(t,b)\in Y} E(p,t) < b ><<= M(p) \quad \forall \quad \text{untimed } p \in P \tag{53}$$

- Für zeitbehaftete Plätze sind die Anforderungen ähnlich, nur dass die globale Uhr und die Zeitstempel von Marken berücksichtigt werden müssen. Die Multimenge an Marken, die von p entfernt werden, wenn Y zur Zeit t' eintritt, wird durch die Festlegung der Ausgangskantenausdrücke von p und durch die Wertaddierung der globalen Uhr t' zu den Zeitstempeln der resultierenden zeitbehafteten Multimenge bestimmt.

$$\sum_{(t,b)\in Y} (E(p,t) < b >)_{+t'} <<<= M(p) \quad \forall \quad \text{timed } p \in P \tag{54}$$

- Ein Schritt Y ist zu einer Zeit t' in einer zeitbehafteten Markierung (M,t*) freigegeben, wenn gilt:

$$t^* \leq t' \tag{55}$$

- t' ist der kleinste Zeitwert, für den ein Schritt existiert, der die oberen Bedingungen erfüllt.

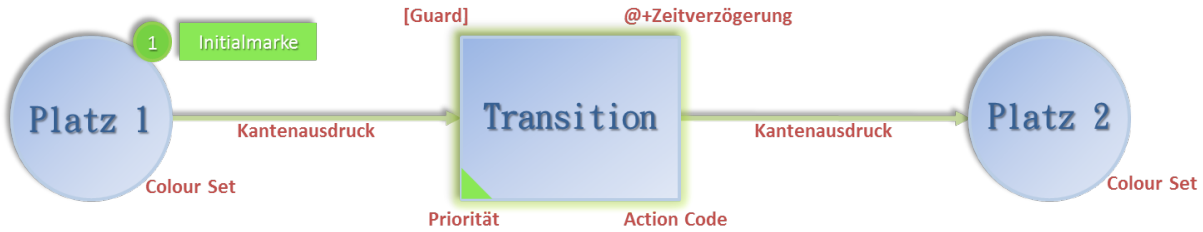

Abb. 16: *Grafische Notation von kolorierten Petri-Netzen*

Wenn der Schritt Y in (M,t*) zu einer Zeit t' freigegeben ist, kann es zur Zeit t' passieren, dass dies zu einer zeitbehafteten Markierung (M',t') führt, die definiert ist durch:

$$M'(p) = (M(p) - \sum_{(t,b)\in Y} E(p,t) < b >) + \sum_{(t,b)\in Y} E(t,p) < b > \quad \forall \quad \text{untimed} p \in P \tag{56}$$

$$M'(p) = (M(p) - \sum_{(t,b)\in Y} E(p,t) < b >_{+t'}) + \sum_{(t,b)\in Y} E(t,p) < b >_{+t'} \quad \forall \quad \text{timed} p \in P \tag{57}$$

Für das Stattfinden eines Schritts wird folgende Notation verwendet:

$$(M_1, t_1^*) \xrightarrow{Y_1} (M_2, t_2^*) \tag{58}$$

Dabei führt das Stattfinden des Schritts Y_1 zum Übergang von (M_1, t_1^*) zu (M_2, t_2^*).

Die formalen Definitionen von CPN, HCPN und TCPN basieren auf den Quellen [46] und [2].

4.2 Grafische Notation von kolorierten Petri-Netzen

Abb. 16 zeigt die wesentlichen Elemente der grafischen Notation. Die grafische Form umfasst zwei Teile: Einen Graphen, welcher Netzelemente grafisch darstellt und textuelle Aufschriften trägt, und Deklarationen, welche alle Typen, Variablen, Konstanten und Funktionen definiert, die zur weiteren Erläuterung des Graphen genutzt werden. Die Deklaration enthält in der Regel auch die Initialmarkierung. Der Graph besteht aus Plätzen, Transitionen, Kanten und Marken bzw. Markierungen. Die genannten Elemente werden im Folgenden genauer erläutert.

Plätze werden mit Ellipsen bzw. Kreisen dargestellt. Drei Bestandteile sind mit einem Platz p verknüpft:

- der Platzname;
- der Name des Colour Sets, welches mit dem Platz verknüpft ist; und
- die Initialmarkierung $M_0(p)$.

Die Position dieser Elemente in Bezug auf die Plätze ist nicht festgelegt. Allerdings befindet sich der Platzname in der Regel innerhalb des Kreises. Das dazugehörige Colour Set ist rechts unterhalb des Kreises angeordnet. Wenn die Initialisierungsmarkierung leer ist, kann diese auch wegfallen.

Eine Transition wird durch ein Rechteck repräsentiert. Fünf Bestandteile können mit dieser verknüpft sein:

- der Transitionsname;
- der Guard, welcher durch rechteckige Klammern gekennzeichnet ist;
- das Delay, welches an einem vorangestellten @-Zeichen erkennbar ist;
- einem Action Code, der beim Feuern der Transition ausgeführt wird; und
- einer Priorität.

Wenn Guard, Delay oder Action Code bei einer Transition nicht notwendig sind, können diese auch entfallen. Die Position der einzelnen Bestandteile in Bezug auf die Transition ist nicht festgelegt. Allerdings befindet sich der Transitionsname in der Regel innerhalb des Rechtecks. Der Guard ist an der linken, oberen Ecke und das Delay an der rechten, oberen Ecke angeordnet. Der optionale Action Code ist an der Ecke rechts unterhalb des Rechtecks angebunden. Die optionale Priorität ist an der linken, unteren Ecke angeordnet. Sie wird in Form einer positiven, ganzen Zahl angegeben. Je kleiner der Wert, desto höher ist die Priorität. Standardwerte sind P_HIGH (100), P_NORMAL (1000) und P_LOW (10000). Sie dient zur Vermeidung von Konflikten, wenn beispielsweise mehrere Transitionen durch die Marke eines gemeinsamen Eingangsplatzes gleichzeitig feuern könnten. Wird kein Wert angegeben, ist die Priorität P_NORMAL.

Eine Kante wird durch einen Pfeil dargestellt. Im Fall von $(p,t) \in A$ verläuft ein Pfeil vom Platz p zur Transition t. Auf identische Weise startet der Pfeil bei $(t,p) \in A$ in der Transition t und endet im Platz p. Wenn zwei gegenläufige Pfeile die gleiche Aufschrift tragen, können diese durch einen Doppelpfeil ersetzt werden. Die Aufschrift bleibt erhalten. Kanten sind mit Kantenausdrücken verknüpft.

Eine Marke ist die Zuordnung von konkreten Werten zu einem Colour Set. Die grafische Darstellung einer Markierung erfolgt bei einem Platz mit dessen Mulitmenge an Marken durch die Anzeige einer symbolischen Summe. Die Darstellung bei kolorierten Petri-Netzen besteht aus zwei Bestandteilen. Einerseits einem Kreis, welcher an den Platz angehaftet ist. Dieser trägt die Gesamtanzahl an Marken. Neben diesem Kreis befindet sich noch ein Rechteck, welches die konkreten Marken anzeigt. Die Notation besteht aus einer ganzen Zahl, einem Hochkomma und dem konkreten Wert der Marke. Die ganze Zahl vor dem

Hochkomma gibt an, wie viele Marken mit einer identischen Wertzuweisung auf dem entsprechenden Platz liegen. Marken mit unterschiedlichen Werten werden getrennt durch ein zweifaches Plus aufgelistet. Die Position der beiden grafischen Bestandteile ist nicht festgelegt.

Informationen zu der grafischen Notation von kolorierten Petri-Netzen stammen aus der Quelle [46].

4.3 Formale Definition von Zustandsräumen

Es ist möglich, jedes Petri-Netz in einen Zustandsraum zu transformieren, der aus einem Graphen besteht.
Ein gerichteter Graph mit Kantenbeschriftungen aus der Menge L ist ein Tupel DG=(N, A) mit

N	ist eine Menge von Knoten.
A	ist eine Menge von Kanten.

Für die Menge der Kanten A gilt:

$$A \subseteq N \times L \times N \tag{59}$$

Eine Kante a ∈ A führt von einem Knoten n zu einem Knoten n' (n, n' ∈ N) und ist mit l ∈ L beschriftet. Bei der Darstellung eines gerichteten Graphen wird jeder Knoten n durch eine abgerundete Box mit der Aufschrift n repräsentiert. Jede Kante ist ein Pfeil mit der Beschriftung l, welcher vom Quellknoten startet und im Zielknoten endet (siehe Abb. 17).

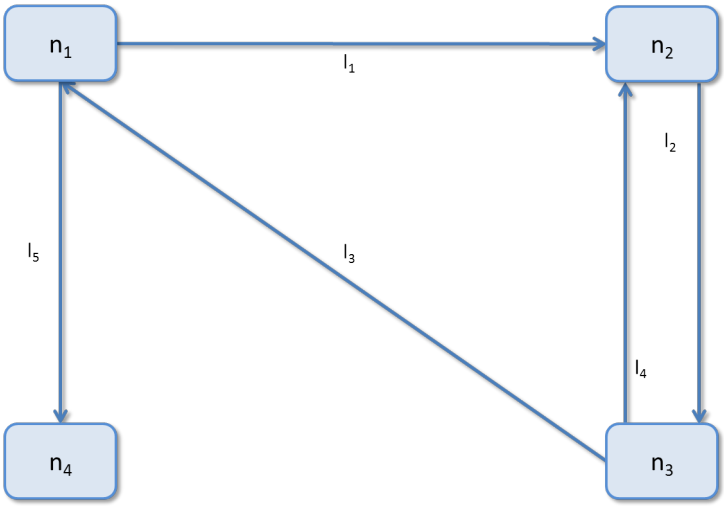

Abb. 17: *Darstellung des State Space*

Ein gerichteter Graph ist endlich, wenn die Anzahl an Knoten und Kanten endlich ist. Ansonsten ist er unendlich. Die Menge an Knoten ohne abgehende Kanten wird als Endknoten bezeichnet. Knoten ohne ankommende Kanten sind Initialknoten.

Wenn DG=(N, A) ein gerichteter Graph ist, so kann ein endlicher gerichteter Pfad der Länge $k \geq 0$ als wechselnde Sequenz von Knoten und Kanten in folgender Form dargestellt werden:

$$n_1(n_1, l_1, n_2)n_2(n_2, l_2, n_3)\cdots n_k(n_k, l_k, n_{k+1})n_{k+1} \tag{60}$$

Ein unendlicher gerichteter Pfad ist eine wechselnde unendliche Sequenz von Knoten und Kanten in der Form:

$$n_1(n_1, l_1, n_2)n_2(n_2, l_2, n_3)n_3(n_3, l_3, n_4)n_4\cdots \tag{61}$$

Angenommen, dass DG=(N, A) und DG'=(N', A') gerichtete Graphen sind, dann ist DG' ein Subgraph von DG, wenn und nur wenn gilt:

$$N' \subseteq N \tag{62}$$

und

$$A' \subseteq A \tag{63}$$

DG' ist ein bedingter Subgraph von DG, wenn und nur wenn gilt:

$$N' \subseteq N \tag{64}$$

und

$$A' = \left\{(n, l, n') \in A | n, n' \in N'\right\} \tag{65}$$

Zwei Knoten n und n' in einem gerichteten Graphen DG sind fest verbunden, wenn ein endlicher Pfad vom Knoten n zu Knoten n' und zurück existiert. Eine Menge von Knoten N' sind fest verbunden, wenn alle Paare von Knoten in N' fest verbunden sind.

Eine fest verbundene Komponente (SCC) ist ein Subgraph bestehend aus einer Menge von Knoten $N' \subseteq N$, sodass

- N' fest verbunden ist.
- Wenn $N'' \subseteq N$ fest verbunden ist und $N' \subseteq N''$ ist, sodass $N' = N''$ gilt.

Die Menge von allen fest verbundenen Komponenten eines gerichteten Graphen DG ist mit SCC_{DG} gekennzeichnet. Eine fest verbundene Komponente ist trivial, wenn sie aus einem einzelnen Knoten und keinen Kanten besteht.

Angenommen, dass DG=(N, A) ein gerichteter Graph mit Kantenbeschriftungen aus einer Menge L ist, dann ist ein fest-verbundener-Komponenten-Graph (SCC Graph) von DG ein gerichteter Graph SG=(N_{SG}, A_{SG}), wobei:

- $N_{SG} = SCC_{DG}$ eine Menge von Knoten ist.
- $A_{SG} = \{((N',A'),l,(N'',A'')) \in N_{SG} \times L \times N_{SG} | \exists (n',l,n'') \in A : n' \in N' \wedge n'' \in N''\}$ eine Menge von Kanten ist.

In Abb. 18 wird die Bildung eines SCC Graphen verdeutlicht.

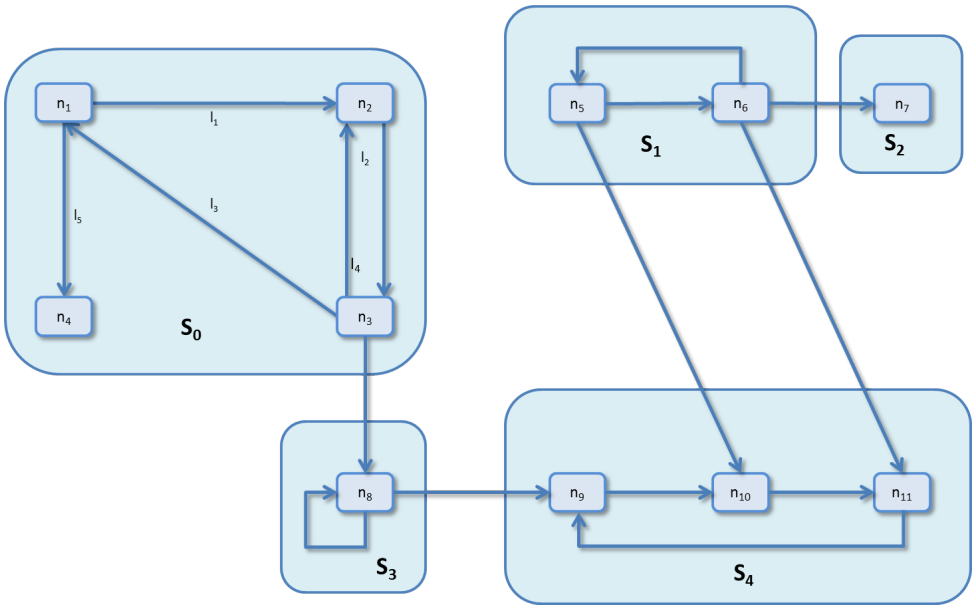

Abb. 18: *Beispiel eines SCC Graphen*

Der Zustandsraum eines Coloured Petri Net ist ein gerichteter Graph SS=(N_{SS}, A_{SS}) mit einer Kantenbeschriftung von BE, wobei:

- $N_{SS} = R(M_0)$ eine Menge von Knoten ist und der Menge erreichbarer Markierungen entspricht.
- $A_{SS} = \left\{ (M,(t,b),M') \in N_{SS} \times BE \times N_{SS} | M \overset{(t,b)}{\rightarrow} M' \right\}$ eine Menge von Kanten ist.

SS ist endlich, wenn und nur wenn N_{SS} und A_{SS} endlich sind.

Die diesem Abschnitt zu Grunde liegenden Informationen stammen aus [46].

4.4 Verhalten von zeitbewerteten Petri-Netzen

Zur Umsetzung eines Zeitverhaltens werden den Marken in „CPN Tools“ Zeitstempel hinzugefügt. Diese Zeitstempel werden durch nichtnegative Integer-Werte realisiert. Der Zeitstempel von Marken kann durch Transitionen und Kanten erhöht werden (siehe Abschnitt 4.1). Der Wert, der dabei dem Zeitstempel hinzuaddiert wird, entspricht der Zeitdauer einer zu modellierenden Operation oder Aktivität. Zudem existiert eine globale

Uhr, welche die Modellzeit definiert. Nachdem einer Marke eine Zeitdauer durch z. B. eine Transition hinzuaddiert wurde, wird diese Marke erst wieder aktiviert, wenn die Modellzeit dem Zeitstempel der Marke entspricht. In „CPN Tools“ wird in jedem Simulationsschritt überprüft, welche Marke den niedrigsten Zeitstempel besitzt. Anschließend springt die Modellzeit zu diesem Wert. Aus diesem Grund ist die Simulationszeit ungleich der zu modellierenden Kommunikationszeit.

5 Systemmodell

5.1 Warum höhere Petri-Netze?

Zur Simulation und Untersuchung des Koexistenzverhaltens von industriellen Funksystemen ist ein Modellansatz erforderlich, welcher die in Kapitel 3 formulierten Anforderungen erfüllt. Höhere Petri-Netze sind dazu in der Lage. Die mathematische Beschreibungsform der höheren Petri-Netze fasst die Eigenschaften zeitbewerteter, kolorierter und stochastischer Petri-Netze zusammen. Gemäß [54] eignen sie sich zur Modellierung ereignisdiskreter Systeme und zur Nachbildung von Nebenläufigkeiten. Wie in [90] beschrieben, hat die Verwendung von Petri-Netzen beim Entwurf von ereignisdiskreten Systemen folgende Vorteile:

- Vollständige und konsistente Beschreibung,
- Graphische Anschauung (verbesserte Fehlererkennung),
- Simulation und Animation (Testmöglichkeit bereits während der Entwurfsphase),
- Mathematische, formale Analyse auf Basis von Graphentheorie und linearer Algebra.

Bei kolorierten Petri-Netzen können den Marken Eigenschaften zugewiesen werden. Dadurch können parameterbehaftete Ereignisse auf Marken abgebildet werden. Zeitbewertete Petri-Netze erlauben die Modellierung von Zeitbedingungen mit Hilfe von zeitbehafteten Transitionen. Die Umsetzung hierarchischer Strukturen ist mit Petri-Netzen ebenfalls möglich ([46]). Letztendlich werden mit höheren Petri-Netzen alle Anforderungen aus Kapitel 3 erfüllt.
Höhere Petri-Netze bieten eine Reihe von Modellelementen wie Plätze, Transitionen und Kanten. Tabelle 6 listet auf, welche zu modellierenden Bestandteile mit welchen Modellelementen abgebildet werden können.

Tab. 6: *Zuordnung von zu modellierenden Bestandteilen zu Petri-Netz-Modellelementen*

Zu modellierender Bestandteil	**Verwendetes Petri-Netz-Modellelement**
Pakete	Marken
Paketeigenschaften (Sequenznummer, Frequenzkanal, Zeitstempel der Paketgenerierung)	Markeneigenschaften
Paketspeicher (Buffer)	Plätze
Fortsetzung auf der nächsten Seite	

Tab. 6: *Zuordnung von zu modellierenden Bestandteilen zu Petri-Netz-Modellelementen – Fortsetzung*

Zu modellierender Bestandteil	Verwendetes Petri-Netz-Modellelement
Zeitverhalten	Zeitstempel an Marken und Delay an Transitionen, Verzögerungen von Marken/Paketen auch durch Bedingungen an Transitionen (Guard) oder durch Struktur des Netzes
Bestimmung zeitlicher Kenngrößen	Zeitstempel der Paketgenerierung als Eigenschaft an Marke und aktuelle Modellzeit nach erfolgreicher Übertragung
Übertragungsmedium	Einzelner Platz, welcher über Kanten mit sämtlichen Verbindungen/Geräten verbunden ist
Eigenschaften des Übertragungsmediums	Durch Markeneigenschaften der Pakete (z. B. Frequenzkanal) beschrieben
Paketkollisionen	Gleichzeitige Belegung des Medium-Platzes mit Marken, welche den gleichen Frequenzkanal als Markeneigenschaft besitzen
Darstellung von Interferenzen	Boolsche Markeneigenschaft „Interferenz“
CCA Check	Überprüfung der Eigenschaft „Frequenzkanal“ der auf dem Medium liegenden Marken
Zeitliche Abfolgen	Wechselnde, sequentielle Aneinanderreihung von Plätzen und Transitionen
Zufallszeiten oder -zahlen	Standard ML-Funktionen an Transitionen (Delay, Guard)
Protokollschichten	Module hierarchischer Petri-Netze

Neben der Eignung für den Entwurf ereignisdiskreter Systeme existieren weitere Vorteile, die durch die Verwendung von Petri-Netzen entstehen:

- Behandlung von Einfluss- und Kenngrößen im Modell,
- durch universelle Modellierbarkeit ist die Ausrichtung der Modelle auf die interessierenden Schwerpunkte und Sachverhalte möglich,
- Koexistenzverhalten von Funksystemen aus Sicht der industriellen Automation (IA) untersuch- und bewertbar,

- Ermittlung IA-spezifischer Kenngrößen,
- neben der direkten Analyse des Modells besteht die Möglichkeit für Performancemessungen und deren Auswertung,
- die Kombination aus kolorierten, zeitbewerteten und stochastischen Petri-Netzen vereinigt sämtliche Funktionalitäten, die zur Koexistenzmodellierung notwendig sind,
- mit zeitbewerteten Transitionen ist die Ermittlung von Zeitkenngrößen, welche die Anforderungen der IA widerspiegeln, möglich,
- Modellierung beispielsweise der zufälligen Wartezeit vor der Übertragung eines Paketes anhand stochastischer Eigenschaften, und
- Verwendung von individuellen Marken, welche individuelle Eigenschaften oder Variablen besitzen können (so kann eine einzelne Marke ein Paket repräsentieren, Eigenschaften können z. B. der Pakettyp oder der Dateninhalt sein).

Zur Umsetzung der Petri-Netz-Modelle wird die Software „CPN Tools“ ([80]) verwendet.

5.2 Übergeordnetes Systemmodell

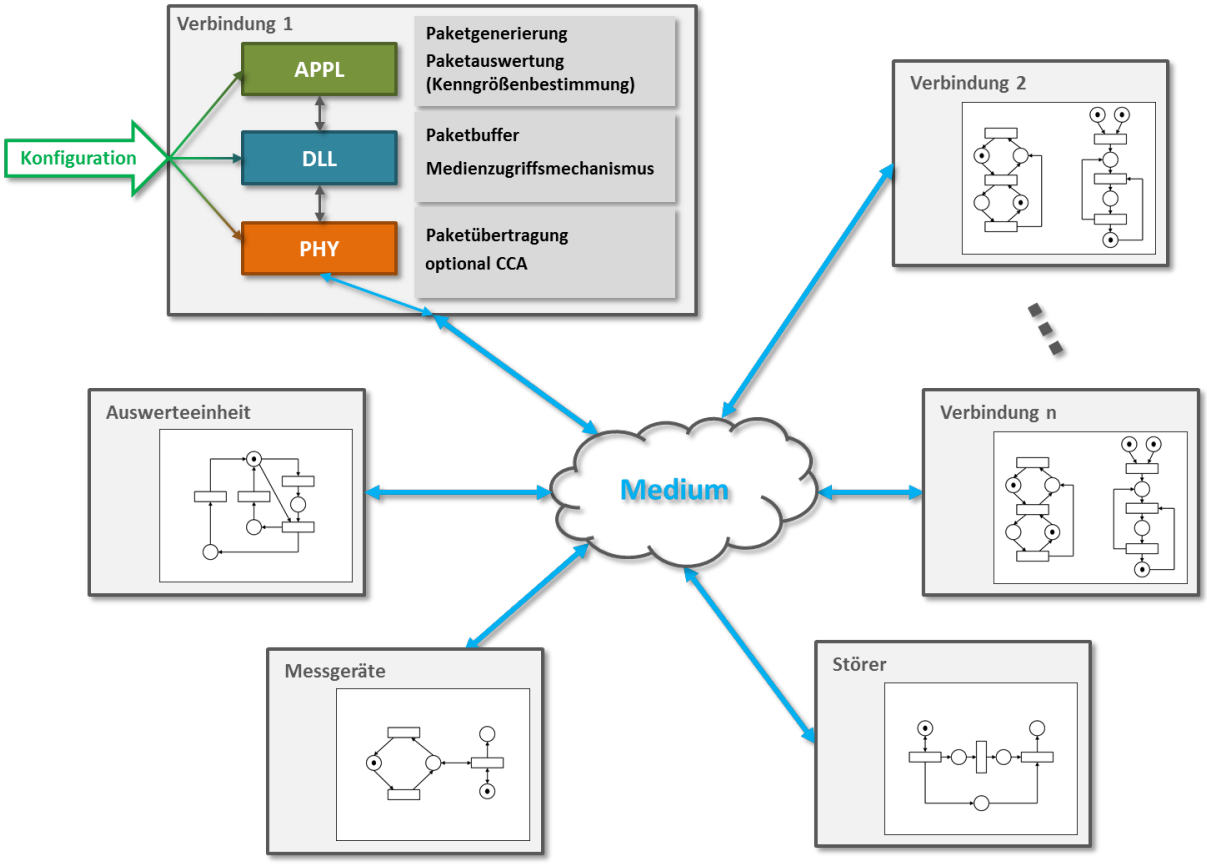

Abb. 19: *Prinzipielle Struktur des Systemmodells*

Zur Modellierung des Koexistenzverhaltens werden Instanzen von Verbindungs- beziehungsweise Gerätemodellen in eine gemeinsame Umgebung eingebracht. Die einzelnen Verbindungen arbeiten so weit wie möglich unabhängig voneinander. Eine gegenseitige Beeinflussung erfolgt über ein gemeinsames Medium nur während eines CCA Checks oder bei einer Kollision von Paketen.

Abb. 19 veranschaulicht die übergeordnete Struktur des Modells. Die Rechtecke sind Module, welche sowohl einzelne Verbindungen als auch Messgeräte, Störer und eine Auswerteeinheit repräsentieren. Jedes Modul enthält ein Petri-Netz-Modell, welches die entsprechenden Funktionalitäten realisiert. Die Petri-Netze der Verbindungen stimmen mit den in der EN 300 328 definierten Anforderungen überein (siehe Abschnitt 2.4.4). Jedes Verbindungsmodell besteht aus drei Schichten, die unterschiedliche Funktionen erfüllen. Die einzelnen Schichten werden in Abschnitt 5.3 genauer erläutert. Jede Verbindung benötigt anhand einer Konfiguration die Vorgabe relevanter Parameter. Die Werte der Parameter bestimmen sowohl das Verhalten der Automatisierungsanwendung als auch des Medienzugriffs.

Sämtliche Module sind mit ein und demselben Medium verbunden. Die zu übertragenden Pakete werden für die Dauer der Medienbelegungszeit auf das Medium gelegt. Wenn CCA Checks durchgeführt werden, wird überprüft, ob sich Pakete auf dem Medium befinden, die denselben Frequenzbereich verwenden. Das Medium ist die einzige Verbindungsstelle zwischen den Modulen. Neben den Verbindungsmodulen existieren auch Module zur Analyse des Koexistenzverhaltens. Hinter dem Modul „Messgeräte“ verbirgt sich in der Regel ein Spektrumanalyzer. Der Spektrumanalyzer zeichnet die Belegung des Mediums in Abhängigkeit von Zeit und Frequenz auf. Die Auswerteeinheit ist notwendig zur Detektierung und Handhabung von Paketkollisionen auf dem Medium. Wenn zwei Pakete denselben Frequenzbereich zur selben Zeit nutzen, werden beide Pakete als gestört markiert, unabhängig von der Dauer der Kollision. Anhand eines weiteren Moduls können Störer nachgebildet werden, welche ebenfalls auf das Medium einwirken. Ein Beispiel für Störer sind Mikrowellenöfen.

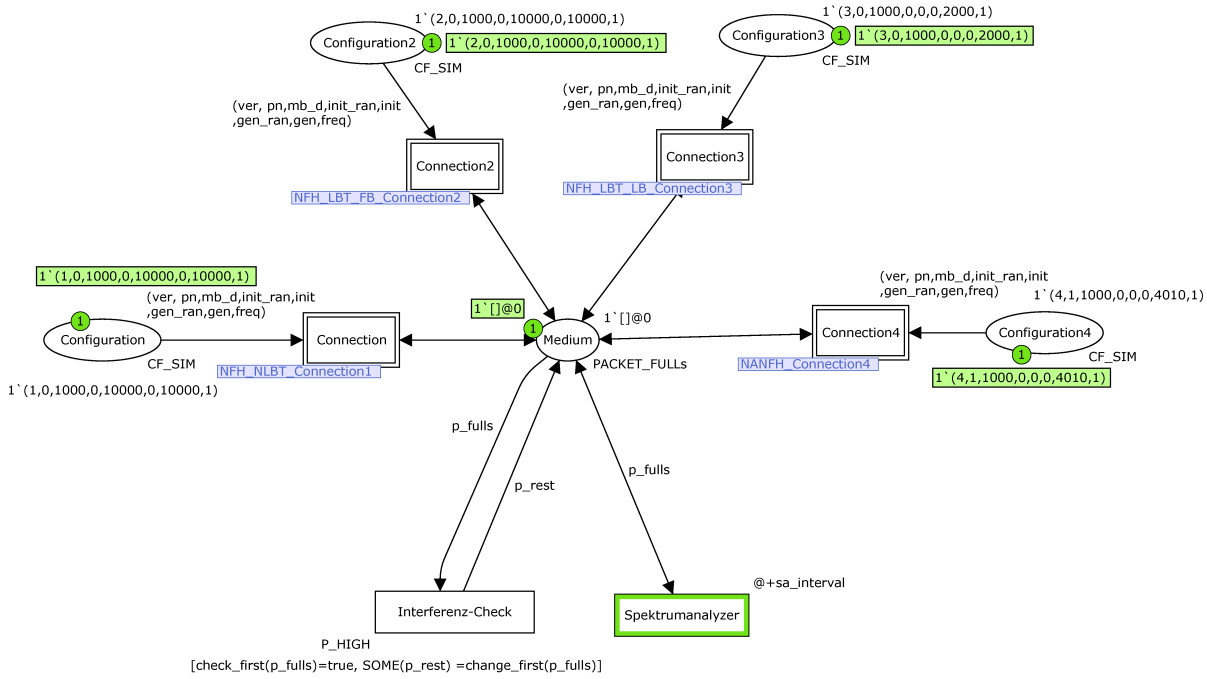

Abb. 20: *Übergeordnetes Systemmodell*

Abbildung 20 zeigt das übergeordnete Systemmodell in Form eines kolorierten Petri-Netzes. Das Medium wird anhand eines einzelnen Platzes modelliert. Sämtliche Module sind über Kanten mit diesem Platz verbunden. Die Verbindungen sind als doppelt gerahmte Rechtecke dargestellt. Jede Verbindung kann Pakete in Form von Marken auf das Medium legen. Die Marken enthalten Eigenschaften wie zum Beispiel die Medienbelegungsdauer, den verwendeten Frequenzbereich und den Status des Paketes („Correct“ / „Interferred“ / „Discarded“). Die Art des Medienzugriffs jeder Verbindungsinstanz wird durch die blaue Markierung unterhalb der Module festgelegt. Jedes Verbindungsmodul ist mit jeweils einem Platz verbunden, welcher eine Konfigurationsmarke trägt. Diese Marke legt die für die einzelne Verbindung relevanten Parameter wie z. B. die Paketanzahl oder die Verbindungsnummer fest. Als Messgerät wird in dem dargestellten Systemmodell ein Spektrumanalyzer verwendet, welcher die Nutzung des Mediums aufzeichnet. Dazu überprüft dieser periodisch den Platz „Medium“ auf aktuell vorhandene Marken. Anhand der Frequenzen, welche diese Marken verwenden, kann der Spektrumanalyzer zurückschließen, welche Teile des Mediums im Augenblick belegt sind. Die Auswerteeinheit wurde mit der Transition „Interferenz-Check“ umgesetzt. Diese feuert jedes Mal, wenn ein neues Paket auf das Medium gelegt wird. Zur Erkennung einer neuen Marke existiert die Markeneigenschaft „first“, die zunächst den Wert true besitzt, und die Funktion „check_first“ (siehe Quelltext 1). Mit der Funktion „change_first“ (siehe Quelltext 2) wird dann überprüft, ob der Frequenzbereich der Marke auch von anderen Marken auf dem Medium verwendet wird. Ist dies der Fall, wechselt die Markeneigenschaft „intf“ der neuen Marke und der Marke, die den identischen Frequenzbereich verwendet, von false auf true. Nach Bearbeitung durch die Transition „Interferenz-Check“ wechselt die Markeneigenschaft „first“ von true auf false.

5.3 Verbindungsmodell

5.3.1 Allgemein

Abb. 21 zeigt das Verbindungsmodell. Dieses ist in drei Ebenen aufgeteilt: Application Layer (APPL), Data Link Layer (DLL) und Physical Layer (PHY). Die einzelnen Ebenen sind als Module umgesetzt, die Petri-Netze enthalten. Der linke Teil der Abbildung zeigt den Platz „Configuration“ und die darauf befindliche Konfigurationsmarke. Beide Objekte sind bereits aus dem übergeordneten Systemmodell bekannt. Die Konfigurationsmarke wird mit Hilfe der Transition „Split_Conf“ auf die nachfolgenden Plätze vervielfältigt. Diese fließen anschließend in die einzelnen Ebenen.

Die genannten Ebenen sind über Plätze miteinander verbunden. Dabei sind Plätze für Marken vorhanden, die Pakete repräsentieren, und Plätze für Marken, welche Events nachbilden. Der Fluss der Marken ist in beide Richtungen zwischen den benachbarten Ebenen möglich. Bisher ist ein Eventfluss (Ergebnis des CCA) nur vom PHY zum DLL notwendig. Bei komplexeren Modellierungen ist dieser auch für die andere Richtung und zwischen den anderen Ebenen denkbar.

Im APPL erfolgt die zeitliche Initialisierung der Verbindung, aber auch die zufällige beziehungsweise zyklische Generierung von Paketen. Weiterhin findet im APPL auch die Auswertung der einzelnen Paketübertragungen durch die Ermittlung von Kenngrößen statt. Der DLL ist für die Umsetzung des Medienzugriffsmechanismus verantwortlich. Für einen möglichen CCA Check liegt der aktuelle Status des Mediums vor. Die DLL-Ebene enthält zusätzlich einen Paketbuffer für eine begrenzte Zwischenspeicherung von Telegrammen. Der PHY legt die Pakete für die Dauer der Medienbelegungszeit auf das Übertragungsmedium und entfernt diese anschließend. Im Physical Layer wird ebenfalls der Status des Mediums bezüglich Belegung erzeugt.

Durch die Aufteilung in Ebenen erfolgt ähnlich dem ISO/OSI-Referenzmodell ein Fluss der Pakete vom APPL über den DLL in den PHY und wieder zurück.

5.3.2 Application Layer

Im Folgenden soll die Umsetzung der einzelnen Funktionen des APPL mit Petri-Netzen dargestellt werden.

Abb. 22 zeigt die Initialisierung der Verbindung und die Generierung von Paketen. Dabei handelt es sich um die Pakete, die mit einem vorherigen, spezifischen Medienzugriffsmechanismus übertragen werden sollen. Die Transition „Conf_Split“ im unteren Teil der Abbildung zerlegt die Konfigurationsmarke in ihre einzelnen Bestandteile. Dabei handelt es sich um die Medienbelegungsdauer und die Anzahl der zu generierenden Pakete. Es wird

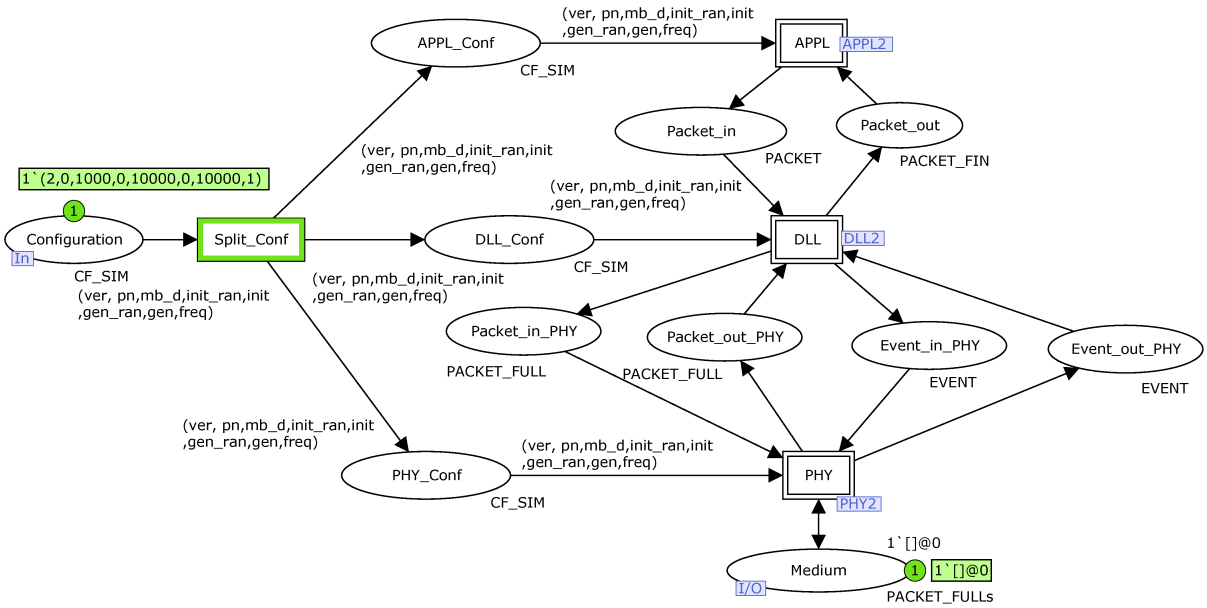

Abb. 21: *Verbindungsmodell*

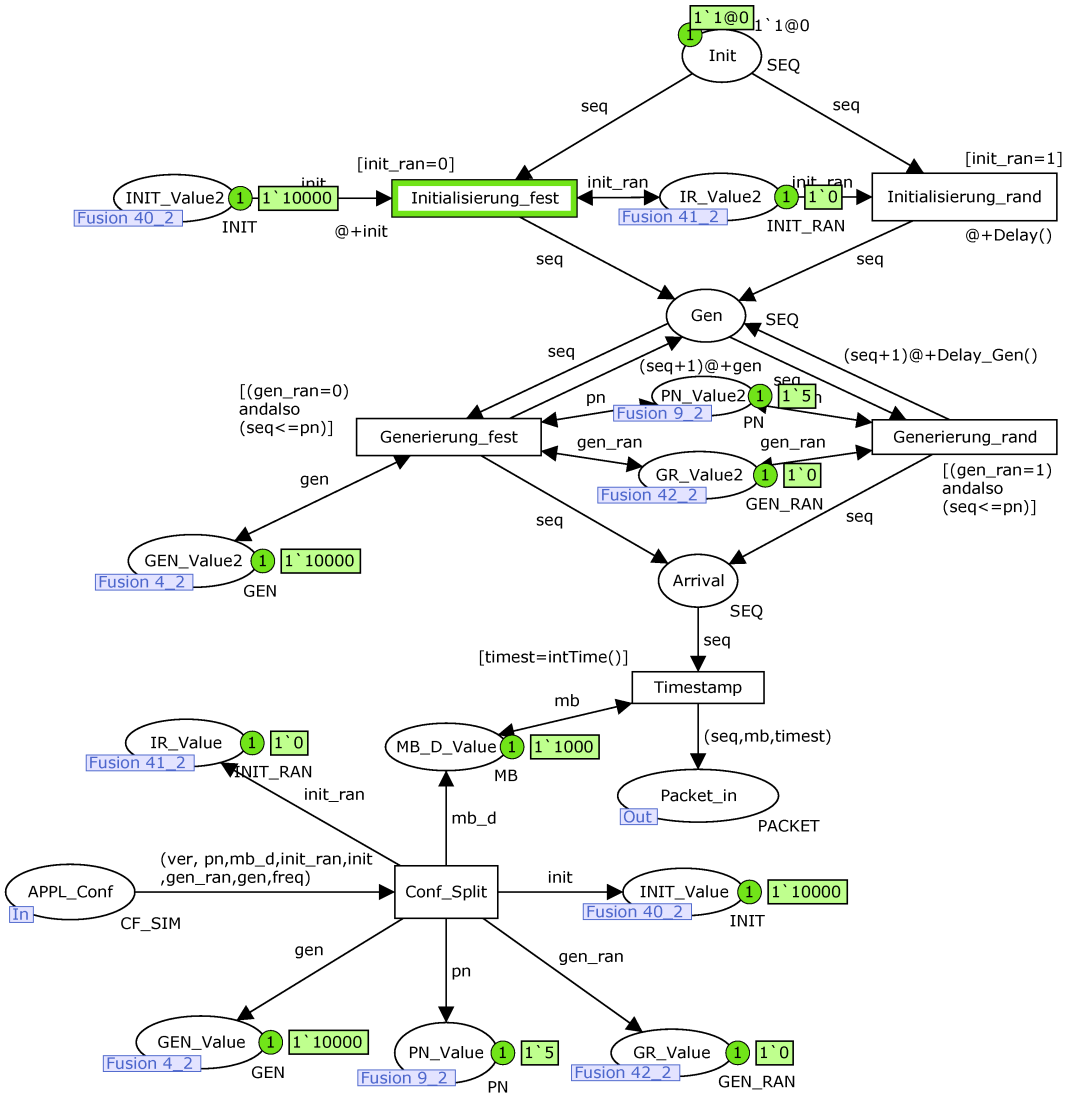

Abb. 22: *Generierung von Paketen im APPL*

festgelegt, ob die Initialisierung und Generierung in einem zufälligen oder festgelegten Zeitabstand erfolgen sollen. Im Falle eines festen Abstandes werden die Zeitwerte ebenfalls über die Konfigurationsmarke angegeben. Über Fusionsplätze gelangen die Marken, welche die einzelnen Konfigurationsparameter beinhalten, zu den Transitionen, die auf die entsprechenden Parameter zurückgreifen müssen. Die Konfigurationsmarke ist notwendig, da in der Entwicklungsumgebung „CPN Tools“ nur globale Variablen und Konstanten verfügbar sind. Die Möglichkeit von Variablen und Konstanten für einzelne Module existiert nicht.

Die Initialisierung beginnt mit der Initialisierungsmarke auf dem Platz „Init“. Von dem Platz der Initialisierungsmarke verlaufen zwei alternative Pfade zum Platz „Gen“. Die Wahl des Pfades erfolgt abhängig vom Konfigurationsparamter „init_ran“. Der linke Pfad über die Transition „Initialisierung_fest“ addiert einen festen Zeitwert, der ebenfalls ein Konfigurationsparameter ist, auf die Marke. Der Pfad auf der rechten Seite addiert einen zufälligen Zeitwert.

Die Generierung erfolgt ähnlich. Der linke Pfad über „Generierung_fest“ addiert einen festen Zeitwert zu der Marke, die zu dem Platz „Gen“ zurückläuft. Im rechten Pfad wird ein zufälliger Zeitwert an die Marke übergeben. Bei der Generierung läuft die Marke immer vom Platz „Gen“ zu einer der Transitionen und wieder zurück. Dabei wird die Sequenznummer hochgezählt und eine neugenerierte Marke wandert auf den Platz „Arrival“. Der linke Pfad ist für eine periodische Generierung verantwortlich, der rechte Pfad für eine zufällige. Die Generierung von neuen Marken wird so lange fortgesetzt, bis die Sequenznummer den Wert von „pn“, die maximale Paketanzahl, überschreitet. Danach verhindert der Guard der Transitionen „Generierung_fest“ und „Generierung_rand“ ein weiteres Feuern dieser.

Mit der Transition „Timestamp“ wird der neugenerierten Marke die aktuelle Systemzeit und ihre Medienbelegungsdauer angefügt. Anschließend wird die Marke dem DLL übergeben.

Die Auswertung jedes Paketes beziehungsweise die Bestimmung der Kenngrößen ist in Abb. 23 dargestellt. Jedes Paket wird über den Platz „Packet_out“ vom DLL zum APPL zurückgegeben. Die Pakete enthalten eine Sequenznummer (seq), einen Zeitstempel ihrer Generierung (timest) und einen Status (status). Der Status ist entweder „Discarded“, „Interferred“ oder „Correct“. Für jedes Paket werden in der Transition „Calculate_Characteristics“ die Kenngrößen Übertragungszeit (tt) und Aktualisierungszeit (ut) ermittelt. Für die Übertragungszeit wird von der aktuellen Simulationszeit der Zeitstempel des Paketes subtrahiert. Zur Berechnung der Aktualisierungszeit ist der Platz „Zeitspeicher“ notwendig. Die Marke auf diesem Platz speichert den Zeitpunkt, an welchem das

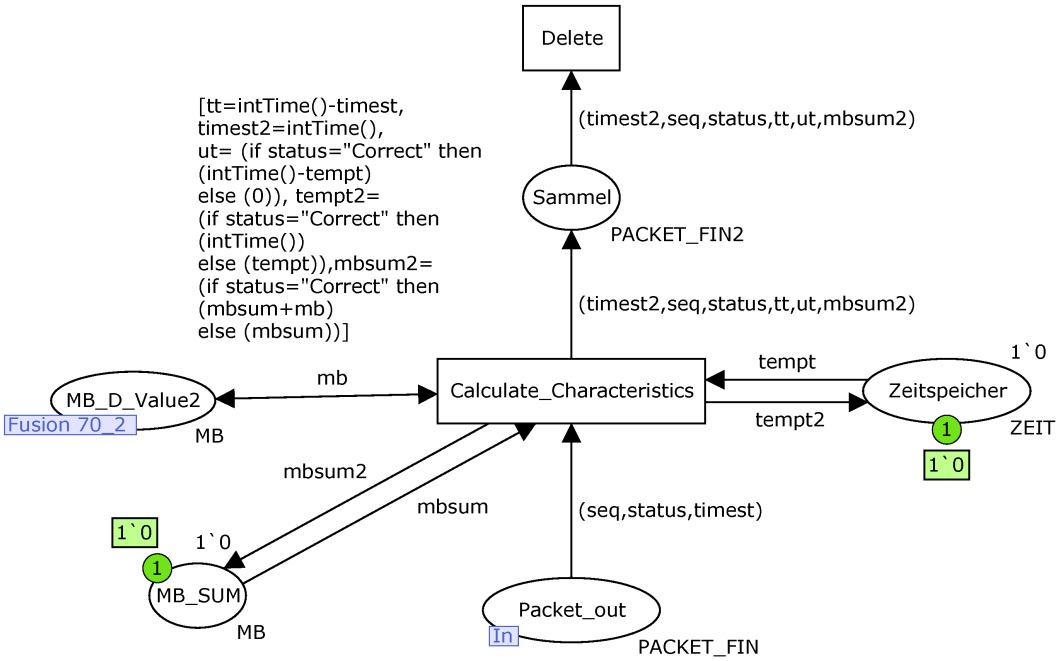

Abb. 23: *Auswerteeinheit im APPL*

letzte korrekte Paket eingetroffen ist. Die Aktualisierungszeit wird nur berechnet und der Zeitspeicher wird nur aktualisiert, wenn ein korrektes Paket durch die Transition fließt. Ansonsten behält der Zeitspeicher seinen alten Wert bei und als Aktualisierungszeit wird der Wert Null angegeben. Weiterhin werden in der Transition der Simulationszeitpunkt, an dem das Paket die Übertragung beendet hat (timest2), und die Gesamtmedienbelegungszeit (mbsum2) festgelegt. Die Aufsummierung dieser wird mit der konfigurierten Medienbelegung „MB_D_Value2“ und nur bei korrekt übertragenen Paketen vorgenommen. Die aktuelle Simulationszeit, die Sequenznummer, der Paketstatus, die Übertragungszeit, die Aktualisierungszeit und die Gesamtmedienbelegungszeit werden in einer Marke auf den Platz „Sammel“ gelegt. Anschließend wird diese Marke mit der Transition „Delete“ gelöscht.

CPN Tools bietet die Möglichkeit, mit Monitorfunktionen Werte aus der Simulation in eine Datei zu schreiben. Dazu wird als Dateiformat CSV genutzt. Ein Monitor zum Sammeln von Daten besteht aus vier Funktionen: Start-Funktion, Predicate-Funktion, Observation-Funktion und Stop-Funktion. Mit der Start-Funktion besteht die Möglichkeit, Daten aus der Initialmarkierung zu sammeln. Sie kann auch genutzt werden, um die erste Zeile der csv-Datei zu erstellen (siehe Quelltext 3). Mit der Predicate-Funktion wird festgelegt, wann Daten gesammelt und geschrieben werden. Im Falle der Auswertung ist dies der Zeitpunkt, an dem die Transition „Calculate_Characteristics“ feuert (siehe Quelltext 4). Die Observation-Funktion bestimmt, welche Daten gesammelt und geschrieben werden. Dabei wird festgelegt, dass der Inhalt der in „Calculate_Characteristics“ erstellten Marke als eine separate Zeile in der csv-Datei abgelegt wird (siehe Quelltext 5). Mit der Stop-Funktion können bei Simulationsende Daten der finalen Markierung verwendet werden. Im Quelltext 6 bildet die Gesamtsimulationszeit die letzte Zeile der csv-Datei.

5.3.3 Data Link Layer

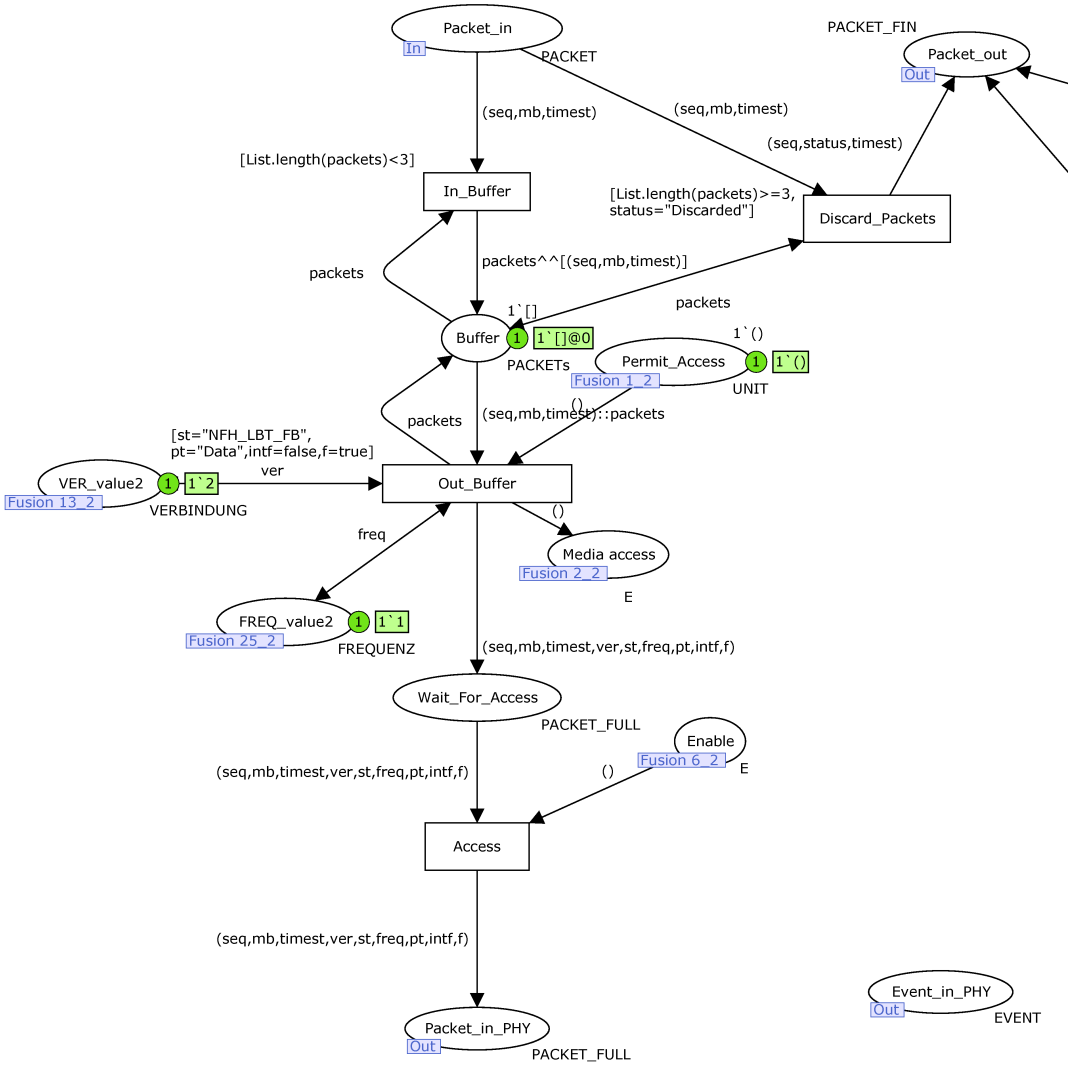

Abb. 24: *Paketpfad mit Buffer im DLL (neuste Paket wird verworfen)*

In diesem Abschnitt werden die Funktionen des DLL veranschaulicht.

Abb. 24 zeigt den Paketfluss vom APPL zum PHY im DLL. Das zuvor generierte Paket liegt auf „Packet_in“. Der Buffer wurde mit Hilfe einer Liste realisiert, welche auf drei Elemente beschränkt ist. Sollte der Buffer schon voll sein, fließt das ankommende Paket über die Transition „Discard_Packets“ zurück zum APPL. Es erhält zusätzlich den Status „Discarded“. Ist der Buffer allerdings nicht vollständig belegt, kann die Marke in der Liste auf dem Platz „Buffer“ abgelegt werden. Der Buffer arbeitet nach dem FIFO-Prinzip. Damit eine Verbindung nur ein Paket auf das Medium legen kann, existiert der Platz „Permit_Access“. Wenn auf diesem Platz eine Marke liegt, wird ein Paket aus dem Buffer genommen und auf den Platz „Wait_For_Access“ gelegt. Der Platz „Media access“, welcher im selben Augenblick belegt wird, initialisiert das Petri-Netz für den Medienzugriff. Sollte der Mechanismus, der für den Zugriff auf das Medium verantwortlich ist, eine Übertragung

erlauben, wird eine Marke auf den Platz „Enable“ gelegt. Die Marke fließt daraufhin zum PHY. Eine neue Marke wird wieder auf den Platz „Permit_Access“ gelegt und damit die Übertragung eines neuen Paketes angestoßen, wenn das übertragende Paket vom PHY zurückgekehrt ist.

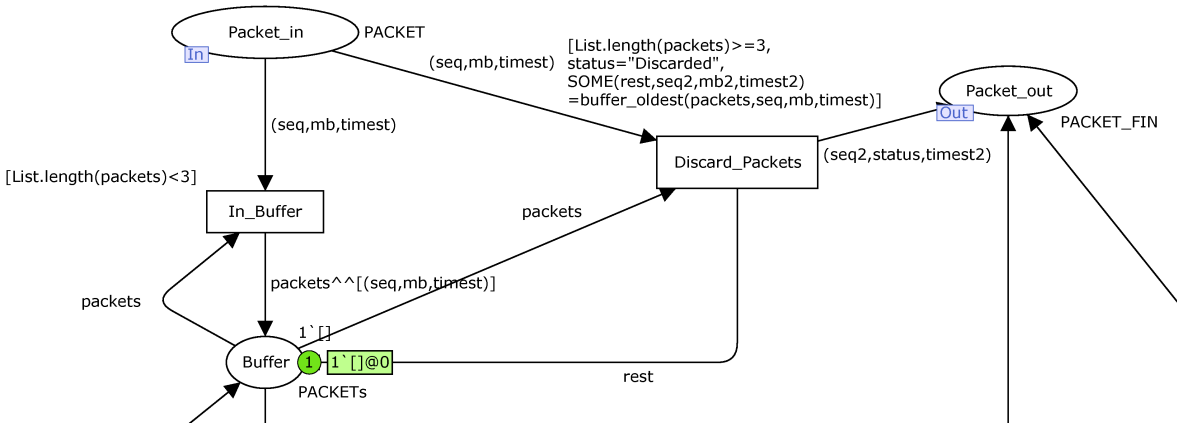

Abb. 25: *Buffer im DLL (älteste Paket wird verworfen)*

Abb. 25 präsentiert einen Buffer, bei dem im Falle eines vollen Speichers das älteste Paket im Buffer verworfen wird. Das vom APPL kommende Paket reiht sich gemäß des FIFO-Prinzips in der letzten Stelle der Warteschlange ein. In Abb. 25 ist wie bereits in Abb. 24 der Buffer auf drei Pakete beschränkt. Umgesetzt wird die Funktionsweise durch die Transition „Discard_Packets“, welche bei einem vollen Buffer anstatt der Transition „In_Buffer“ schaltet, und der Funktion „buffer_oldest“ (siehe Quelltext 7). Beim Schalten der Transition wird das Paket an der ersten Stelle der Bufferliste auf den Platz „Packet_out“ gelegt. Das Paket auf dem Platz „Packet_in“ wird an die nun gekürzte Liste angehängt.
Abb. 26 zeigt den Fluss der Pakete vom PHY zurück zum APPL durch den DLL. Dabei wird das Markenattribut „intf“ ausgewertet. Es wird überprüft, ob bei der Übertragung über das Medium Interferenzen aufgetreten sind oder nicht. Bei Interferenzen wird dem Paket der Status „Interferred“ zugewiesen. Traten keine Interferenzen auf, erhält das Paket den Status „Correct“. Weiterhin wird eine Marke auf den Platz „Permit_Access2“ gelegt, welcher mit dem Paketpfad vom APPL zum PHY verbunden ist. Mit dieser Marke wird das nächste Paket aus dem Buffer genommen und der Medienzugriff gestartet.

Abb. 27 zeigt die Umsetzung des Medienzustandes im DLL. Dabei hat das Medium für jeden Kanal zwei Zustände, welche durch die beiden Plätze „Media free“ und „Media occupied“ dargestellt werden. Der DLL erhält von dem PHY zwei mögliche Eventarten: „Free“ und „Busy“. Wenn der bisherige Zustand des Mediums im DLL frei ist und vom PHY das Event geliefert wird, dass das Medium belegt ist, fließt eine Marke von „Media free“ zu „Media occupied“. Das gleiche Verhalten erfolgt in andere Richtung, wenn das Event „Free“ eintritt. Die Transitionen „Free_Free“ und „Busy_Busy“ existieren, falls zwei

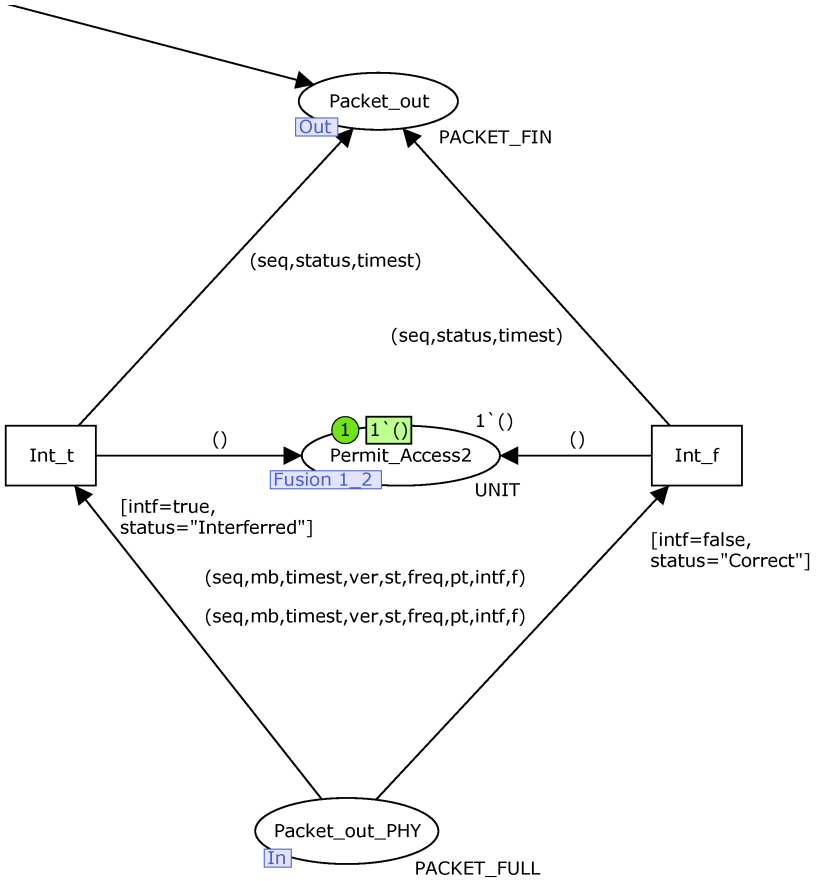

Abb. 26: *Rückfluss der Pakete im DLL*

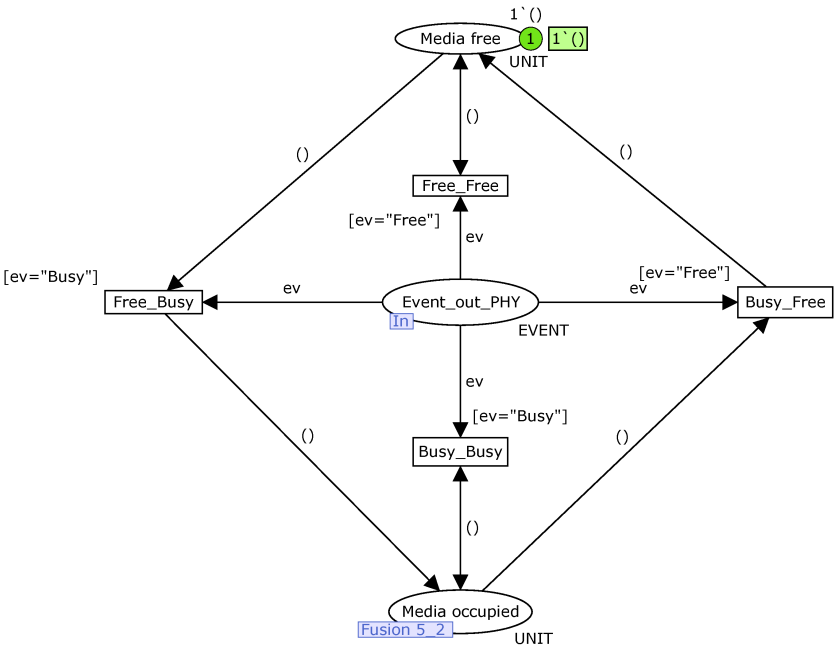

Abb. 27: *Darstellung des Medienzustandes im DLL*

aufeinanderfolgende Events gleich sind. Die Marken dieser Events müssen zerstört werden, ändern den Medienzustand im DLL allerdings nicht. „Media free“ und „Media occupied“ sind als Fusionsplätze direkt mit dem Petri-Netz verbunden, das für den Medienzugriff verantwortlich ist.

5.3.4 Physical Layer

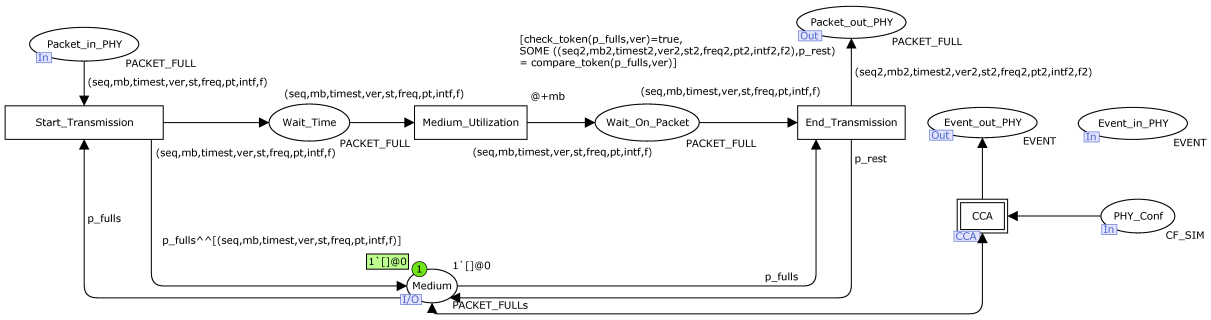

Abb. 28: *Das Modell des Physical Layers*

Dieser Abschnitt erläutert die Funktionsweise des PHY.

Abb. 28 zeigt das Modell des Physical Layers. Pakete aus dem DLL gelangen über den Platz „Packet_in_PHY“ in den PHY. Diese werden über die Transition „Start_Transmission“ auf das Medium gelegt. Das Medium ist wie schon der Buffer über eine Liste realisiert. Parallel zum Medium existiert ein Pfad bestehend aus den Plätzen „Wait_Time“ und „Wait_On_Packet“ sowie der Transition „Medium_Utilization“. Dieser Strang bildet die Zeitdauer der Medienbelegung nach. Wenn die Marke, welche dieselben Informationen wie die Marke auf dem Medium enthält, diesen Strang durchlaufen hat, wird die Transition „End_Transmission“ aktiviert. Hier wird mit der Funktion „check_token“ (siehe Quelltext 8) überprüft, ob die Liste des Mediums ein Paket beinhaltet, welches die gleiche Verbindungsnummer wie die Marke, die sich auf dem Platz „Wait_On_Packet“ befindet, aufweist. Mit der Funktion „compare_token“ (siehe Quelltext 9) wird das entsprechende Paket der Liste entnommen und über den Platz „Packet_out_PHY“ dem DLL übergeben. Ein weiterer Bestandteil des PHY ist der CCA Check, welcher in einem separaten Modul zusammengefasst ist.

Der CCA Check (siehe Abb. 29) wird mit der konfigurierten Frequenz über den Platz „PHY_Conf“ und der Transition „Split_Conf“ initialisiert. Daraufhin liegt eine Marke, die den Frequenzwert enthält, immer auf einen der beiden Plätze „Channel free“ und „Channel busy“. Bei jedem Übergang zwischen den Plätzen wird ein Event „Busy“ oder ein Event „Free“ über den Platz „Event_out_PHY“ an den DLL gesendet. Wenn sich die Marke auf dem Platz „Channel free“ befindet, wird mit der Funktion „check_same_freq“ (siehe Quelltext 10) überprüft, ob Pakete mit derselben Frequenz auf das Medium gelegt werden. In diesem Fall fließt die Marke auf den Platz „Channel busy“. Wenn die Marke

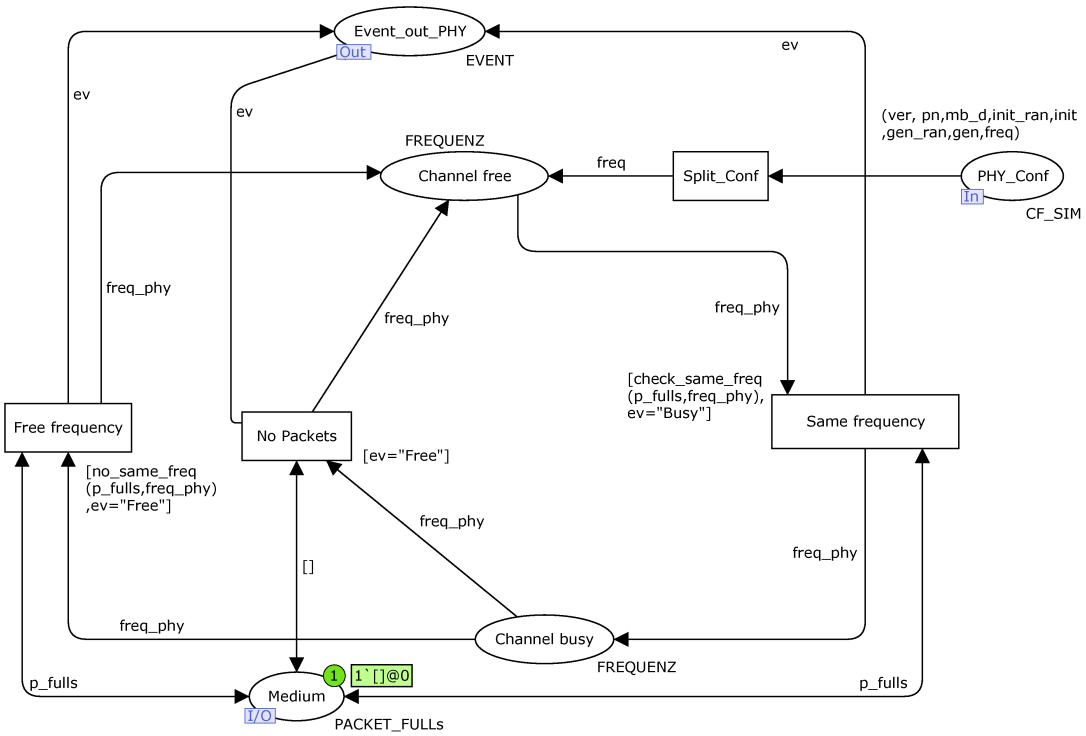

Abb. 29: *Das Modul CCA im Physical Layer*

auf „Channel busy“ liegt, erfolgt der Übergang zurück, wenn die Liste des Mediums leer ist (Transition „No Packets“) oder sich keine Pakete mit der konfigurierten Frequenz auf dem Medium befinden (Transition „Free frequency“). Bei der Transition „Free frequency“ wird die Funktion „no_same_freq“ (siehe Quelltext 11) im Guard ausgeführt.

5.4 Zusammenfassung

In diesem Abschnitt wurden die Vorteile von Petri-Netzen beschrieben und gezeigt, dass mit höheren Petri-Netzen die Anforderungen aus Kapitel 3 erfüllt werden können. Weiterhin wurde eine tabellarische Zuordnung von zu modellierenden Bestandteilen zu Petri-Netz-Modellelementen vorgenommen. Ebenfalls wurden in diesem Kapitel sowohl das übergeordnete Systemmodell als auch das Verbindungsmodell, bestehend aus den drei Ebenen Application Layer, Data Link layer und Physical Layer, eingeführt. In den folgenden Kapiteln werden die Modellteile zur Umsetzung der verschiedenen Medienzugriffsmechanismen in den Data Link Layer integriert. Die anderen Systemmodellelemente und Ebenen des Verbindungsmodells bleiben weitestgehend unverändert.

6 ETSI EN 300 328

6.1 Allgemeines

Das European Telecommunications Standards Institute (ETSI) ist ein gemeinnütziges Institut zur Schaffung europaweit einheitlicher Standards im Bereich der Telekommunikation ([44]). Im Gegensatz zur Conférence Européenne des Administrations des Postes et des Télécommunications (CEPT), die sich mit der Frequenzverwaltung und Regulierung befasst, konzentriert sich die ETSI auf die technischen Fragestellungen sämtlicher Anwendungsbereiche der Telekommunikation (siehe Abbildung 30).

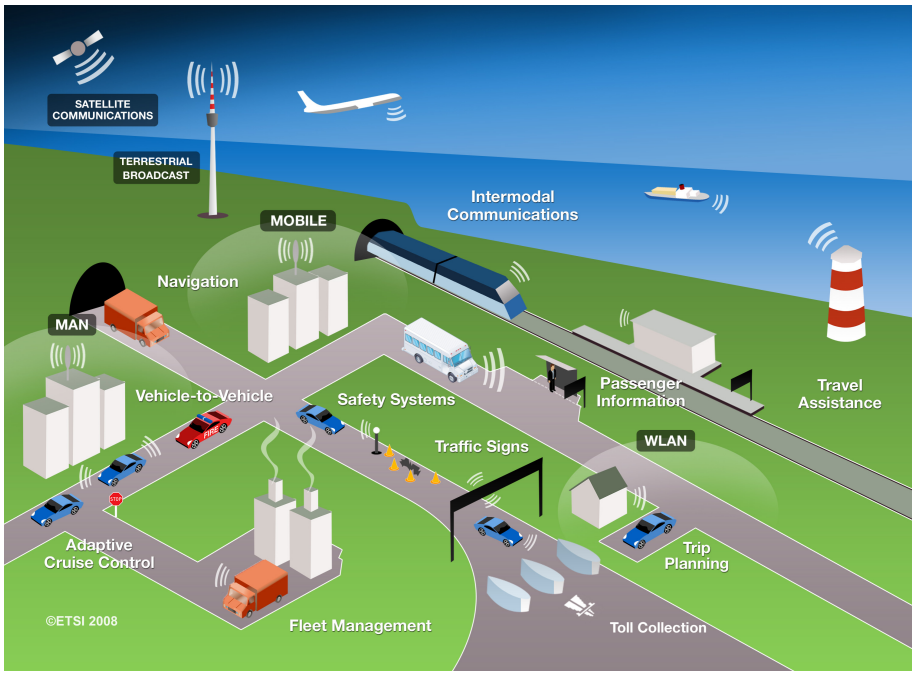

Abb. 30: *Anwendungsfelder der ETSI ([44])*

Die ETSI Arbeitsgruppe ERM-TG11 ist für die Erarbeitung des Standards EN 300 328 verantwortlich ([85]). Dieses Dokument bezieht sich auf Wide-Band-Datenübertragungssysteme, welche eine maximale Sendeleistung von 100 mW besitzen. Das betrachtete Frequenzband ist beschränkt auf das ISM-Band von 2,4 bis 2,4835 GHz. Der aufkommende Standard definiert Anforderungen an den Spektrenzugriff, um eine gemeinsame Spektrennutzung mit anderen Geräten zu erleichtern. Zum Beispiel basieren diese Geräte auf den Spezifikationen IEEE 802.11, IEEE 802.15.1 und IEEE 802.15.4. Systeme mit den genannten Technologien können in festen, mobilen oder portablen Anwendungen eingesetzt werden, zum Beispiel in PCs, Handhelds, APs und Set-Top Boxen.

In der Version 1.8.1 der EN 300 328 werden die Anforderungen an sieben Systemtypen spezifiziert:

- Frequency Hopping (FH) non-adaptive (NA) Systems (FH-NA)

- Adaptive Frequency Hopping (AFH) Systems, welche LBT basiertes Detect-And-Avoid (DAA) verwenden (AFH-LBT)
- AFH Systems, welche non-LBT basiertes DAA verwenden (AFH-NLBT)
- Non-FH NA Systems (NFH-NA)
- Non-FH Systems, welche non-LBT basiertes DAA verwenden (NFH-NLBT)
- Frame based (FB) non-FH Systems, welche LBT basiertes DAA verwenden (NFH-FB)
- Load based (LB) non-FH Systems, welche LBT basiertes DAA verwenden (NFH-LB)

Die Systeme in der Auflistung unterscheiden sich in FH und non-FH, aber auch in adaptive und nicht-adaptive. Frequenzhopping-Systeme wechseln wiederholt während der Funkübertragung den verwendeten Frequenzkanal entsprechend eines Hoppingschemas. Dagegen werden mit „non-FH“ alle die Systeme bezeichnet, die Wide Band Modulation verwenden und keinen Kanalwechsel vornehmen. Nicht-adaptive Systeme haben keine automatischen Mechanismen zur Erkennung anderer Teilnehmer. Allerdings sind sie dadurch an bestimmte Beschränkungen bezüglich der Mediennutzung gebunden. Adaptive Systeme identifizieren die Kanäle, die bereits von anderen Systemen benutzt werden. FH-Systeme können beispielsweise den Hoppingalgorithmus anpassen. Dabei werden belegte Kanäle von der Liste verfügbarer Kanäle gelöscht.

Das Ziel der EN 300 328 ist die Sicherstellung einer fairen Medienaufteilung zwischen den genannten Systemen. Wenn die Anzahl an Komponenten und Systemen ansteigt, sollte die Beeinträchtigung des Kommunikationsverhaltens für alle Teilnehmer, die auf das Medium zugreifen, gleich sein. Im Folgenden werden die Medienzugriffsmodelle für die non-FH Systeme beschrieben. Anschließend wird die Verifikation der Modelle näher erläutert. Das Kapitel wird mit ausgewählten Simulationsergebnissen und deren Auswertung beendet.

6.2 Medienzugriffsverhalten

6.2.1 Non-FH non-adaptive systems

Anforderungen

Abbildung 31 zeigt die Medienzugriffsanforderungen eines non-adaptive non-FH Systems. Es betreibt kein Frequency Hopping. Es ist auch kein adaptiver Mechanismus vorhanden. Die Teilnehmer eines NFH-NA Systems können daher ohne vorherige Prüfung des Mediums darauf zugreifen. Dabei definiert die Sequenzdauer (t_{seq}) die Zeitdauer, während der einzelne oder mehrere Übertragungen auftreten können. Nach der abgelaufenen Sequenzdauer muss die Gap-Zeit (t_{gap}) folgen. Während dieser Zeit ist keine Übertragung

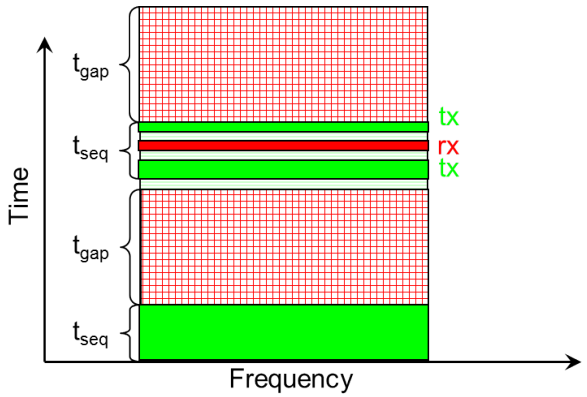

Abb. 31: *Anforderungen an NFH-NA Systeme*

des Teilnehmers erlaubt. Die maximale Sequenzdauer und die minimale Gap-Zeit müssen gleich groß sein und im Intervall zwischen 3,5 und 10 ms liegen. Jedoch ist der Medienzugriff nicht nur durch die Aufteilung in Sequenzdauer und Gap-Zeit begrenzt. Eine weitere Restriktion erfolgt durch den Medium Utilization (MU) Faktor, welcher die verwendete Menge an Zeit und Leistung bei nicht-adaptiven Geräten angibt. MU ist wie folgt beschrieben:

$$MU = \frac{P}{100mW} \cdot DC[\%] \tag{66}$$

wobei P die Sendeleistung (e.i.r.p.) in mW und DC der Duty Cycle ist. Bei NA-NFH Systemen ist der Duty Cycle als Verhältnis aus gesamter Einschaltzeit des Transmitters zu einem Betrachtungszeitraum von 1 s definiert. Der maximale MU Faktor von diesen Geräten wird auf 10 % beschränkt.

Modellbeschreibung

Abbildung 32 zeigt das Modell, welches den Medienzugriff eines NFH-NA Systems umsetzt. In der Ausgangssituation liegt eine Marke auf „Access permitted“. Wenn für ein neues Paket der Medienzugriff durchgeführt werden soll, wird eine Marke auf den Platz „Media access2“ gelegt. Die Transition „Permit transmission“ schaltet. Es werden Marken auf den Plätzen „Tx in t_seq“, „t_seq started“ und „Enable2“ erzeugt. Mit „Enable2“ bekommt das aktuelle Paket die Freigabe für die Übertragung. Über den Strang „t_seq started“ - „Timer for t_seq“ - „t_seq expired“ wird die Sequenzzeit abzüglich der Medienbelegung realisiert. Die Medienbelegung wird abgezogen, damit die Sequenzzeit nicht durch ein Paket, welches kurz vor Ablauf der Sequenzzeit auf das Medium gelegt wird, überschritten wird. Während eine Marke den eben genannten Strang durchläuft, liegt eine Marke auf dem Platz „Tx in t_seq“. Sollte nun ein Paket Medienzugriff beanspruchen (Marke auf „Media access2“) schaltet die Transition „Immediate transmission“ und die Übertragung des Paketes wird gestartet (Marke auf „Enable2“). Das Schalten der Transition „Stop new transmissions“ beendet die sofortige Übertragung neuer Pakete. Anschließend wird die Medienbelegungszeit („Wait MBD“, „Timer for MBD“) und die Gap-Zeit („Wait Gap“, „Timer for Gap“) abgewartet.

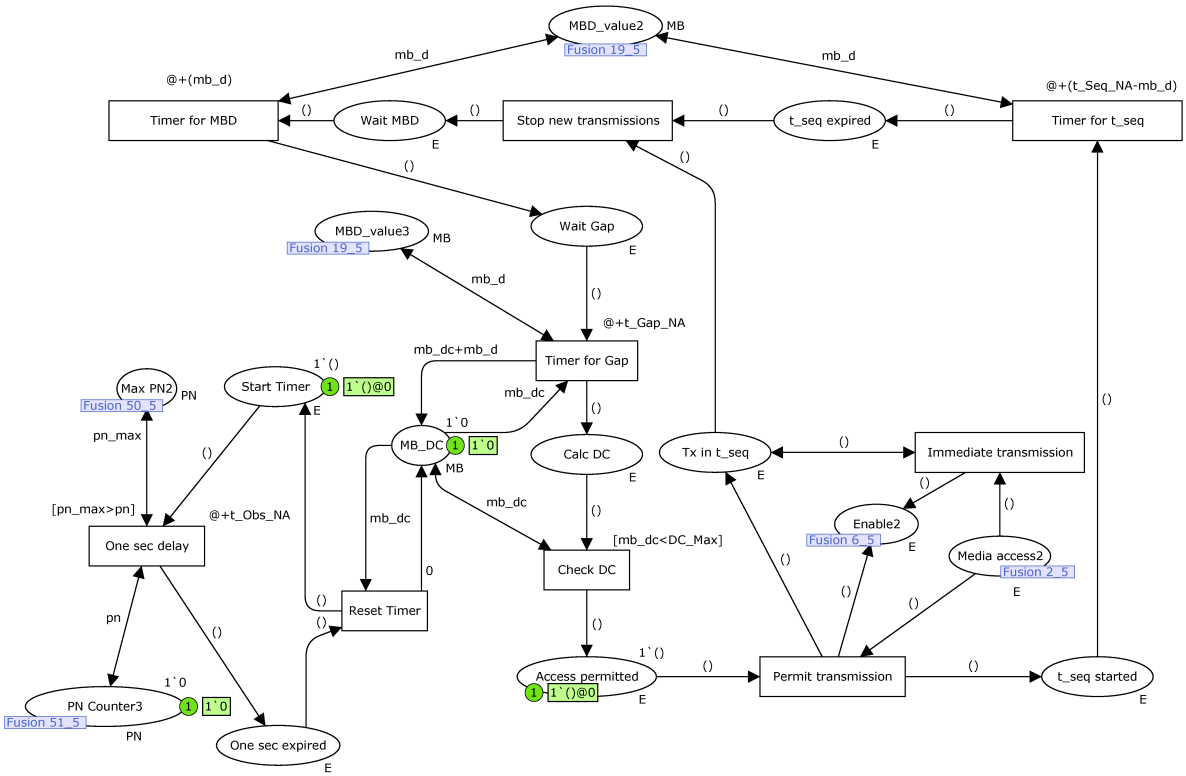

Abb. 32: *Modell zum Medienzugriff eines NFH-NA Systems*

Mit dem Kreislauf, bestehend aus den Plätzen „Start Timer“ und „One sec expired“, sowie den Transitionen „Reset Timer“ und „One sec delay“ wird der Beobachtungszeitraum von 1 s umgesetzt. Die Transition „One sec delay“ schaltet nur, wenn noch nicht alle Pakete der Applikation übertragen wurden. Ansonsten würde durch den Kreislauf die Simulation nicht selbstständig enden. Beim Schalten der Transition „Timer for Gap“ werden die Medienbelegungszeiten der durchlaufenden Pakete aufaddiert und die Summe in der Marke des Platzes „MB_DC“ gespeichert. Beim Schalten der Transition „Reset Timer“ wird der Wert der Marke auf „MB_DC“ auf Null zurückgesetzt. Mit Hilfe der Transition „Check DC“ und dem Wert der Marke auf „MB_DC“ wird überprüft, ob der Duty Cycle im aktuellen Beobachtungszeitraum den Wert von 10 % bereits überschritten hat oder weitere Übertragungen erlaubt sind.

6.2.2 Non-FH systems using non-LBT based DAA

Anforderungen

Bei „Non-FH systems using non-LBT based DAA“ wird ein gegebener Kanal als „unbenutzbar“ eingestuft, wenn Interferenzen nach der Übertragung in diesem Kanal ermittelt wurden. Solche Geräte sollen einer Reihe von Anforderungen entsprechen (siehe Abbildung 33). Während des Normalbetriebs sollen Geräte die Präsenz von Signalen auf den aktuell verwendeten Kanal bewerten. Wenn die Präsenz eines Signals durch die Überschreitung einer Detektierungsschwelle festgestellt wurde, dann wird dieser als „unbenutzbar“

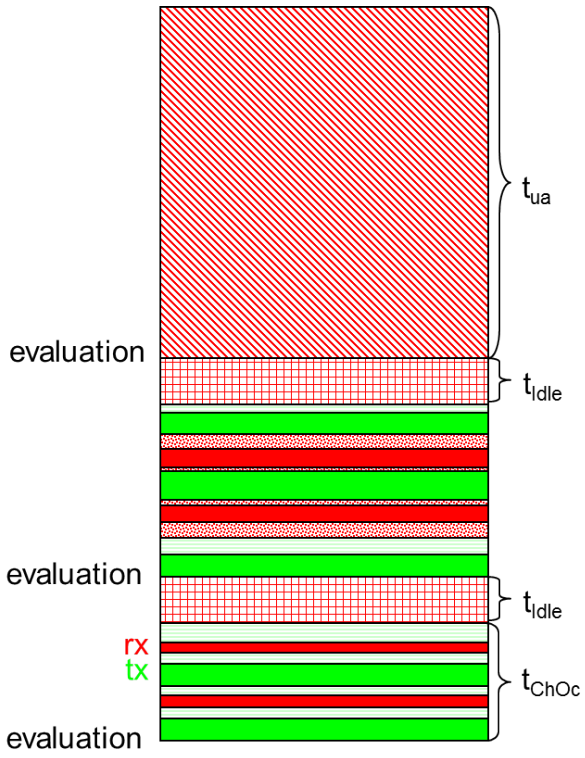

Abb. 33: *Anforderungen an NFH-NLBT Systeme*

markiert. Die Zeit, die der entsprechende Kanal gesperrt ist, wird als „Unavailable Time“ t_{ua} bezeichnet. Die Sperrung des Kanals soll mindestens eine Sekunde betragen. Nach Ablauf dieser Zeit kann der Kanal wieder als benutzbar betrachtet werden.

Innerhalb der gesamten Kanalbelegungszeit t_{ChOc}, welche kleiner als 40 ms sein soll, kann das Gerät ungehindert auf das Medium zugreifen. Dieser Zeit folgt eine Idle-Period t_{Idle}. Die Dauer der Idle-Period beträgt:

$$t_{Idle} \geq max(100\mu s, 5\% t_{ChOc}) \tag{67}$$

Innerhalb der Idle-Period erfolgt kein Zugriff auf das Medium. Nach dieser Zeit wird mit der bereits beschriebenen Kanalbewertung fortgesetzt.

Modellbeschreibung

Abbildung 34 stellt das Modell zum Medienzugriff eines NFH-NLBT Systems dar. Es arbeitet ähnlich dem Modell des NFH-NA Systems. Im Ausgangszustand liegt eine Marke auf dem Platz „Allowed“. Vor der Übertragung eines neuen Paketes wird zum Starten des Medienzugriffs eine Marke auf „Media access2“ gelegt. Damit kann die Transition „Permit transmission“ schalten. Das neue Paket erhält durch die Belegung des Platzes „Enable2“ mit einer Marke das Recht, sofort übertragen zu werden. Es werden weitere Marken auf „ChOc“ und „Inside ChOc“ gelegt. Über den Strang „ChOc2“ - „Wait ChOc“ - „ChOc off“ wird die Kanalbelegungszeit abzüglich der Medienbelegungszeit realisiert. In dieser Zeit ermöglicht die Marke auf „Inside ChOc“ die sofortige Übertragung weiterer Pakete. Sobald

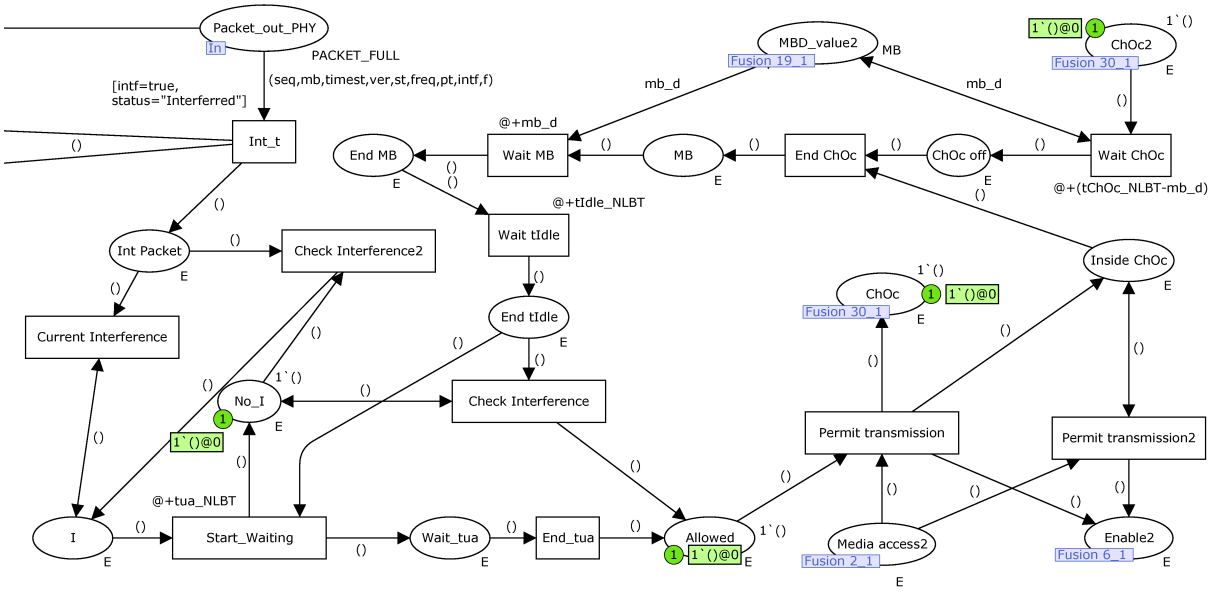

Abb. 34: *Modell zum Medienzugriff eines NFH-NLBT Systems*

eine neue Marke auf „Media access2“ gelegt wird, kann die Transition „Permit transmission2“ schalten und eine neue Marke fließt nach „Enable2“. Nachdem die Transition „End ChOc“ geschaltet hat, ist dies nicht mehr möglich. Es wird dann die Medienbelegungszeit („MB“, „Wait MB“ und „End MB“) und die Idle-Period („Wait tIdle“ und „End tIdle“) abgewartet, in denen neue Pakete keine Freigabe erhalten.

Das Modell für den NFH-NLBT Medienzugriff ist mit dem Strang verbunden, der fertig übertragende Pakete von dem PHY zur Applikation leitet. Mit der Transition „Int_t“ wird ausgewertet, ob bei diesen Paketen Interferenzen auftraten. Wurden Interferenzen festgestellt, wird eine Marke auf „Int Packet“ gelegt. Im Ausgangszustand liegt eine Marke auf „No_I“. Mit der Marke auf „Int Packet“ fließt diese nun über die Transition „Check Interference2“ nach „I“. Sollten innerhalb der Kanalbelegungszeit weitere Pakete Interferenzen aufweisen, schaltet die Transition „Current Interference“ und neue Marken auf „Int Packet“ würden zerstört werden. Nach Ablauf der Idle-Period wird überprüft, ob Interferenzen innerhalb der Kanalbelegungszeit aufgetreten sind. Wenn keine Interferenzen festgestellt wurden, liegt eine Marke auf „No_I“ und die Transition „Check Interference“ schaltet. Neue Paketübertragungen sind sofort möglich. Sind Interferenzen aufgetreten, liegt eine Marke auf „I“ und der Strang „Start_Waiting“ - „Wait_tua“ - „End_tua“ realisiert die „Unavailable Time“. Nach der Interferenzüberprüfung liegt eine Marke auf „No_I“ und keine auf „I“.

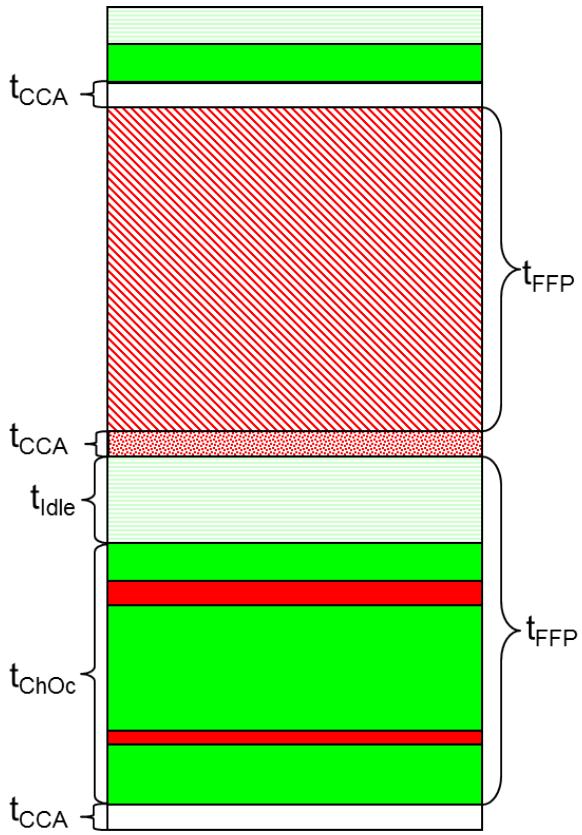

Abb. 35: *Anforderungen an NFH-FB Systeme*

6.2.3 Non-FH systems frame based devices using LBT based DAA

Anforderungen

„Frame based“ und „Load based“ Systeme verwenden beide LBT und sind sich sehr ähnlich. Es ist ihnen erlaubt, dynamisch zwischen den beiden Betriebsarten zu wechseln.

Das FB System soll vor einer Übertragung einen Clear Channel Assessment (CCA) Check, welcher den Energielevel des verwendeten Frequenzbereiches überprüft, durchführen. Das Gerät soll den entsprechenden Kanal für die Dauer der „CCA Observation Time“ (t_{CCA}) beobachten, welche nicht kleiner als 20 μs sein darf. Der Kanal wird als belegt betrachtet, wenn der Energielevel des Kanals den vorgegebenen Grenzwert überschreitet. Wenn das Gerät den Kanal als frei erkennt, kann sofort der Übertragungsvorgang gestartet werden. Erkennt das Gerät allerdings den Kanal als belegt, dann sollen auf diesem Kanal während der nächsten Fixed Frame Period keine Übertragungen initiiert werden. Die Fixed Frame Period t_{FFP} ist dabei die Summe aus Kanalbelegungszeit t_{ChOc} und der darauffolgenden Idle-Zeit t_{Idle} (siehe Abbildung 35). Dem Gerät ist es allerdings auch erlaubt, in den nicht-adaptiven Modus zu wechseln und mit der Übertragung auf diesem Kanal fortzusetzen. Jedoch müssen die Anforderungen an nicht-adaptive Geräte eingehalten werden.

Die maximale Kanalbelegungszeit soll im Bereich zwischen 1 und 10 ms liegen. Die Idle-

Zeit soll mindestens 5 % der maximalen Kanalbelegungszeit, die vom Gerät in der aktuellen Fixed Frame Period verwendet wurde, betragen.

Ein Gerät kann nach dem korrekten Empfang eines Paketes, welches auch für dieses Gerät bestimmt war, den CCA Check auslassen und sofort mit der Übertragung von Management und Control Paketen fortfahren (z. B. sind ACK und Block ACK Pakete erlaubt, aber Datenpakete nicht). Ein fortlaufendes Senden solcher Übertragungen durch das Gerät ohne einen erneuten CCA Check soll die maximale Kanalbelegungszeit nicht überschreiten.

Modellbeschreibung

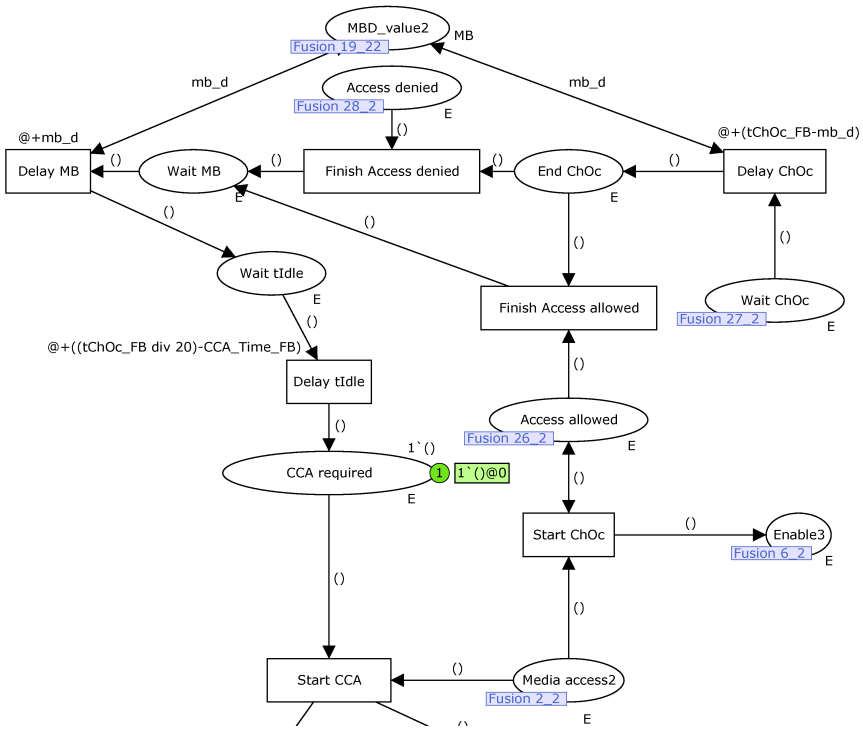

Abb. 36: *Modell zum Medienzugriff eines NFH-FB Systems (Teil 1)*

Abbildung 36 und Abbildung 37 zeigen das Modell zum Medienzugriff eines NFH-FB Systems. Abbildung 37 zeigt die Umsetzung des CCA Checks. Der Modellteil in Abbildung 36 realisiert die Kanalbelegungszeit und Idle-Period.

Der Medienzugriff vor der Übertragung eines neuen Paketes wird mit einer Marke auf „Media access2“ initiiert. Die Dauer des dafür notwendigen CCA Checks wird mit dem Strang „CCA“ - „Wait CCA“ - „CCA finished“ realisiert (siehe Abbildung 37). Gleichzeitig liegt eine Marke auf dem Platz „No Interference“. Ist in der Zeit des CCA Checks das Medium belegt, schaltet die Transition „Check Interference“ und die Marke fließt von „No Interference“ nach „Interference“. Nach Ablauf der CCA Zeit wird ausgewertet, auf welcher der beiden Stellen eine Marke liegt. Wurde keine Belegung festgestellt, schaltet die

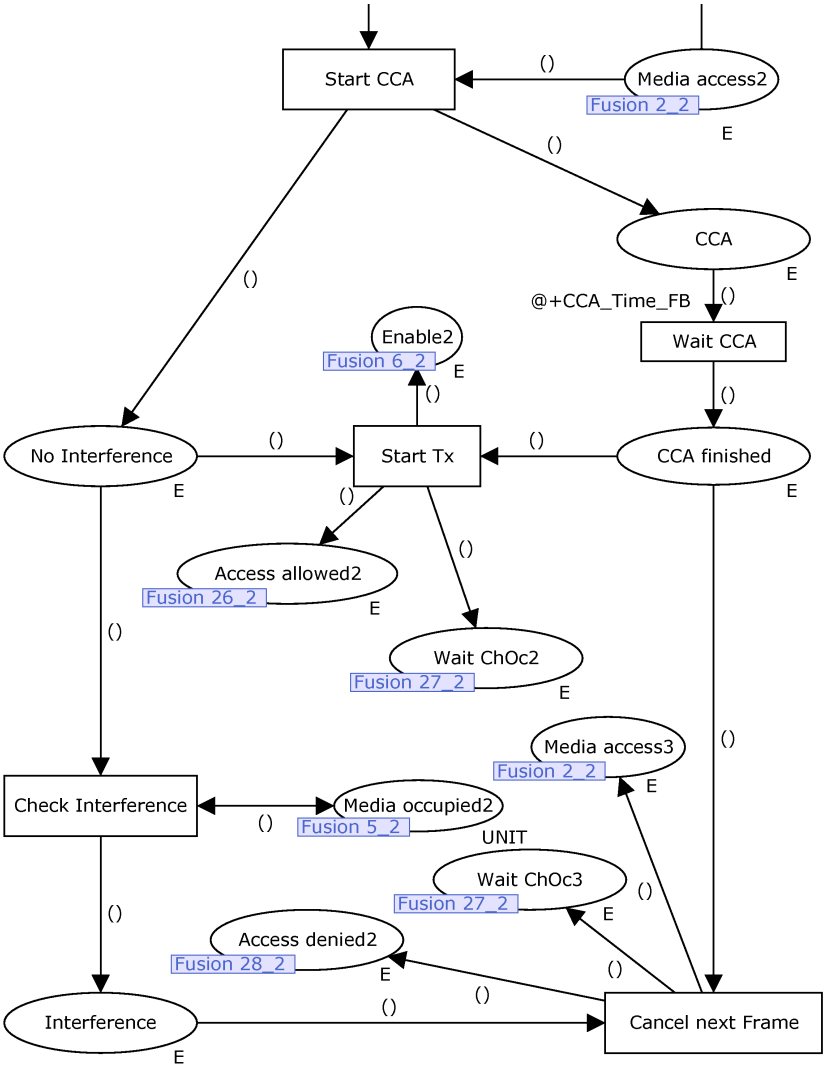

Abb. 37: *Modell zum Medienzugriff eines NFH-FB Systems (Teil 2)*

Transition „Start Tx“. Mit „Enable2“ wird die Übetragung gestartet. Weitere Übertragungen werden erlaubt („Access allowed2“) und die Kanalbelegungszeit beginnt abzulaufen („Wait ChOc2“). Wurde innerhalb der CCA Zeit eine Medienbelegung festgestellt, schaltet die Transition „Cancel next Frame“. Es wird die Rückgabe der Marke auf „Media access3“ veranlasst. Die Kanalbelegungszeit wird zwar gestartet („Wait ChOc3“), die Übertragung von Paketen allerdings gestoppt („Access denied2“).

Die Kanalbelegungszeit abzüglich der Medienbelegungszeit ist in Abbildung 36 über den Strang „Wait ChOc“ - „Delay ChOc“ - „End ChOc“ realisiert. Bei erfolgreichem CCA Check liegt eine Marke für die Dauer der Kanalbelegungszeit abzüglich der Medienbelegungszeit auf der Stelle „Access allowed“. Neue Pakete von „Media access2“ werden sofort zur Übertragung freigegeben („Enable3“). Nach Ablauf der Kanalbelegungszeit schaltet die Transition „Finish Access allowed“ und die Marke von „Access allowed“ wird abgezogen. Bei einem fehlgeschlagenen CCA Check liegt eine Marke auf „Access denied“. Nach Ablauf der Kanalbelegungszeit wird diese durch das Schalten der Transition „Finish Access denied“ abgezogen. Nach Ablauf der Kanalbelegungszeit wird die Medienbelegungszeit („Wait MB“ und „Delay MB“) und Idle-Period („Wait tIdle“ und „Delay tIdle“) abgewartet. Während dieser beiden Zeiten werden keine neuen Pakete zur Übertragung freigegeben. Nach Ablauf von Kanalbelegungszeit, Medienbelegungszeit und Idle-Period ist wieder ein CCA-Check von Nöten („CCA required“).

6.2.4 Non-FH systems load based devices using LBT based DAA

Anforderungen

„Load based“ Geräte haben einen auf LBT basierten „Spectrum Sharing“ Mechanismus implementiert. Dieser nutzt einen CCA Modus, welcher auf Energie- bzw. Trägererkennung basiert. Auch LB Geräte müssen einem minimalen Satz an Anforderungen entsprechen (siehe Abbildung 38).

Vor einer Übertragung oder vor einem Burst von Übertragungen soll das Gerät einen CCA Check durchführen. Das Gerät soll den entsprechenden Kanal für die Dauer der „CCA Observation Time“ (t_{CCA}), welche nicht kleiner als 20 μs sein sollte, überwachen. Der Kanal wird als belegt betrachtet, wenn das Energieniveau auf dem Kanal eine Schwelle überschreitet. Sollte das Gerät den Kanal als frei erkennen, so kann die Übertragung sofort beginnen. Andernfalls findet zunächst keine Übertragung statt. Das Gerät soll einen erweiterten CCA (ECCA) Check durchführen, in welchem der Kanal für die Dauer t_{ECCA} von einem Zufallsfaktor R multipliziert mit der „CCA Observaton Time“ beobachtet wird. R definiert die Anzahl an freien Idle-Slots. Diese Dauer t_{ECCA} muss der Kanal mindestens beobachtet werden, bevor eine Übertragung initiiert wird. Der Wert von R soll zufällig aus dem Bereich 1 bis q ausgewählt werden, jedes Mal, wenn ein ECCA Check benötigt wird. Der Wert wird in einem Counter abgelegt. Der Wert von q kann vom Hersteller aus dem

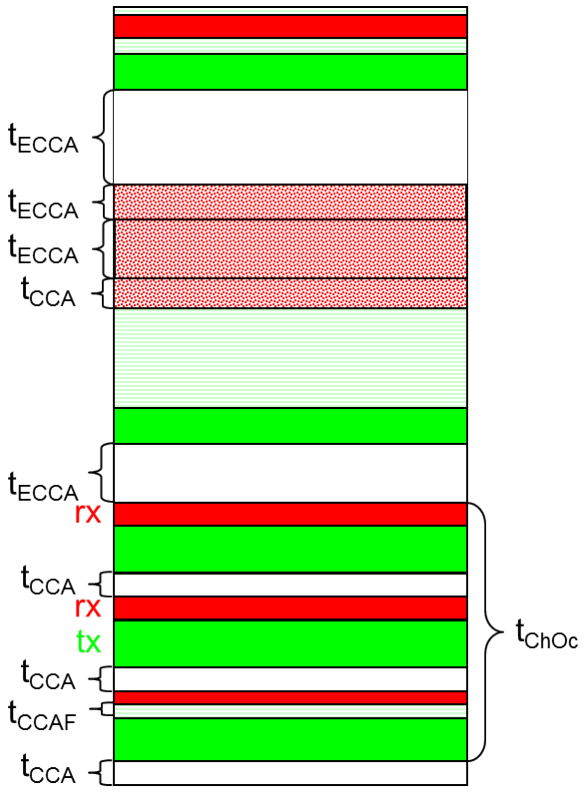

Abb. 38: *Anforderungen an NFH-LB Systeme*

Bereich 4 bis 32 für jeden ECCA Check ausgewählt werden. Der Zähler wird jedes Mal dekrementiert, wenn der Kanal für die Dauer der „CCA Observation Time“ als unbelegt erkannt wurde. Diese Zeit des unbelegten Kanals wird als Idle-Slot bezeichnet. Wenn der Zähler Null erreicht, kann das Gerät übertragen. Ebenfalls hat das Gerät die Möglichkeit, in den nicht-adaptiven Modus zu wechseln und dort seine Übertragung fortzusetzen. Allerdings müssen dabei die Anforderungen für nicht-adaptive Systeme eingehalten werden.

Die Kanalbelegungszeit t_{ChOc} soll kleiner als 13/32*q ms sein. Nach dieser Zeit muss das Gerät erneut einen ECCA Check durchführen. Wird eine Übertragung innerhalb der Kanalbelegungszeit initiiert, so muss lediglich ein CCA Check durchgeführt werden. Ist dieser negativ, ist ein ECCA Check zwingend erforderlich.

Ein Gerät kann nach dem korrekten Empfang eines Paketes, welches auch für dieses Gerät bestimmt war, den CCA Check auslassen und sofort mit der Übertragung von Management und Control Paketen fortfahren (z. B. sind ACK und Block ACK Pakete erlaubt, aber Datenpakete nicht). Ein fortlaufendes Senden solcher Übertragungen durch das Gerät ohne einen erneuten CCA Check soll die maximale Kanalbelegungszeit nicht überschreiten.

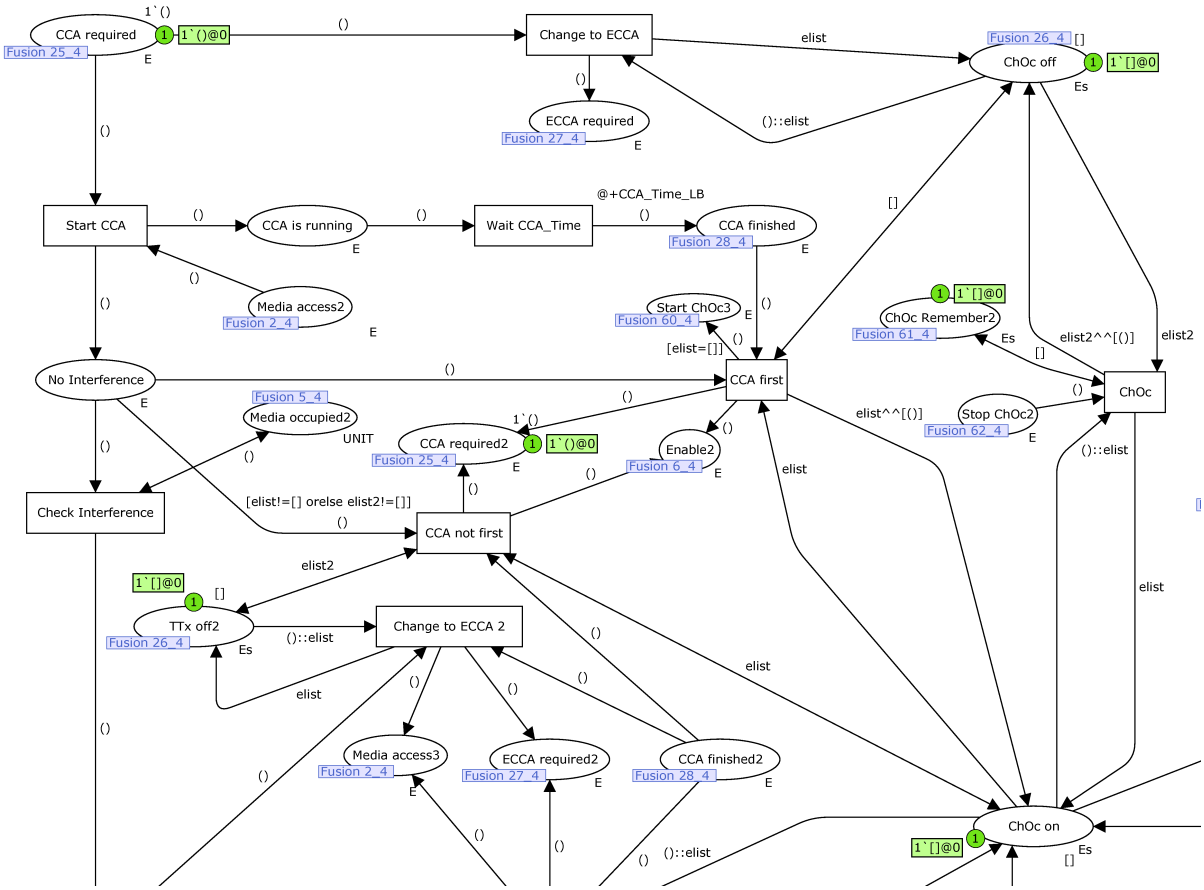

Abb. 39: *Modell zum Medienzugriff eines NFH-LB Systems (Teil 1)*

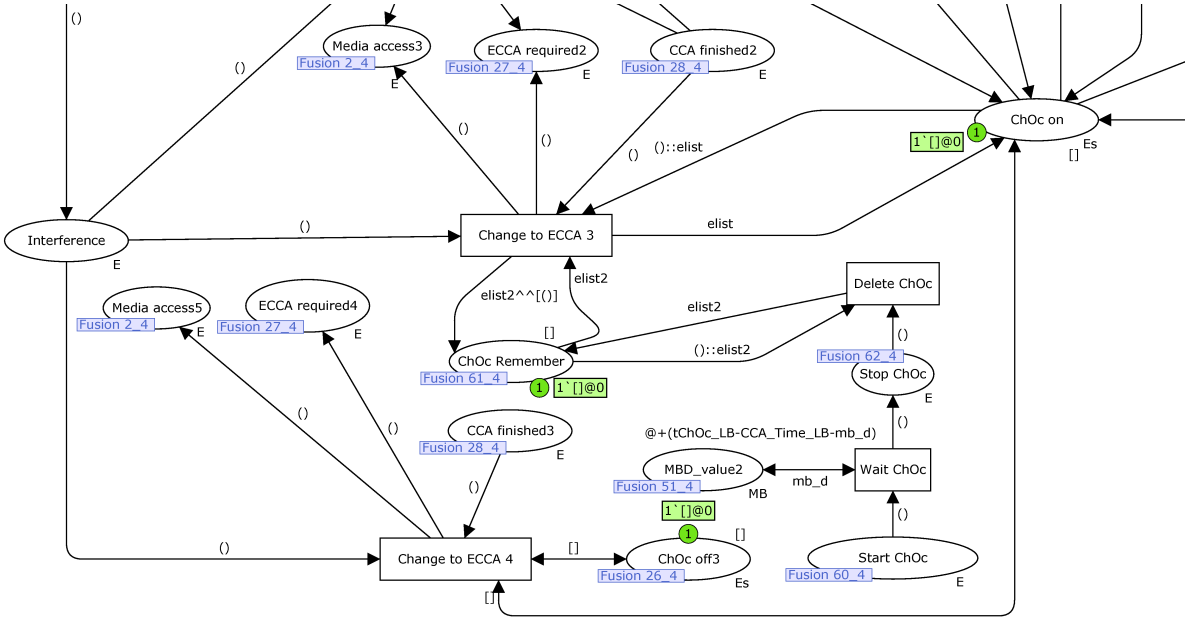

Abb. 40: *Modell zum Medienzugriff eines NFH-LB Systems (Teil 2)*

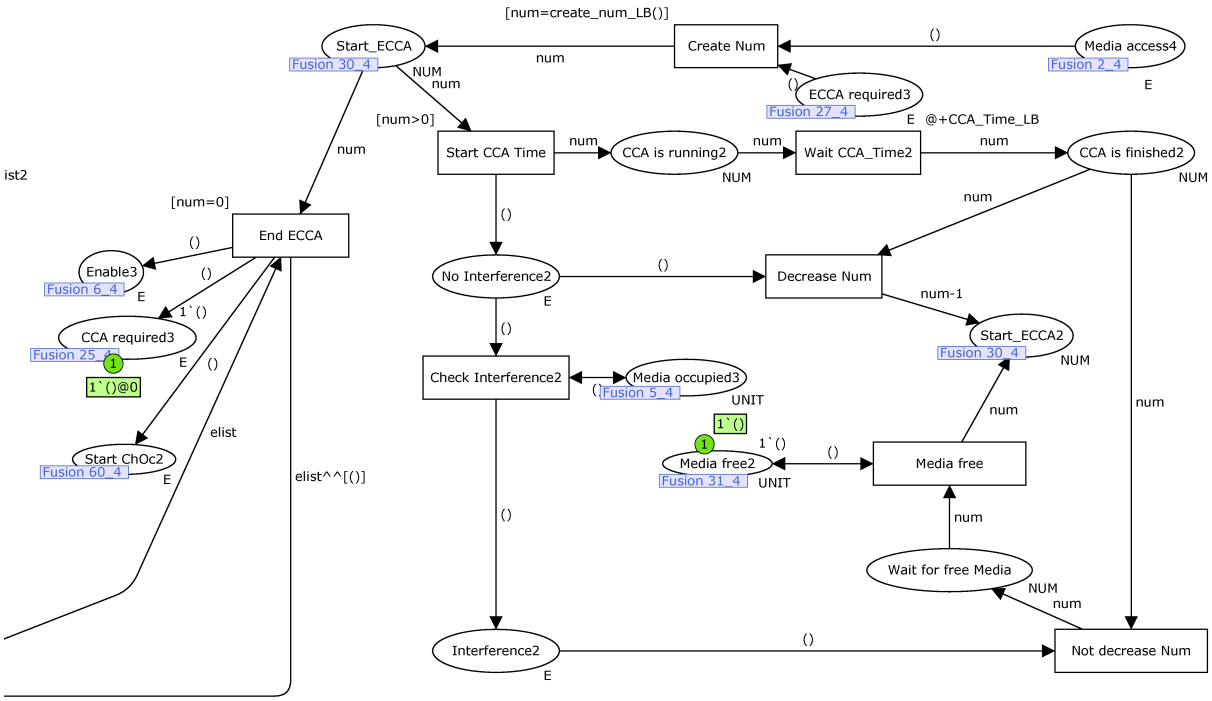

Abb. 41: *Modell zum Medienzugriff eines NFH-LB Systems (Teil 3)*

Modellbeschreibung

Abbildung 39, Abbildung 40 und Abbildung 41 zeigen die Umsetzung des Modells zum Medienzugriff eines NFH-LB Systems. Abbildung 39 und Abbildung 40 stellen den CCA Check und die Kanalbelegungszeit dar. Abbildung 41 veranschaulicht den erweiterten CCA Check.

Wenn ein neues Paket für eine Übertragung Medienzugriff beansprucht, wird eine Marke auf den Platz „Media access2“ gelegt (siehe Abbildung 39). Mit der Marke auf „CCA required“ kann die Transition „Start CCA“ schalten und der CCA Check wird gestartet. Über den Strang „CCA is running“ - „Wait CCA_Time“ - „CCA finished“ wird die CCA Zeit realisiert. Mit den beiden Plätzen „No Interference“ und „Interference“ sowie der Transition „Check Interference“ wird die Medienbelegung während der CCA Zeit überprüft. Wurde keine Belegung festgestellt, schaltet entweder „CCA first“, wenn der CCA Check das erste Mal durchgeführt wurde, oder „CCA not first“, wenn der CCA Check schon einmal zum Einsatz kam. Bei beiden Transitionen wird die Übertragung freigegeben („Enable2“), die Kanalbelegungszeit gestartet („ChOc on“) und eine Marke zurück auf „CCA required“ gelegt.

Bei fehlgeschlagenem CCA Check (Marke liegt auf „Interference“) wird die Übertragung nicht freigegeben, sondern stattdessen ein erweiterter CCA Check („ECCA required“) veranlasst (siehe auch Abbildung 40). Dabei wird unterschieden, ob die Kanalbelegungszeit noch nicht gestartet wurde („Change to ECCA 4“), ob die Kanalbelegungszeit läuft

(„Change to ECCA 3“) und ob die Kanalbelegungszeit abgelaufen ist („Change to ECCA2“). Der Zustand der Kanalbelegungszeit wird mit dem Strang „ChOc on“ - „ChOc“ - „ChOc off“ realisiert. Bei Ablauf der Kanalbelegungszeit wird automatisch in den ECCA Modus gewechselt („Change to ECCA“). Der eigentliche Zeitwert der Kanalbelegungszeit wird mit Hilfe des Strangs „Start ChOc“ - „Wait ChOc“ - „Stop ChOc“ umgesetzt. Beim Schalten der Transition „Change to ECCA 3“ ist es nun aber so, dass die Kanalbelegungszeit innerhalb dieses Stranges bereits läuft, allerdings die Marke von dem Platz „ChOc on“ entfernt wird. Wenn nun die Marke auf den Platz „Stop ChOc“ gelangt, kann die Transition „ChOc“ nicht mehr schalten. Um dieses Problem zu umgehen, existiert ein zusätzlicher Speicher „ChOc Remember“, welcher die Marke auf „Stop ChOc“ entfernt.

Beim erweiterten CCA Check in Abbildung 41 wird durch die Transition „Create Num“ eine Zufallszahl generiert, deren Wert in der Marke gespeichert wird. Diese erzeugte Marke auf „Start_ECCA“ startet einen einfachen CCA. Der Strang „CCA is running2“ - „Wait CCA_Time2“ - „CCA is finished2“ realisiert die CCA Zeit. Der Strang „No Interference2“ - „Check Interference2“ - „Interference2“ überprüft die Belegung des Mediums. Wurde keine Belegung festgestellt, wird der Wert der Zufallszahl, die sich in der Marke befindet, um eins reduziert und das einfache CCA Check beginnt von vorn („Start_ECCA“). Wurde eine Belegung festgestellt, wird gewartet, bis diese Belegung des Mediums beendet ist. Erst danach wird erneut ein CCA Check gestartet. Die Zufallszahl wird dabei nicht reduziert. Wenn die Marke den Wert Null erreicht, schaltet die Transition „End ECCA“. Die Übertragung wird freigegeben und die Kanalbelegungszeit gestartet.

6.3 Vereinfachungen

Wie bei der Modellierung üblich, werden gegenüber der Realität Vereinfachungen angenommen. Die gravierendsten Modellgrenzen betreffen die Betrachtung von Leistung und Interferenz. Es wird bei dem gewählten Ansatz angenommen, dass jeder Teilnehmer mit allen anderen Teilnehmern des Modells interagiert, wenn die Geräte jeweils denselben Frequenzbereich nutzen. Das heißt, es wird eine Betrachtung der Frequenzen, aber nicht der Leistungen vorgenommen. Weder die Sendeleistung noch die Signalleistung am Empfänger findet Berücksichtigung. Damit sind im Modell keine Pfadverluste implementiert. Auch zeit- und frequenzselektive Varianzen entfallen. Bei der Interferenzmodellierung wird angenommen, dass jede Überlappung von Zeit und Frequenz zu einer Zerstörung der involvierten Pakete führt. Das Petri-Netz-Modell bildet nur Datenübertragungen nach, allerdings keine Managementpakete wie Acknowledgements oder Paketwiederholungen.

Angelehnt an die industrielle Automation erfolgt die Initiierung der Übertragung periodisch. Bei der Modellierung werden die Pakete als Ganzes betrachtet. Eine Nachbildung der Bit-Ebene erfolgt nicht. Die Pakete weisen während der gesamten Simulationsdauer eine konstante Länge auf.

Das Modell ist auf Untersuchungen des Medienzugriffsverfahrens und der Kollision während der Übertragung beschränkt. Aus diesem Grund ist auch nur die unbestätigte Übertragung von einem zu einem anderen Gerät Bestandteil der Modellierung.

6.4 Verifikation und Validierung

6.4.1 Formale Verifikation

Ausgangspunkt für eine formale Verifikation ist ein Zustandsraum (State Space), welcher mittels Transformation eines beliebigen Petri-Netzes generiert werden kann. Ein Zustandsraum ist ein gerichteter Graph mit Knoten (Nodes), die für einen erreichbaren Zustand des zugrundeliegenden Petri-Netzes stehen, und Kanten (Arcs), welche von den auftretenden Binding Elements gebildet werden. Ein Zustand wird auch als Markierung (Marking) bezeichnet. Eine Markierung ist nicht zu verwechseln mit einer Marke (Token). Eine fest verbundene Komponente (Strongly Connected Component, SCC) ist ein maximaler Subgraph, in dem alle enthaltenen Knoten untereinander erreichbar sind. Mit Hilfe von SCC können Zustände bzw. Knoten des State Spaces zusammengefasst werden. Der Zustandsraum wird verkleinert.

Mit Hilfe des State Spaces können verschiedene Eigenschaften des Petri-Netzes überprüft werden:

1. Grenzen (Boundedness Properties)
    - Best Upper Integer Bounds: Gibt Auskunft über die maximale Anzahl an Marken, die auf einem Platz in jedem erreichbaren Zustand liegen können.
    - Best Lower Integer Bounds: Gibt Auskunft über die minimale Anzahl an Marken, die auf einem Platz in jedem erreichbaren Zustand liegen können.
    - Best Upper/Lower Multi-Set Bounds: Gibt nicht nur Auskunft über die Anzahl, sondern auch über die Färbung (Colour) von Marken auf einem Platz. Diese Eigenschaft gibt für jede Färbung in einem Colour-Set eines Platzes die maximale/minimale Anzahl von Marken mit dieser Färbung an.
2. Home Properties
    - Home Markings: Sind Zustände, die von jedem Zustand aus erreicht werden können. Es ist somit unmöglich einen Pfad zu haben, welcher nicht zu diesem Zustand führt. Diese Eigenschaft beschreibt nur die Möglichkeit, diesen Zustand zu erreichen. Es gibt aber keine Garantie durch beispielsweise Schleifen.
3. Liveness Properties
    - Dead Marking: Ist ein Zustand, in dem kein Binding Element aktiviert ist und damit keine Transition schalten kann. Es existiert kein Folgezustand.

- Live Transitions: Eine Transition ist „live“, wenn in jedem erreichbaren Zustand ein Pfad existiert, der diese Transition enthält. Es kann somit nicht verhindert werden, dass diese Transition auftritt. Wenn es Dead Markings gibt, kann es keine Live Transitions geben, da in einem Dead Marking keine Transition aktiviert werden kann.
- Dead Transitions: Eine Transition wird als „Dead Transition“ bezeichnet, wenn es keinen erreichbaren Zustand gibt, in dem diese aktiviert wird. Sie tritt nie auf. Wenn keine „Dead Transitions“ vorhanden sind, besteht die Möglichkeit, dass jede Transition in einem Modell mindestens einmal aktiviert werden kann. „Dead Transitions“ können aus dem Modell entfernt werden, ohne dass sich dessen Verhalten ändert.

Die aufgelisteten Eigenschaften können in CPN Tools mit einem State Space Report oder mit spezifischen Standard ML Funktionen überprüft werden. Der State Space Report gibt zusätzlich zu den genannten Eigenschaften grundlegende Informationen zur Größe des State Space und des SCC Graphen an. Dabei wird auch kontrolliert, ob der State Space eine endliche Größe aufweist. Wenn vergleichsweise weniger Knoten im SCC Graphen als im State Space existieren, deutet dies auf Schleifen hin.

Für alle beschriebenen Modelle wurde der State Space Report erzeugt. Der Zustandsraum hatte in allen Fällen eine endliche Größe. Die Petri-Netze bewegten sich in festen Grenzen. Es existierten weder „Live“ noch „Dead Transitions“. SCC Graph und Zustandsraum hatten immer die gleiche Größe, weil keine Schleifen im Modell möglich sind. Für jedes Modell wurde ein „Dead Marking“ ermittelt. Da bei jedem Modell nur eine endliche Anzahl an Paketen übertragen wird, ist eine „Dead Marking“ und damit ein Deadlock des Petri-Netzes eine logische Konsequenz. Allerdings ist jede „Dead Marking“ auch eine „Home Marking“. Das bedeutet, dass die Modelle einen konkreten Ausgang haben, die von jedem Zustand aus erreichbar sind. Damit sind die Modelle partiell richtig.

6.4.2 Verifikation und Validierung des Medienzugriffsverhaltens

Für die Verifikation und Validierung des Medienzugriffsverhaltens wird, wie bei der formalen Verifikation, vom State Space ausgegangen. Die Verifikation dient der Überprüfung, ob die Modelle genau das Verhalten nachbilden, welches im EN 300 328-Standard festgelegt wurde. Dagegen erbringt die Validierung den Nachweis, dass die funktionalen Anforderungen von den Modellen erfüllt werden und damit die aus den Modellen gewonnenen Simulationsergebnisse reproduzierbar sind. Tabelle 7 listet die einzelnen Bestandteile auf, die im Rahmen der Verifikation und Validierung überprüft wurden. Das Hauptaugenmerk lag dabei auf dem Zeitverhalten und den Abhängigkeiten der verschiedenen Zeiten. Außerdem wurden die Frequenzen der Pakete und die Einhaltung vorgegebener Abfolgen untersucht. Für die einzelnen Punkte wird auf Quelltexte im Anhang referenziert. Auf Basis dieser

Quelltexte wurden die Messergebnisse zur Verifikation bzw. Validierung gewonnen. Da die Quelltexte in „CPN Tools“ ausgeführt werden mussten, kam die Programmiersprache „Standard ML“ zum Einsatz.

Tab. 7: *Die zu verifizierenden bzw. validierenden Bestandteile der einzelnen Zugriffsverfahren*

Bestandteil	**NFH-NA**	**NFH-NLBT**	**NFH-FB**	**NFH-LB**
Zeiten	- Zeit, in der das Medium uneingeschränkt genutzt werden darf (Quelltext 12[1]): $t_{seq} = 3,5 - 10ms$ - Zeit, in der nicht auf das Medium zugegriffen werden darf (Quelltext 13[1]): $t_{Gap} =$ $3,5 - 10ms$ (t_{seq} und t_{Gap} müssen gleich sein) - Beobachtungszeitraum, auf den sich die Medium Utilization bezieht (Quelltext 16[1]): $t_{Obs} = 1s$ - Zeit, in der das Medium belegt wird (Quelltext 16[1])	- Zeit, in der das Medium uneingeschränkt genutzt werden darf (Quelltext 12[1]): $t_{ChOc} < 40ms$ - Zeit, in der nicht auf das Medium zugegriffen werden darf (Quelltext 13[1]): $t_{Idle} \geq$ $max(100ms, 5\% \cdot$ $t_{ChOc})$ - Zeit, in der nach Erkennung einer Interferenz der Kanal gesperrt ist (Quelltext 16[1]): $t_{ua} \geq 1s$ - Zeit, in der das Medium belegt wird (Quelltext 16[1])	- Zeit, in der das Medium auf Belegung überprüft wird (Quelltext 23[1]): $t_{CCA} \geq 20\mu s$ - Zeit, in der ohne ein weiteres CCA auf das Medium zugegriffen werden darf (Quelltext 12[1]): $t_{ChOc} = 1 - 10ms$ - Zeit, in der nach Ablauf der t_{ChOc} nicht auf das Medi- um zugegriffen werden darf (Quelltext 13[1]): $t_{Idle} = 5\% \cdot t_{ChOc}$ - Fixed Frame Period (Quelltext 16[1]) $t_{FFP} =$ $t_{Idle} + t_{ChOc}$	- Zeit, in der das Medium auf Belegung überprüft wird (wenn vorher keine Belegung festgestellt wurde, Quelltext 23[1]): $t_{CCA} \geq 20\mu s$ - Zufällige Zeit, die das Medium überprüft wird, wenn eine Belegung festgestellt wurde (Quelltext 26[1]): $t_{ECCA} =$ $t_{CCA} - q \cdot t_{CCA}$ (q=4-32) - Zeit, in der nur ein einfaches CCA ausgeführt werden muss.
				Fortsetzung auf der nächsten Seite

[1]Quelltext, auf dessen Basis die Messergebnisse zur Verifikation bzw. Validierung gewonnen wurden

Tab. 7: *Die zu verifizierenden bzw. validierenden Bestandteile der einzelnen Zugriffsverfahren – Fortsetzung*

Bestandteil	**NFH-NA**	**NFH-NLBT**	**NFH-FB**	**LB**
Zeiten			- Zeit, in der das Medium belegt wird (Quelltext 16[2])	Nach Ablauf dieser Zeit muss ein Extended CCA ausgeführt werden (Quelltext 16[2]): $t_{ChOc} <$ $[13/32 \cdot q]ms$ - Zeit, in der das Medium belegt wird (Quelltext 16[2])
Frequenz	Alle Pakete müssen auf derselben Frequenz übertragen werden (Quelltext 14[2]).	Alle Pakete müssen auf derselben Frequenz übertragen werden (Quelltext 14[2]).	Alle Pakete müssen auf derselben Frequenz übertragen werden (Quelltext 14[2]).	Alle Pakete müssen auf derselben Frequenz übertragen werden (Quelltext 14[2]).
Sperrzeiten (In dieser Zeit darf kein eigenes Paket auf dem Medium liegen.)	Während t_{Gap} darf das Medium nicht belegt werden (Quelltext 15[2]).	Während t_{ua} und t_{Idle} darf das Medium nicht belegt werden (Quelltext 15[2]).	Während t_{CCA} und t_{Idle} darf das Medium nicht belegt werden (Quelltext 15[2]). Bei einem negativen CCA Check darf das Medium in der darauffolgenden FFP ebenfalls nicht belegt werden (Quelltext 24[2]).	Während t_{CCA} und t_{ECCA} darf das Medium nicht belegt werden (Quelltext 15[2]). Auch bei der Erkennung einer bereits vorhandenen Belegung darf nicht auf das Medium zugegriffen werden (Quelltext 27[2]).

Fortsetzung auf der nächsten Seite

[2]Quelltext, auf dessen Basis die Messergebnisse zur Verifikation bzw. Validierung gewonnen wurden

Tab. 7: *Die zu verifizierenden bzw. validierenden Bestandteile der einzelnen Zugriffsverfahren – Fortsetzung*

Bestandteil	**NFH-NA**	**NFH-NLBT**	**NFH-FB**	**LB**
Sonstige Einschränkungen	Eine Medium Utilization von 10 % darf nicht überschritten werden (Quelltext 17[3]).			
Pakete	Wird das gleiche Paket vom Medium genommen, welches auch darauf gelegt wird (Quelltext 18[3])?	Wird das gleiche Paket vom Medium genommen, welches auch darauf gelegt wird (Quelltext 18[3])?	Wird das gleiche Paket vom Medium genommen, welches auch darauf gelegt wird (Quelltext 18[3])?	Wird das gleiche Paket vom Medium genommen, welches auch darauf gelegt wird (Quelltext 18[3])?
Sequenzzeit	Tritt bei aktiver t_{seq} keine Verzögerung beim Medienzugriff auf (Quelltext 19[3])?	Tritt bei aktiver t_{ChOc} keine Verzögerung beim Medienzugriff auf (Quelltext 19[3])?	Tritt bei aktiver t_{ChOc} keine Verzögerung beim Medienzugriff auf (Quelltext 19[3])?	Wird innerhalb t_{ChOc} nur CCA ausgeführt (Quelltext 28[3])?
Abfolge	Wird nach Ablauf der t_{seq} wirklich t_{Gap} ausgeführt (Quelltext 20[3])? Wird nach Ablauf der t_{Obs} die Medium Utilization wirklich zurückgesetzt (Quelltext 21[3])?	Wird nach Ablauf der t_{ChOc} wirklich t_{Idle} ausgeführt (Quelltext 20[3])? Wird bei Interferenz wirklich t_{ua} abgewartet (Quelltext 22[3])?	Wird nach Ablauf der t_{ChOc} wirklich t_{Idle} ausgeführt (Quelltext 20[3])? Ist nach negativem CCA der nachfolgende FFP wirklich gesperrt (Quelltext 25[3])?	Wird nach Ablauf der t_{ChOc} wirklich ECCA ausgeführt (Quelltext 20[3])?

[3]Quelltext, auf dessen Basis die Messergebnisse zur Verifikation bzw. Validierung gewonnen wurden

6.5 Simulation

6.5.1 Vorbetrachtungen zu den ausgewählten Simulationsszenarien

Wie bereits erwähnt, ist die ETSI-Arbeitsgruppe ERM TG11 für den Standard EN 300 328 verantwortlich. Ihre Aufgabe ist es:

> ...to study the changes necessary to EN 300 328 in response to the market driven requirements.[4]

Die ERM TG11 soll also die EN 300 328 überarbeiten, wenn die Anforderungen des Marktes eine solche Änderung notwendig machen. Allerdings sind Ergebnisse von Studien, welche eine Anpassung erforderlich machen, nicht öffentlich verfügbar. Es existieren auch keine Veröffentlichungen über marktgetriebende Anforderungen, die eine Überarbeitung der EN 300 328 notwendig machen würden. Aus diesem Grund ist anzunehmen, dass nicht der Markt die Änderung der EN 300 328 erfordert, sondern stattdessen die überarbeitete Version dieses Standards Einfluss auf den Markt nehmen wird.

Für eine Überarbeitung der EN 300 328 wird im Rahmen einer R&TTE Direktive die folgende Anforderung gestellt:

> ...radio equipment shall be constructed that it effectively uses the spectrum allocated to terrestrial/space radio communications and orbital resources...[5]

Jedes Gerät, das konform zu den technischen Anforderungen der EN 300 328 ist, soll das Spektrum effizient nutzen.

Eine andere Anforderung an die Überarbeitung der EN 300 328 stammt aus einer TCAM 26 Bestimmung:

> A mitigation technique is considered as appropriate if: ...it ensures an equal spectrum sharing with other devices, i. e. congestion, if any, occures in a gradual way impacting equally the service of all types of devices.[6]

Das heißt, dass alle Geräte, welche die technischen Anforderungen der EN 300 328 erfüllen, das Spektrum gleichmäßig aufteilen sollen.

Die Überprüfung, ob die genannten Anforderungen erfüllt werden, soll mit Hilfe ausgewählter Simulationsszenarien erfolgen:

- Das Spektrum wird effizient genutzt.
- Alle Geräte teilen sich das Spektrum gleichmäßig auf.

[4] Task of ETSI ERM TG11 (see http://portal.etsi.org/erm/ERMtg11_ToR.asp)
[5] Directive 1999/5/EC (R&TTE Directive) article 3.2
[6] TCAM 26 decision on Sub-class 22 (see http://ec.europa.eu/enterprise/sectors/rtte/files/note_subclass22_en.pdf)

- Instabilität
- Unvorhersagbares Zeitverhalten

Mit einigen der aufgezählten Szenarien werden zudem die Anforderungen der industriellen Automation überprüft.

Zur Untersuchung, inwieweit die vier Anforderungen von den Medienzugriffsmechanismen der EN 300 328 erfüllt werden, soll die Wahrscheinlichkeit für eine gegenseitige Beeinflussung möglichst hoch gewählt werden. Dazu wurde sich zum einen bei den Szenarien bewusst auf non-FH Systeme beschränkt. Diese müssen sich einen Frequenzkanal teilen und können zudem nicht zur Vermeidung von Kollisionen auf einen anderen ausweichen. Des Weiteren wurde der Zeitabstand bei der Paketgenerierung so gewählt, dass die Übertragungsfähigkeit des Mediums überschritten wird. Bei der Verwendung von FH-Systemen ist es sehr aufwendig, gezielt eine gegenseitige Beeinflussung herzustellen. Auch ist es sehr schwer, Aussagen bezüglich der oben genannten Anforderungen zu gewinnen, wenn sich die Systeme durch beispielsweise einer Unterbelegung des Mediums einer gegenseitigen Beeinflussung entziehen können. Aus diesem Grund sind durch die Beschränkung auf non-FH Systeme und einer Überlastung des Mediums die angeführten Thesen und die unterschiedlichen Medienzugriffsmechanismen ideal zu bewerten.

6.5.2 Simulationsszenario: „Das Spektrum wird effizient genutzt“

Beschreibung

Im ersten Szenario soll die Anforderung an die EN 300 328 überprüft werden, ob das Spektrum effizient genutzt wird. Effizienz beschreibt dabei die Größe, wie gut Ressourcen für einen vorgesehenen Zweck verwendet werden. Das Spektrum soll als Ressource bestmöglich zur Realisierung der Kommunikation ausgenutzt werden. Tabelle 8 listet die relevanten Parameter auf.

Die Simulationsumgebung betrachtet vier Systeme, die alle NFH-LB als Medienzugriff verwenden. Der Wert für q ist fest auf 32 eingestellt. Jedes der vier Systeme enthält vier Verbindungen (n_{Con}). Die vier Systeme unterscheiden sich in den Parametern t_{onmax}, t_{TI} und der Paketanzahl für jede Verbindung. t_{onmax} meint dabei die Dauer, die eine Verbindung das Medium ohne Unterbrechung belegen darf. Sie ist gleichbedeutend mit der Medienbelegungszeit, also der zeitlichen Dauer eines Paketes. Die Pakete werden periodisch generiert. Der Wert von t_{TI} steht für den Sendezeitabstand. Er definiert das Intervall, in dem für eine einzelne Verbindung Pakete generiert werden. t_{TImin} ergibt sich aus der Medienbelegungsdauer t_{onmax} und dem gewählten Medienzugriffsverfahren. Dieser Parameter kennzeichnet den minimal zulässigen Sendezeitabstand. Aus der Anzahl an Verbindungen,

Tab. 8: *Parameter des Simulationsszenarios: "Das Spektrum wird effizient genutzt"*

System	t_{onmax} [ms]	n_{Con}	t_{TImin} [ms]	t_{TI} [ms]	Req [%]	Init [ms]	Packets
NFH-LB	1	4	1,33	16	25%	rand(t_{TI})	50.000
NFH-LB	3	4	3,33	48	25%	rand(t_{TI})	16.667
NFH-LB	5	4	5,33	80	25%	rand(t_{TI})	10.000
NFH-LB	7	4	7,33	112	25%	rand(t_{TI})	7.143
					100%		

t_{onmax} und t_{TI} ergibt sich die Kommunikationsanforderung eines Systems:

$$CR = \frac{n_{Con} \cdot t_{onmax}}{t_{TI}} \tag{68}$$

Jedes System dieses Simulationsszenarios weist eine Kommunikationsanforderung von 25 % auf. Aus der Summe der Kommunikationsanforderungen der vier Systeme wird eine Gesamtanforderung von 100 % an das Medium gestellt. Das Medium wird an der Grenze seiner Übertragungsfähigkeit betrieben. Die Initialisierung der Übertragung sämtlicher Verbindungen erfolgt zufällig. Das heißt, dass nach dem Simulationsstart die Verbindungen zufällig innerhalb der Dauer des Sendezeitabstands mit Senden beginnen. Die Paketanzahl ist ebenfalls unterschiedlich (siehe Tabelle 8). Diese wurde für die einzelnen Systeme angepasst, damit alle Verbindungen für die gesamte Simulationsdauer Pakete übertragen. Jedes Gerät hat einen Paketbuffer mit einer Größe von drei. Im Falle eines Speicherüberlaufs wird das älteste Paket verworfen.

Zusammenfassend ist zu bemerken, dass in diesem Simulationsszenarium alle Systeme die gleiche Kommunikationsanforderung haben und durch ihre Summe das Medium am Limit dessen Übertragungsfähigkeit betrieben wird. Allerdings unterscheiden sich die Systeme in der Paketlänge und damit der Dauer der Medienbelegung sowie dem zeitlichen Abstand, in dem die Pakete generiert werden.

Ergebnisse

Abbildung 42 zeigt das Ergebnis der Paketverteilung der übertragenen Pakete für die einzelnen Verbindungen. Grün dargestellt sind die korrekt übertragenen Pakete, die roten Bereiche stehen für die Pakete, die aufgrund von Interferenzen zerstört wurden, und mit gelb werden die durch Speicherüberlauf verworfenen Pakete angegeben. Offensichtlich werden zwischen 10 und 25 % der Pakete der einzelnen Verbindungen durch Interferenzen zerstört. Der andere Teil der Pakete wurde korrekt übertragen. Verworfene Pakete traten nicht auf. Weiterhin ist kein eindeutiger Unterschied durch die unterschiedlichen Sendezeitabstände und Medienbelegungsdauern auszumachen.

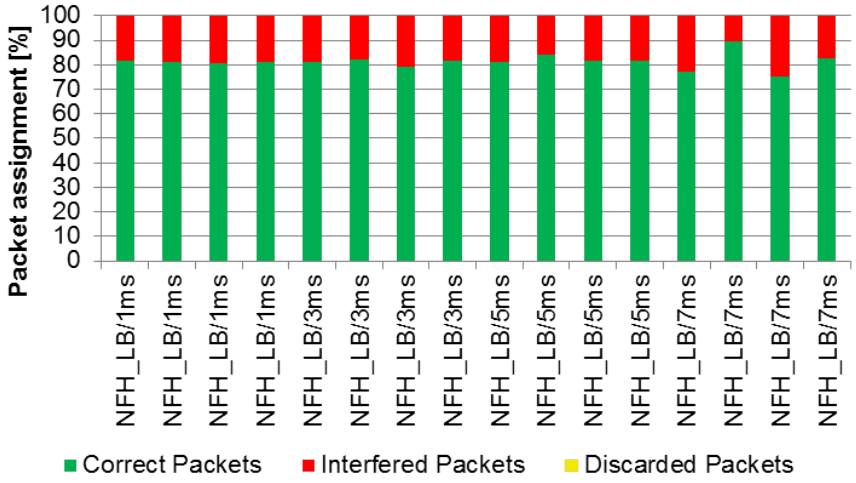

Abb. 42: *Paketverteilung des Simulationsszenarios: "Das Spektrum wird effizient genutzt"*

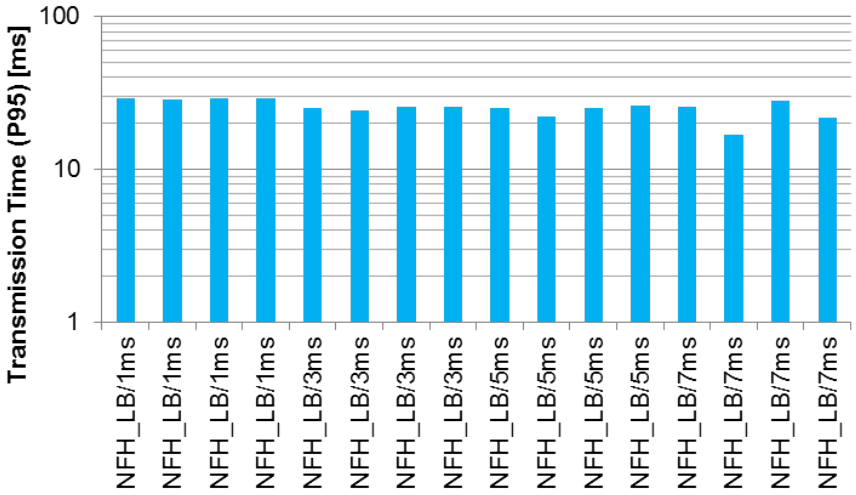

Abb. 43: *Perzentil 95 der Übertragungszeit im Simulationsszenario: "Das Spektrum wird effizient genutzt"*

In Abbildung 43 wird das Perzentil 95 der Übertragungszeit für die einzelnen Verbindungen dargestellt. Dieses befindet sich in einem Bereich zwischen 15 und 30 ms. Auch in diesem Diagramm ist kein Unterschied zwischen den unterschiedlichen Sendezeitabständen und Medienbelegungsdauern auszumachen. Pakete mit kleiner Medienbelegungsdauer, die einen wesentlichen Teil der Gesamtübertragungszeit ausmacht, haben die gleiche Übertragungszeit wie Pakete mit großer Medienbelegungszeit.

Der Umstand, dass 20 % der Pakete durch Interferenzen zerstört werden, obwohl das Modell keine Übertragungswiederholungen enthält und sich eine kleinere Medienbelegungsdauer nicht wesentlich auf die Gesamtübertragungszeit auswirkt, weist nach, dass mindestens von einem der in der EN 300 328 definierten Systeme (NFH-LB) das Spektrum ineffizient genutzt wird. Die nachfolgenden Simulationsszenarien zeigen allerdings, dass das Spektrum von allen vier untersuchten Medienzugriffsmechanismen der EN 300 328 ineffizient verwendet wird.

6.5.3 Simulationsszenario: „Alle Geräte teilen sich das Spektrum gleichmäßig auf“

Beschreibung

In diesem Szenario soll die Anforderung an die EN 300 328 überprüft werden, ob alle Geräte gleichmäßig das Spektrum untereinander aufteilen. Das bedeutet, dass Geräte mit unterschiedlichem Zugriffsmechanismus sich gleichmäßig vom Medium zurückziehen sollen, wenn dieses überlastet ist. Tabelle 9 listet die relevanten Parameter auf.

Die Simulationsumgebung besteht aus den vier Systemen NFH-NA, NFH-NLBT, NFH-FB und NFH-LB. Alle Systeme bestehen aus jeweils vier Verbindungen. Die Medienbelegungszeit der Pakete und der Sendezeitabstand sind bei allen Systemen gleich groß. Damit ergibt sich für alle Systeme die identische Kommunikationsanforderung von 25 %. Das Medium wird mit einer Gesamtanforderung von 100 % erneut am Limit betrieben. Die jeweilige periodische Übertragung wird zufällig innerhalb der Dauer des Sendezeitabstandes initialisiert. Jede Verbindung soll 10.000 Pakete übertragen.

Ergebnisse

Das Diagramm in Abbildung 44 zeigt die Paketverteilung für das Simulationsszenario “Alle Geräte teilen sich das Spektrum gleichmäßig auf“. Die Darstellung offenbart, dass sämtliche Pakete des NFH-NA Systems durch Interferenzen zerstört werden. Die Ursache dafür ist, dass das Medium und die Übertragung nicht überprüft werden. Es wird einfach gesendet unabhängig vom möglichen Kollisionsrisiko. Auch beim NFH-NLBT System gehen annährend 100 % der Pakete verloren. Aber hier werden die Pakete verworfen, bevor sie übertragen werden. Die Ursache liegt in der Sperrung des Kanals für eine Sekunde,

Tab. 9: *Parameter des Simulationsszenarios: "Alle Geräte teilen sich das Spektrum gleichmäßig auf"*

System	t_{onmax} [ms]	n_{Con}	t_{TImin} [ms]	t_{TI} [ms]	Req [%]	Init [ms]	Packets
NFH-NA	5	4	50	80	25%	rand(t_{TI})	10.000
NFH-NLBT	5	4	5,05	80	25%	rand(t_{TI})	10.000
NFH-FB	5	4	6,25	80	25%	rand(t_{TI})	10.000
NFH-LB	5	4	5,33	80	25%	rand(t_{TI})	10.000
					100%		

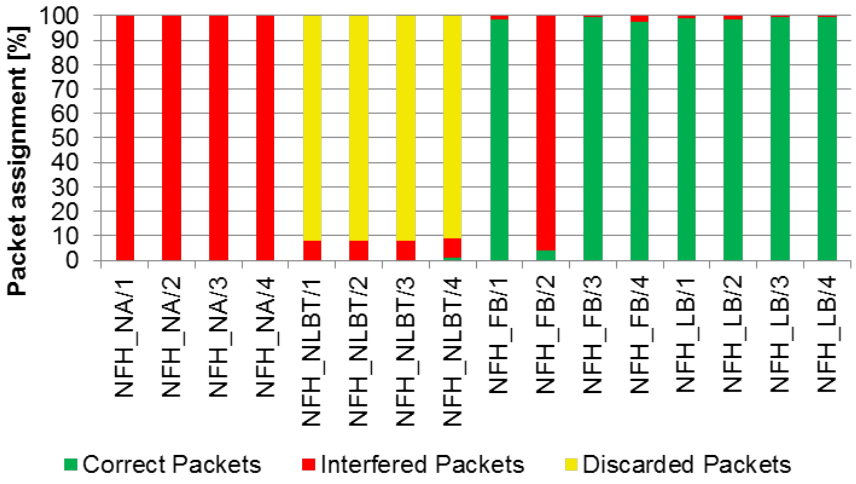

Abb. 44: *Paketverteilung des Simulationsszenarios: "Alle Geräte teilen sich das Spektrum gleichmäßig auf"*

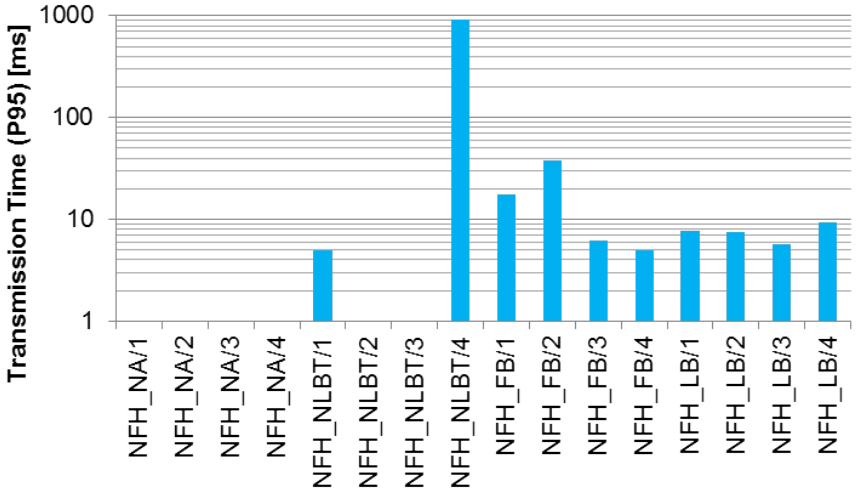

Abb. 45: *Perzentil 95 der Übertragungszeit im Simulationsszenario: "Alle Geräte teilen sich das Spektrum gleichmäßig auf"*

wenn eine Interferenz bzw. Kollision aufgetreten ist. Die LBT basierten Systeme übertragen annähernd 100 % ihrer Pakete. Nur eine NFH-FB Verbindung weist einen Großteil an durch Interferenzen zerstörten Paketen auf. Die Ursache hierfür liegt in der zeitlichen Überlagerung dieser Verbindung mit einer NFH-NA Verbindung. Die NFH-NA Verbindung kann die Belegung durch die NFH-FB Verbindung nicht erkennen. Dadurch startet sie die Übertragung von Paketen, obwohl das Medium nicht frei ist. Paketkollisionen sind die Folge.

Abbildung 45 stellt das Perzentil 95 der Übertragungszeit für die einzelnen Verbindungen dar. In den Fällen, in denen eine Verbindung kein einziges Paket übertragen hat, ist keine blaue Säule dargestellt. Bei der vierten Verbindung des NFH-NLBT Systems tritt ein Wert von fast einer Sekunde auf. Dieser ist auf die eine Sekunde Wartezeit zurückzuführen. Die LBT basierten Systeme bewegen sich im Bereich von etwa 10 ms. Nur die zweite Verbindung des NFH-FB Systems, welche bereits eine große Zahl an Paketverlusten aufwies, zeigt einen Ausreißer in der Übertragungszeit.

Die Ergebnisse zeigen, dass wenn das Medium an der Grenze der Übertragungsfähigkeit betrieben wird, die LBT basierten Systeme gegenüber den NFH-NA und NFH-NLBT basierten Systemen bevorzugt werden. Die unterschiedlichen Medienzugriffsmechanismen ziehen sich nicht gleichmäßig vom Medium zurück. Aus diesem Grund wurde die These “Alle Geräte teilen sich das Spektrum gleichmäßig auf“ widerlegt. Allerdings muss auch festgehalten werden, dass die These gültig ist, wenn das Medium unterhalb seiner Leistungsgrenze betrieben wird. Ein separates Simulationsszenario zum Beweisen dieser Aussage wird in dieser Arbeit aus Platzgründen nicht präsentiert.

6.5.4 Simulationsszenario: „Instabilität“

Beschreibung

In diesem Szenario wurde das Medium bewusst und sehr stark überlastet, um zu überprüfen, wie sich die einzelnen Medienzugriffsverfahren in dieser Extremsituation verhalten. Tabelle 10 listet die relevanten Parameter auf.

Die Simulationsumgebung besteht erneut aus den vier unterschiedlichen Systemen: NFH-NA, NFH-NLBT, NFH-FB und NFH-LB. Alle Systeme umfassen vier Verbindungen. Die Verbindungen unterscheiden sich in ihrer Paketlänge t_{onmax} sowie ihrem Sendezeitabstand t_{TI} und damit auch in ihrer Kommunikationsanforderung. Die Gesamtkommunikationsanforderung liegt bei 315 % und beträgt damit mehr als das Dreifache der möglichen Übertragungsfähigkeit des Mediums. Die Paketanzahl wurden zur Sicherstellung einer gleichen Simulationsdauer aller Pakete entsprechend angepasst. Die Initialisierung der periodischen Übertragung ist zufällig.

Tab. 10: *Parameter des Simulationsszenarios: "Instabilität"*

System	t_{onmax} [ms]	n_{Con}	t_{TImin} [ms]	t_{TI} [ms]	Req [%]	Init [ms]	Packets
NFH-NA	5	4	50	50	40%	rand(t_{TI})	16.020
NFH-NLBT	40	4	40,05	160,2	100%	rand(t_{TI})	5.000
NFH-FB	10	4	12,5	50	80%	rand(t_{TI})	16.020
NFH-LB	7	4	7,33	29,32	95%	rand(t_{TI})	27.320
					315%		

Ergebnisse

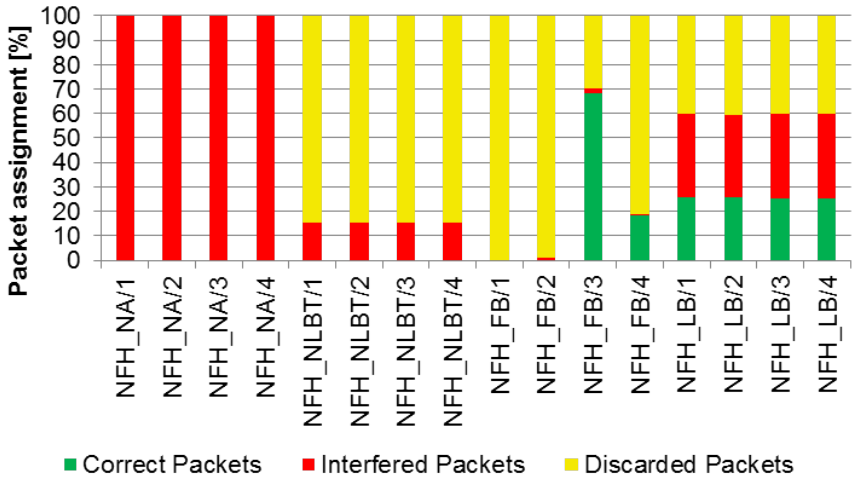

Abb. 46: *Paketverteilung des Simulationsszenarios: "Instabilität"*

Abbildung 46 zeigt für die Paketverteilung ein ähnliches Verhalten wie schon das vorherige Szenario. Das NFH-NA und das NFH-NLBT System haben kaum bzw. keine korrekten Übertragungen. Nur die LBT basierten Systeme schaffen es, einen Teil der Pakete korrekt zu übertragen. Allerdings ist die Anzahl deutlich geringer als bei nur 100 % Gesamtkommunikationsanforderung. Ein besonderes Verhalten zeigt das NFH-FB System. Einige Verbindungen übertragen keine Pakete korrekt und die anderen übertragen bis zu 70 % der Pakete fehlerfrei.

Auch das Diagramm in Abbildung 47 ähnelt in gewisser Weise dem vorherigen Simulationsszenario. Beim NFH-NLBT existiert erneut die markante Grenze bei einer Sekunde. Das Perzentil 95 der Übertragungszeiten der Verbindungen des NFH-LB Systems sind höher als im vorherigen Szenario, allerdings erneut gleichgroß. Besonderheiten zeigen sich bei dem NFH-FB System. Hier schwanken die Übertragungszeiten zwischen 200 ms und 40 s.

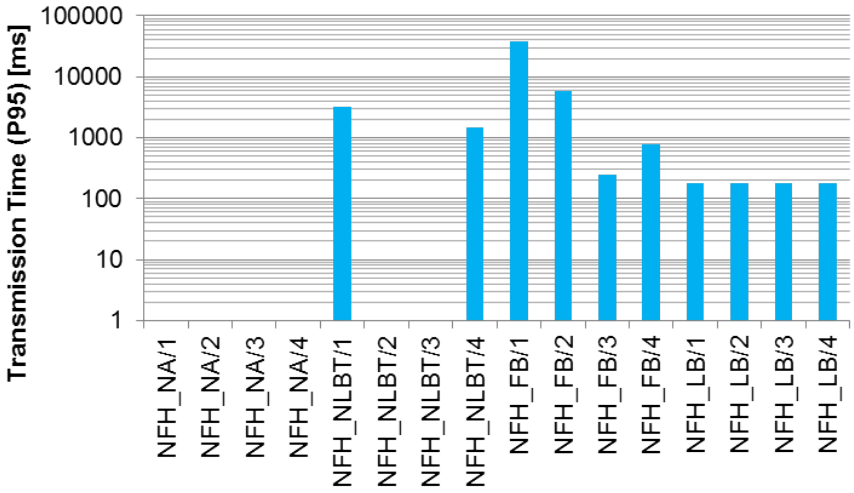

Abb. 47: *Perzentil 95 der Übertragungszeit im Simulationsszenario: "Instabilität"*

Die Ursache für das besondere Verhalten des NFH-FB Systems liegt in der zufälligen Initialisierung und in der anschließenden periodischen Übertragung. So ist es möglich, dass eine NFH-NA Verbindung jedes Mal, wenn das NFH-FB seinen CCA Check durchführt, das Medium belegt. Dadurch kann die nachfolgende Fixed Frame Period nicht zur Übertragung genutzt werden und das entsprechende Paket wird verzögert und zurückgehalten. In einem anderen Fall kann es passieren, dass das Medium bei jedem CCA Check frei ist. Für die industrielle Automation bedeutet das allerdings, dass die Systeme nicht jeden Tag dasselbe Verhalten aufweisen. Nach einem Neustart eines Systems kann dieses ganz anders agieren als zuvor. Dies ist schon ein Beispiel für ein unvorhersagbares Zeitverhalten, welches im nachfolgenden Simulationsszenario genauer betrachtet wird.

6.5.5 Simulationsszenario: „Unvorhersagbares Zeitverhalten"

Beschreibung

Im letzten Szenario wird untersucht, wie sich ein NFH-NA und ein NFH-LB bei einem Interferer, der Pakete als Burst überträgt, verhalten. Tabelle 10 listet die relevanten Parameter auf.

NFH-NA und NFH-LB besitzen jeweils 4 Verbindungen. Die Medienbelegungszeit, der Sendezeitabstand und die Paketanzahl sind unterschiedlich. Das NFH-NA erzeugt eine Kommunikationsanforderung von 25 %. Die Verbindungen des NFH-NA Systems werden in einem Abstand von 20 ms gestaffelt initialisiert, so dass das Medium alle 20 ms für 5 ms belegt ist. Das NFH-LB System erzeugt eine Kommunikationsanforderung von 100 %. Damit ergibt sich ohne Interferer eine Gesamtkommunikationsanforderung von 125 %. Die Initialisierung des NFH-LB erfolgt zufällig innerhalb der Größe des Sendezeitabstandes. Der Wert für q beträgt 32.

Der Interferer besteht aus einer Verbindung, die nach Ablauf der halben Simulationszeit

Tab. 11: *Parameter des Simulationsszenarios: "Unvorhersagbares Zeitverhalten"*

System	q	t_{onmax} [ms]	n_{Con}	t_{TI} [ms]	Req [%]	Init [ms]	Packets
NFH-NA	-	5	4	80	25%	0/20/40/60 ms	1.000
NFH-LB	32	7	4	28	100%	rand(t_{TI})	2.858
Intf., LB	4	7	1	Burst	-	39200 ms	200
					125%		

200 Pakete an einem Stück übertragen soll. Dabei handelt es sich um ein weiteres NFH-LB System. Der Wert von q liegt bei 4. Dadurch hat der Interferer eine höhere Priorität als das bereits existierende NFH-LB System, da die Wartezeiten durch ein ECCA deutlich reduziert sind.

Ergebnisse

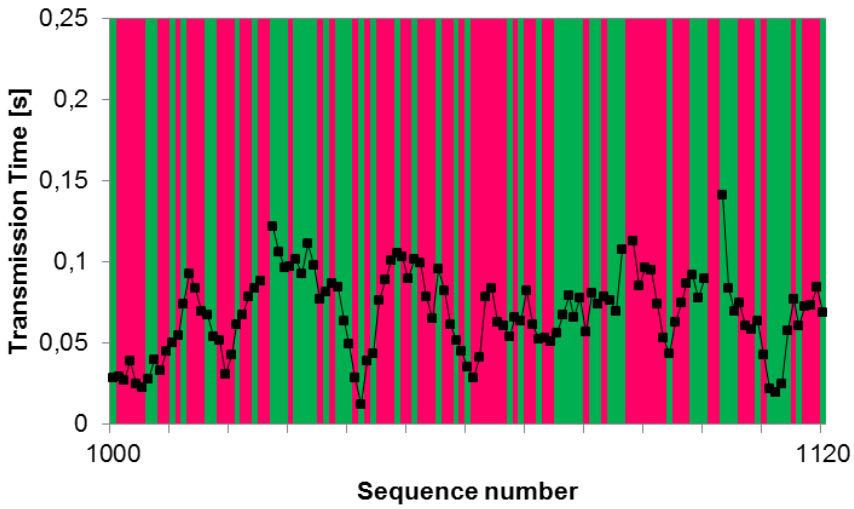

Abb. 48: *Übertragungszeiten im Simulationsszenario "Unvorhersagbares Zeitverhalten" im ungestörten Fall*

Abbildung 48 zeigt die Übertragungszeiten der einzelnen Pakete einer Verbindung (schwarze Rechtecke). Die Übertragungszeiten werden für korrekte Telegramme und Pakete mit Interferenzen aufgetragen. Für verworfene Pakete existiert kein Wert für die Übertragungszeit. Korrekte Pakete sind grün hinterlegt. Verworfene Pakete oder Telegramme mit Interferenzen haben einen roten Hintergrund. Abbildung 48 zeigt das Übertragungsverhalten im ungestörten Fall. Verworfene Pakete treten sehr selten auf. Korrekt übertragene Pakete und Pakete mit Interferenzen sind in etwa gleichmäßig verteilt. Die Übertragungszeiten sind kleiner als 150 ms.

Das Diagramm in Abbildung 49 zeigt das Übertragungsverhalten während des Bursts. Es

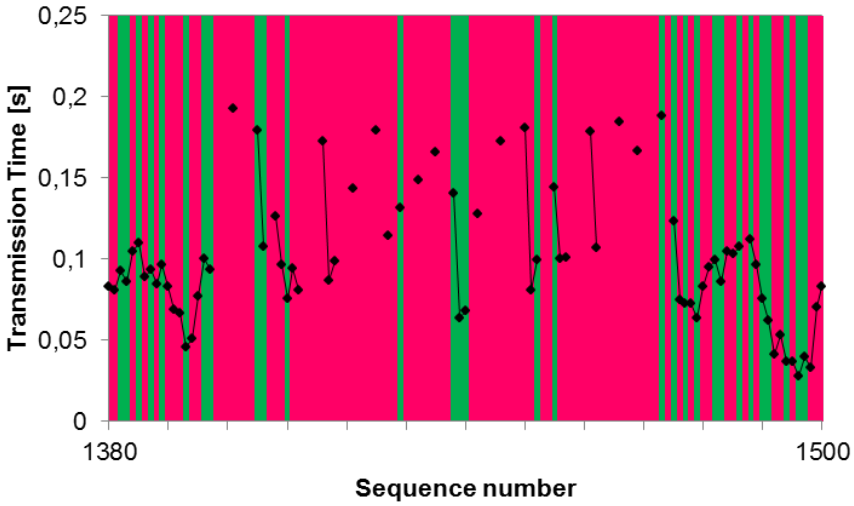

Abb. 49: *Übertragungszeiten im Simulationsszenario "Unvorhersagbares Zeitverhalten" im gestörten Fall*

werden deutlich mehr Pakete verworfen. Die Werte der Übertragungszeiten steigen bis auf 200 ms an. Nur vereinzelte Pakete werden korrekt übertragen. Der Rest wird durch Interferenzen zerstört.

Dieses Simulationsszenario veranschaulicht deutlich, dass das Zeit- und Fehlerverhalten besonders von LBT basierten Systemen nicht vorhersagbar ist. Nur ein einzelner Störer hat riesige Auswirkungen auf die Übertragung. Die Anforderungen der Automatisierungsapplikation können dadurch unter Umständen nicht mehr eingehalten werden.

6.5.6 Zusammenfassung

Die ausgewählten Simulationsszenarien haben gezeigt, dass die Anforderungen an die EN 300 328 nicht erfüllt werden. Das Spektrum wird nicht effizient genutzt. Zudem können einige Medienzugriffsverfahren das Spektrum in einem größeren Ausmaß verwenden als andere.

Die Szenarien haben zusätzliche Ergebnisse bezüglich der Anforderungen der industriellen Automation geliefert. NFH-FB Systeme besitzen eine hohe Abhängigkeit gegenüber den Anfangsbedingungen bzw. der Initialisierung. Geräte mit einen NFH-LB Zugriffsmechanismus weisen unter Umständen ein unvorhersehbares Zeitverhalten auf. Ein Kommunikationssystem mit den genannten Eigenschaften ist für die industrielle Automation nicht zulässig.

Das grundlegende Problem der EN 300 328 ist, dass Koexistenz aus der Gerätesicht erreicht werden soll. Dieser Ansatz zeichnet sich durch einen geringeren Aufwand bei der Umsetzung aus, führt allerdings zu einer ineffizienten Nutzung und zu einer ungerechten

Aufteilung des Mediums.

Hieraus muss die Schlussfolgerung gezogen werden, dass ein zentraler Ansatz mit einer entsprechenden Koexistenzplanung geeigneter ist. Dabei kommt auch der Einsatz eines Koexistenzmanagers zum Tragen. Die einzelnen Frequenznutzer können optimal über das Spektrum verteilt werden. Notwendige Abstände im Zeit- und Frequenzbereich sind plan- und umsetzbar. Kollisionen werden vermieden. Auch die Anforderungen der jeweiligen Applikationen können berücksichtigt werden. Mit dem Ansatz des Koexistenzmanagments wird das Spektrum effizient genutzt und fair aufgeteilt. Die Spezifikation IEC 62657-2 ([8]) beschreibt das Koexistenzmanagement in der industriellen Automation im Detail.

Weitere Simulationsergebnisse präsentiert [71].

7 Medienzugriffsmechanismen für die industrielle Automation

7.1 Allgemein

Da die Simulationsergebnisse in Abschnitt 6.5 gezeigt haben, dass die in der EN 300 328 definierten Medienzugriffsmechanismen für einen Einsatz in der industriellen Automation ungeeignet sind, werden in diesem Abschnitt zwei Medienzugriffsmechanismen untersucht, die den Anspruch haben, ein für die industrielle Automation notwendiges Zeit- und Fehlerverhalten zu gewährleisten. Der erste Ansatz ist MS-Aloha. Dieser stammt aus dem Bereich der Car-to-Car und Car-to-X Kommunikation. Die Anforderungen aus diesem Bereich ähneln denen der industriellen Automation. Im zweiten Fall handelt es sich um einen auf DECT basierten Ansatz, welcher vom Autor spezifiziert wurde. Bei diesem Ansatz wurde DECT um einige Funktionalitäten erweitert.

7.2 MS-Aloha

7.2.1 Funktionsweise

MS-Aloha (Mobile Slotted Aloha) wurde für die Kommunikation zwischen sich bewegenden Sendern und Empfängern entwickelt. Der primäre Anwendungsfall sind VANETs (Vehicular Ad Hoc Networks). MS-Aloha ist verbindungsorientiert und dezentralisiert. Es gibt keinen zentralen Koordinator, sodass ein Ad-hoc-Verhalten zum Tragen kommt.
Das zur Übertragung verwendete Medium wird in Zeitschlitze (Slots) unterteilt, sodass eine Synchronisierung zwischen den Teilnehmern notwendig ist (siehe Abbildung 50). Der sich zyklisch wiederholende Zeitrahmen, bestehend aus einer festgelegten Anzahl an Slots gleicher Länge, wird als Frame bezeichnet. Die Synchronisierung auf diesen Zeitrahmen ist erforderlich, damit jeder Teilnehmer zu jeder Zeit genau bestimmen kann, welche Position in der Framestruktur aktuell vorliegt.

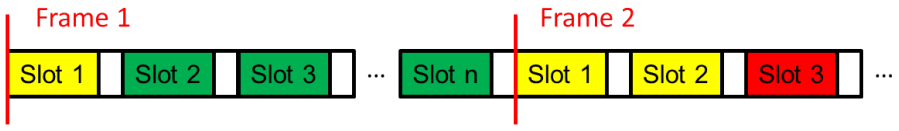

Abb. 50: *Aufbau der MS-Aloha-Framestruktur*

Die Information über den Status der einzelnen Slots (frei - grün, belegt - gelb, Kollision - rot) wird mit jedem übertragenen Paket bekannt gegeben. Der Bereich der Slotzustandsinformationen wird als „Frame Information Structure“ (FI) bezeichnet. Mit Hilfe der im FI enthaltenen Informationen, welche von jedem Teilnehmer empfangen werden, wird das

Hidden-Node-Problem beseitigt. Dieses Problem ist beispielsweise bei WLAN existent.
In der Reservierungsphase wirbt jeder sendewillige Teilnehmer um einen freien Slot. Die Wahl des Slots wird dabei durch die Abtastung des Mediums und durch die empfangenen FIs bestimmt. Ein Sendevorgang gilt automatisch als Reservierung des gleichen Slots für den nächsten Frame, sodass dieser vom selben Teilnehmer erneut verwendet werden kann. Findet in einem Slot keine Übertragung statt, wird dieser wieder als frei gekennzeichnet. Wenn eine Kollision in einem Slot erkannt wurde, müssen sich die betroffenen Teilnehmer einen neuen Slot suchen.
Zur Modellvereinfachung wird idealerweise angenommen, dass die Slotzuweisung zu jedem Frameanfang erfolgt. Dabei wird auch ausgeschlossen, dass mehrere Teilnehmer denselben Slot reservieren.
Die Verwendung eines spezifischen PHY ist bei MS-Aloha nicht vorgegeben. Es wird allerdings der PHY der IEEE 802.11p empfohlen, da dies die Funkspezifikation für die Car-to-Car-Kommunikation ist.

MS-Aloha ist in der ETSI TR 102 861 ([82]) und der TR 102 862 ([83]) spezifiziert.

7.2.2 Modellbeschreibung

Systemmodell

Das Systemmodell besteht aus einem Modul für ein Funksystem, welches entsprechend dem MS-Aloha-Medienzugriffsverfahren arbeitet. Der Unterschied zu den bisherigen Modellen ist, dass mit einem Petri-Netz-Modul gleichzeitig zehn Verbindungen realisiert werden. Den Ausgangspunkt dafür bilden Marken, denen unterschiedliche Eigenschaften zugewiesen werden. Außerdem sind den Marken feste Zeitwerte zugewiesen, auf deren Grundlage das Schalten von Transitionen ausgelöst werden kann. Im Systemmodell existiert auch ein Modul zur Nachbildung eines Störers. Wieder sind die relevanten Module über den Platz „Medium“ miteinander verbunden. Auf diesem findet auch die gegenseitige Beeinflussung statt. Die Transition „Interferenz-Check“ ermittelt mithilfe spezieller Funktionen die Kollision zweier Pakete. Abbildung 51 zeigt das mit kolorierten Petri-Netzen umgesetzte Systemmodell.

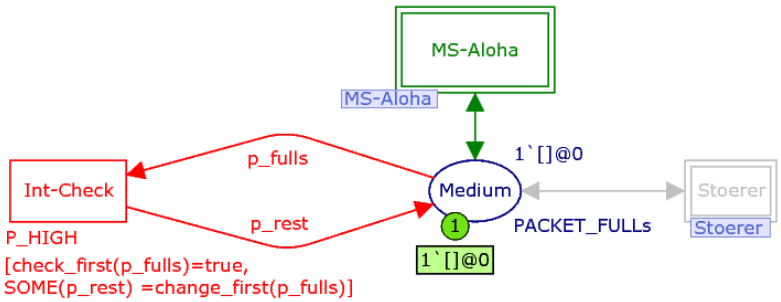

Abb. 51: *Hauptnetz MS-Aloha*

Application Layer

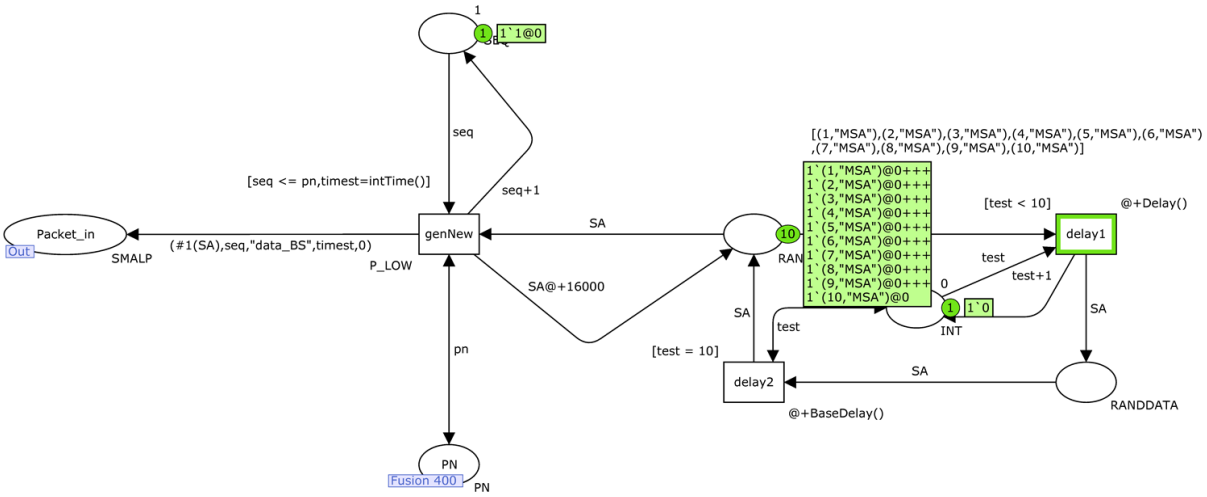

Abb. 52: *Application Layer für MS-Aloha*

Der in Abbildung 52 dargestellte Application Layer dient der Generierung und späteren Auswertung von Paketen. Es existiert für jede Verbindung eine Marke mit einer zufälligen Initialisierungszeit. Nach Erreichen der individuellen Initialisierungszeit wird ein Paket generiert. Gleichzeitig wird eine Marke erzeugt, die nach Ablauf des Sendezeitabstandes die Generierung eines weiteren Paketes auslöst. Die Übertragung der Pakete an den DLL findet über den Platz „Packet_in“ statt. Die Auswertung des Status und der Kenngrößen der Pakete sowie die Aufzeichnung der Simulationsergebnisse mit Hilfe von Monitor-Funktionen ist nicht in Abbildung 52 dargestellt.

Data Link Layer

Der Data Link Layer dient als Brücke zwischen der Paketerzeugung und der Übertragung über das Medium. In Abbildung 53 ist er grafisch dargestellt. Es befinden sich im Petri-Netz des DLLs drei zusätzliche Subtransitionen, die jeweils ein weiteres Petri-Netz verbergen. Wird ein Paket zum DLL übertragen, so muss zuerst ein entsprechender Slot zugewiesen werden. Die Zuweisung übernimmt die Subtransition „Slots“. Dabei ergreift ein ankommendes Paket einen freien Slot, sofern ein solcher vorhanden ist. Die Auswahl aus den freien Slots erfolgt zufällig. Wenn ein Slot aus der Liste der freien entwendet wurde, so ist dieser Slot des Frames für den jeweiligen Knoten reserviert. Im Fall, dass ein Slot für einen Knoten reserviert ist und dieser ein weiteres Paket übertragen möchte, so darf die Marke dieses Paketes ohne eine erneute Reservierung die Subtransition passieren.

Die Pakete werden in den Buffer des DLLs geleitet. Der Buffer besitzt nur Platz für ein Paket jeder Verbindung. Befindet sich von einem Knoten bereits ein Paket im Buffer und sendet dieser ein neues Paket, so wird das ältere der beiden verworfen.

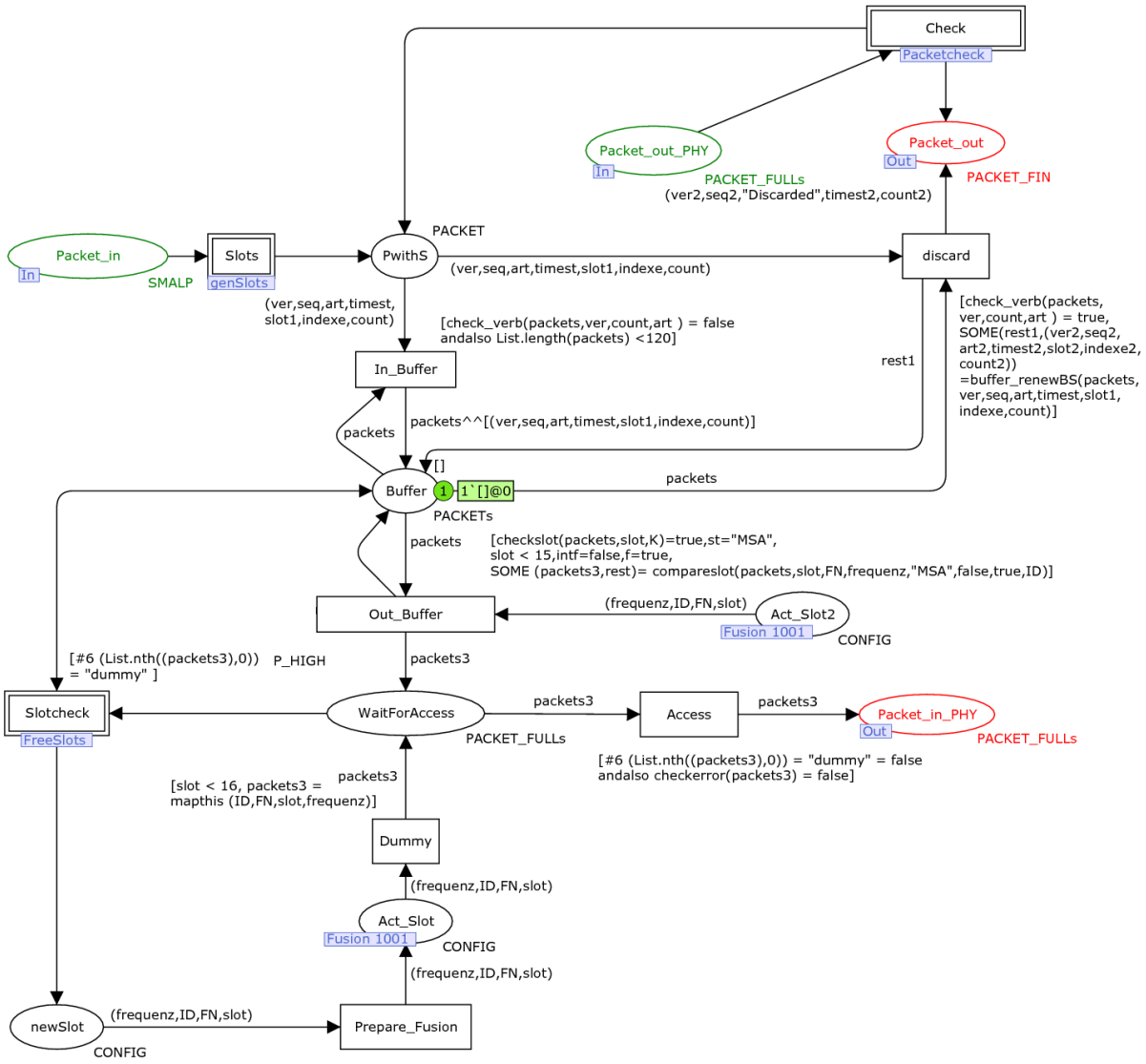

Abb. 53: *Data Link Layer für MS-Aloha*

Wenn sich am Ende des Frames noch Pakete im Buffer befinden, so wird für den Sender der gewählte Sendeslot auch für den nächsten Frame reserviert. Slots, die nicht für den nächsten Frame belegt sind, werden wieder freigegeben. Die Steuerung über die Freigabe der Slots übernimmt die Subtransition „Slotcheck“, in welcher auch der Slotkreislauf des Funkmodells umgesetzt ist.

Stimmen Slotnummer im Paket und Slotnummer aus dem Kreislaufmechanismus überein, so darf das Paket an den Physical Layer weitergegeben werden. Dort wird das Paket auf das Medium gelegt und auf Interferenz hin überprüft. Im Anschluss wird das geprüfte Paket wieder an den DLL übergeben. Die Subtransition „Check“ kümmert sich um die Auswertung des Interferenz-Attributes der Pakete. Der Wert des Interferenz-Attributes kann hier „True“ oder „False“ sein. Bei Interferenz wird das Paket einmalig wiederholt und bei nochmaliger Interferenz als zerstört betrachtet. Zerstörte, korrekt übertragene und verworfene Pakete werden an den Application Layer zurückgegeben.

Physical Layer

Der Physical Layer legt die zu übertragenden Pakete auf das Medium. Diese verbleiben dort, bis die Medienbelegungsdauer erreicht ist. Als Modell dient das bereits eingeführte.

Störer

Das Netz des Störers lässt sich als Erweiterung des Physical Layers beschreiben. Hier wird ein und dasselbe Paket wiederholt auf das Medium gelegt. Die Frequenz, die Medienbelegungsdauer und die Pausenzeit können individuell angepasst werden. Abbildung 54 zeigt das Netz des Störers.

7.2.3 Simulationen

Für die Simulation wird angenommen, dass ein MS-Aloha-System aus zehn Verbindungen besteht und für jede Verbindung Datenpakete im Abstand von 16 ms generiert werden. Die zehn Verbindungen werden durch Marken mit unterschiedlichen Verbindungsnummern in nur einem Petri-Netz-Verbindungsmodell erzeugt. Die Initialisierungszeit ist für alle Verbindungen innerhalb des Sendezeitabstandes zufällig. Jeder Knoten sendet 1.000 Datenpakete. Die Medienbelegungszeit beträgt für jeden Slot 700 µs und weitere 300 µs dienen als Gap. Für die Slotlänge ergibt sich dadurch eine Zeit von 1000 µs. Ein Frame umfasst zehn solcher Slots. Bei der Erkennung von Kollisionen wird das betroffene Paket einmalig in einem der folgenden freien Slots, welcher zufällig bestimmt wird, wiederholt. Wird ein bereits kollidiertes Paket nochmal einer Interferenz ausgesetzt, so gilt dieses als zerstört. Sollte für eine Verbindung ein neues Paket generiert werden, obwohl sich noch ein Paket dieser Verbindung im Buffer befindet, wird das veraltete verworfen.

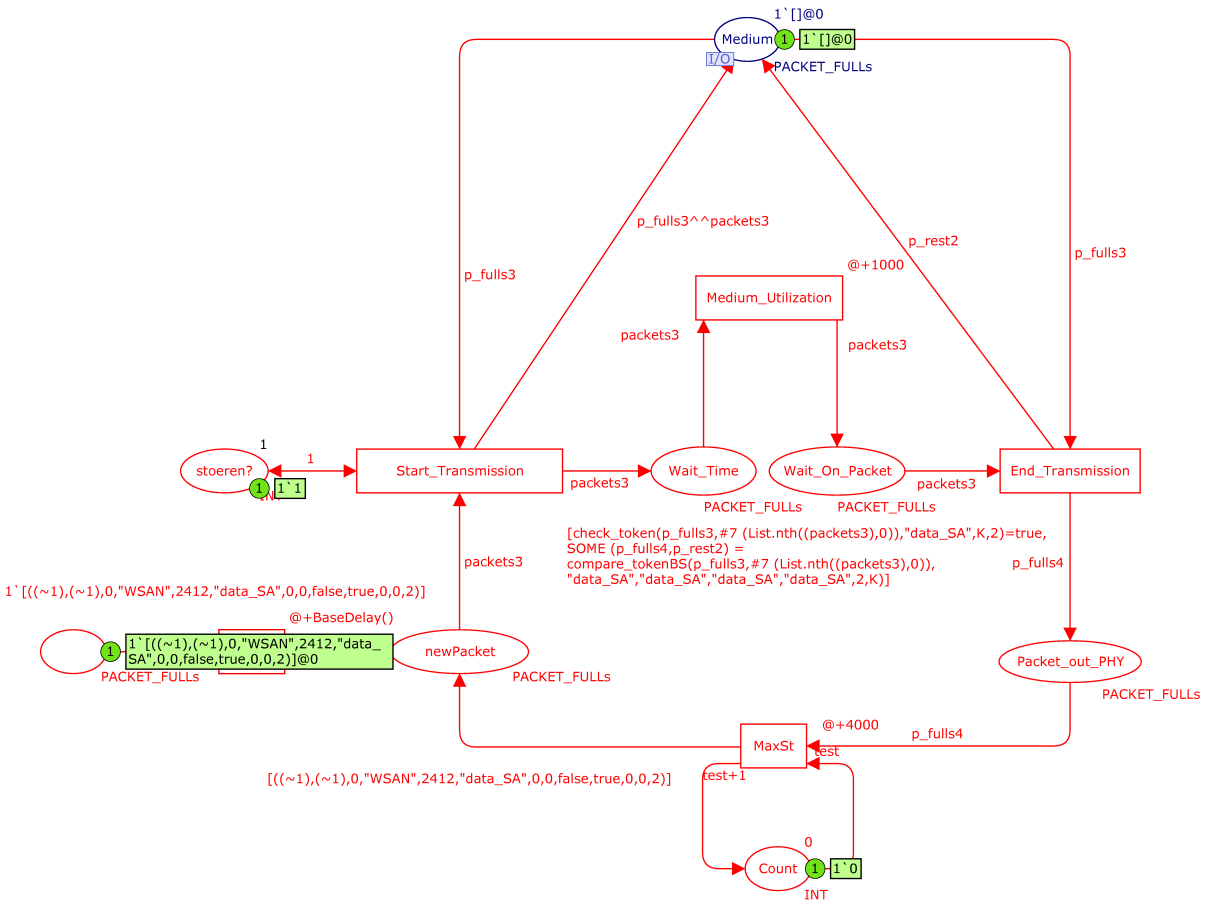

Abb. 54: *Störer*

Der Störer repräsentiert einen weiteren unabhängigen Funkteilnehmer mit fester Frequenz. Der Sendezeitabstand der Störpakete entspricht den bei PROFINET üblichen und in der Automatisierungstechnik häufig angewendeten 4, 8, 16, 32 und 64 ms. Auch beim Störer ist die Medienbelegungszeit 700 µs. Die Initialisierung erfolgt zufällig. Dadurch sendet der Störer asynchron zum eigentlichen Nutzdatensystem.

Abbildung 55 zeigt den Status der übertragenen Nutzdatenpakete für unterschiedliche Sendezeitabstände des Störers. Selbst im ungestörten Fall werden bei der gewählten Konfiguration nicht alle Pakete fehlerfrei übertragen. Mit kleiner werdendem Sendezeitabstand des Störers nimmt der Anteil an zerstörten und verworfenen Paketen zu. Allerdings werden auch bei einem Sendezeitabstand von 4 ms mehr als 86 % der Pakete korrekt übertragen. MS-Aloha weist damit ein sehr robustes Verhalten auf.
Abbildung 56 zeigt die Verteilung der Übertragungszeit des Nutzsystems für unterschiedliche Sendezeitabstände des Störers und im ungestörten Fall. Die Mehrheit der Pakete befindet sich im Bereich unter 17 ms. Die Verteilung ähnelt einem Dreiecksverlauf, wobei dieser mit kleiner werdendem Sendezeitabstand des Störers verformt wird. Je kleiner der Sendezeitabstand des Störers ist, desto mehr Nutzdatenpakete weisen eine größere Übertragungszeit auf. Für einige Anwendungsfälle der industriellen Automation überschreiten die ermittelten Übertragungszeiten das maximal zulässige Zeitlimit.

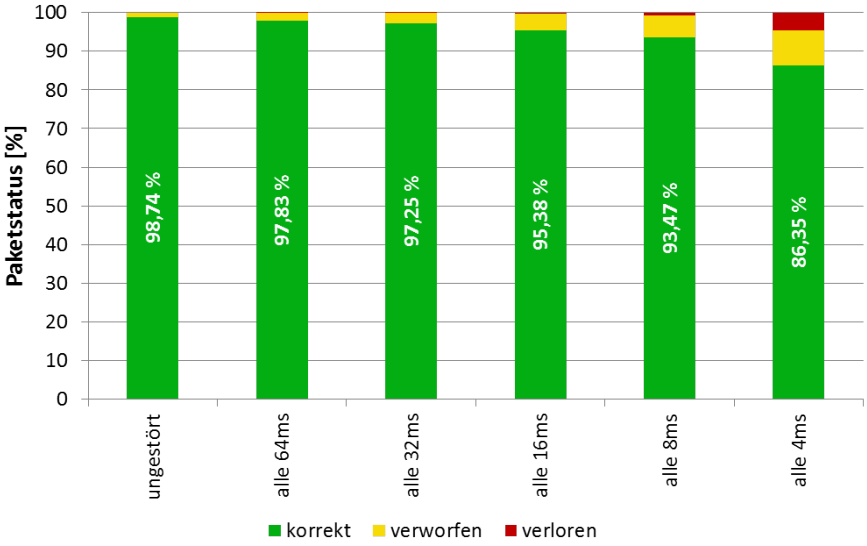

Abb. 55: *Paketverteilung für MS-Aloha mit und ohne Störer*

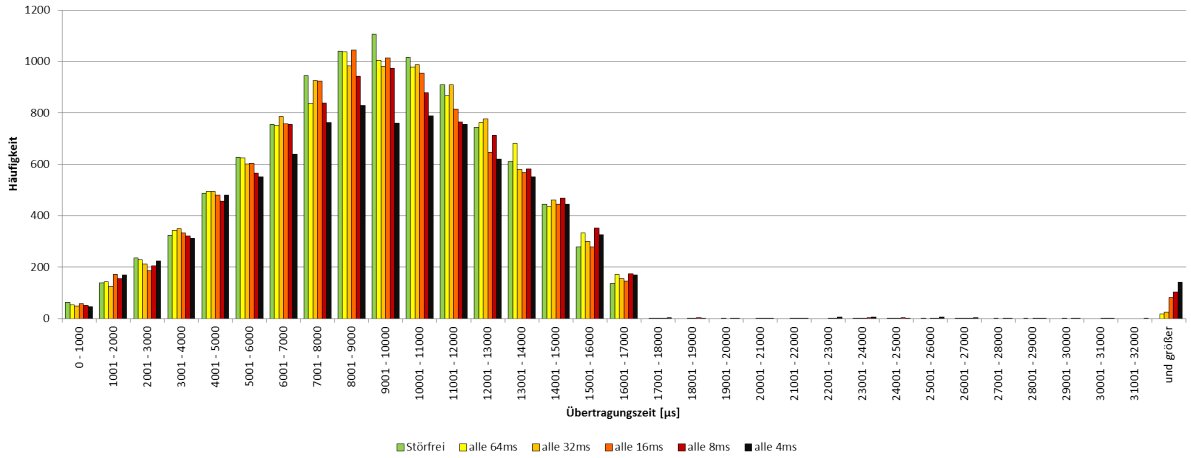

Abb. 56: *Verteilung der Übertragungszeit für MS-Aloha mit und ohne Störer*

Die Simulationsergebnisse zeigen, dass MS-Aloha sehr robust gegenüber Störern ist. Die gewählte Konfiguration weist ein nicht optimales Zeitverhalten auf. Ein weiterer Nachteil von MS-Aloha ist, dass bei einem schmalbandigen Störer nicht in einen alternativen Frequenzkanal gewechselt werden kann. Obwohl Interferenzen erkannt wurden, können weitere Kollisionen bestenfalls zeitmäßig vermieden werden. Bei einem kontinuierlichen Störer würde es allerdings zu einer permanenten Beeinträchtigung der Übertragung kommen.

7.3 Neuer Ansatz für die IA

7.3.1 Herleitung der Funktionsweise

Der neue Ansatz für einen Medienzugriffsmechanismus der IA verwendet als Ausgangspunkt DECT (Digital Enhanced Cordless Telecommunications). DECT ist in der EN 300 175 spezifiziert. Für den Medienzugriff sind die Teile 1 bis 3 relevant ([86], [87], [88]). DECT wird zur digitalen, schnurlosen Telekommunikation verwendet. Ein Frequenzband zwischen 1,88 und 1,9 GHz ist eigens für DECT vorgesehen. In diesem Frequenzband sind zehn Kanäle definiert. DECT besitzt wie MS-Aloha eine Framestruktur bestehend aus Slots. Diese Slots sind allerdings nicht nur festen Zeiten zugeordnet, sondern auch festen Frequenzkanälen. Bei DECT wird in Fixed Terminal (FT) und Portable Terminal (PT) unterschieden. FT entspricht dabei einer Basisstation und PT den angebundenen Clients. DECT ist ein verbindungsorientiertes Protokoll. Bei einem Verbindungsaufbau wird für die Dauer eines Frames DFS (Dynamic Frequency Selection) durchgeführt. Mit DFS wird für jeden Slot überprüft, welcher Kanal das niedrigste Interferenzlevel besitzt. Dieser Kanal wird dann dem Slot fest zugewiesen. Abbildung 57 zeigt die Zuordnung von Kanälen und Teilnehmern zu Slots. Die erste Hälfte der Slots eines Frames dient der Übertragungsrichtung von den PTs zu dem FT (Uplink). In der zweiten Hälfte wird die entgegengesetzte Richtung realisiert (Downlink). Die Zuweisung von Frequenzkanälen und Teilnehmern zu Slots ist statisch.

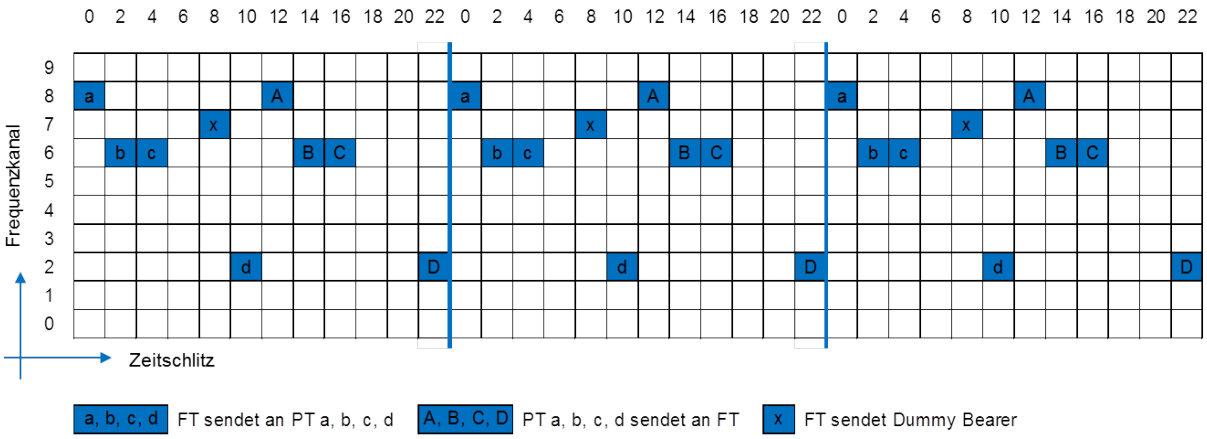

Abb. 57: *Slotzuweisung bei DECT ([61])*

Den Wechsel vom verbindungsorientierten zum paketorientierten Protokoll erreicht CLDPS (Connectionless DECT Packet Service) von der Firma Höft & Wessels ([61]). Bei CLDPS wird die feste Aufteilung in Up- und Downlink beseitigt. Auch die statische Zuweisung von Teilnehmern zu Slots wird aufgehoben (siehe Abbildung 58). Die Teilnehmer können die Slots eines Systems flexibel nutzen, auch innerhalb eines Frames. Der Nachteil an CLDPS ist, dass die Zuweisung von Frequenzkanälen zu Slots weiterhin statisch ist.

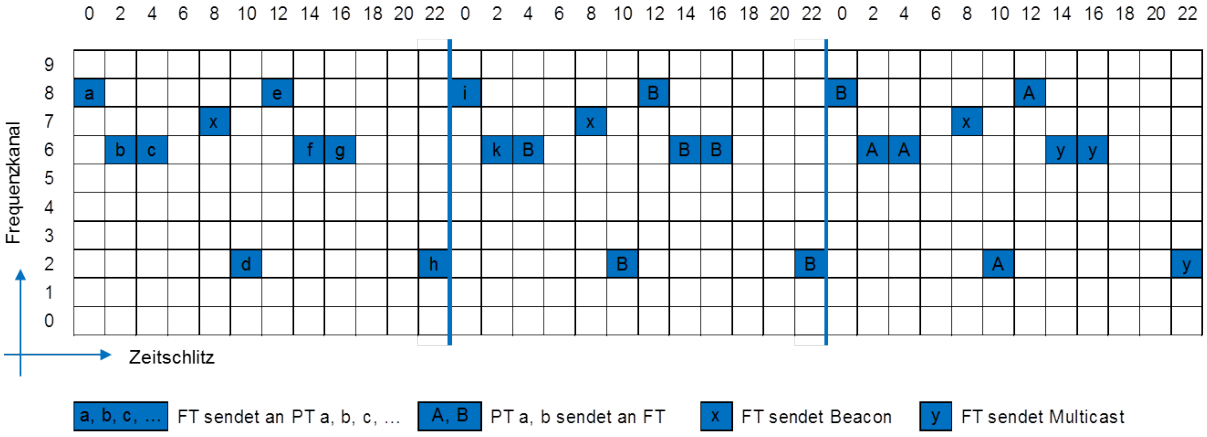

Abb. 58: *Slotzuweisung bei CLDPS ([61])*

Wie die Untersuchungen an der EN 300 328 (siehe Kapitel 6) und deren Gerätesicht gezeigt haben, ist ein Systemansatz für ein Medienzugriffsverfahren der IA notwendig. So sollen bei dem Ansatz für die IA die einzelnen Slots nicht nur für einzelne Teilnehmer, sondern für sämtliche Teilnehmer eines Systems verfügbar sein. In diesem Zusammenhang meint ein System die Menge an Teilnehmern eines Netzwerks. Zur Reduzierung der Kollisionsgefahr sollen sich zusätzlich die einzelnen Systeme zeitmäßig aufeinander synchronisieren. Die Synchronisation ist Teil des Einschaltvorgangs, welcher in Abbildung 59 dargestellt ist.

Beim Einschaltvorgang wird das Band als erstes auf vorhandene Systeme gescannt. Wird

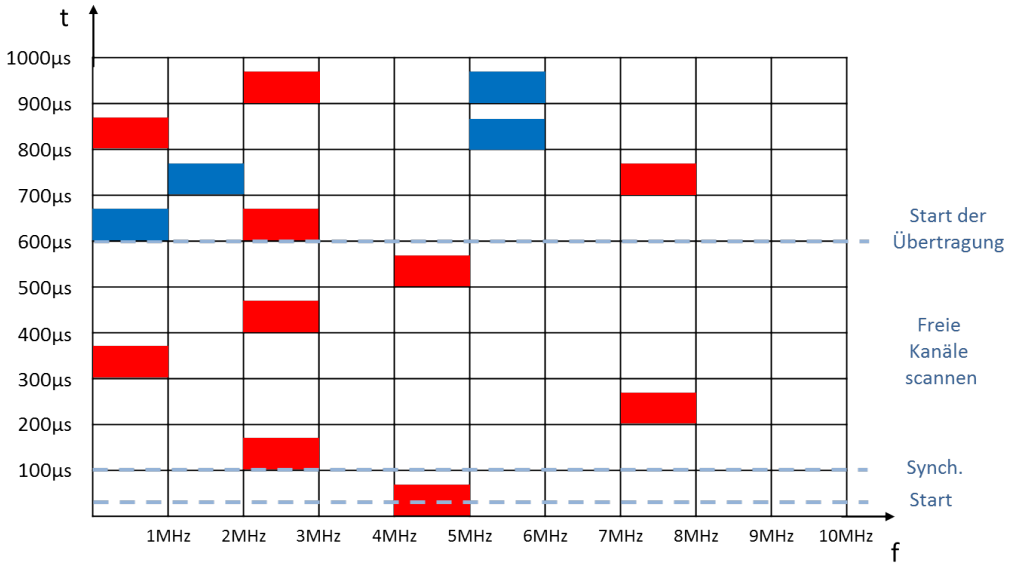

Abb. 59: *Einschaltvorgang*

ein anderes System erkannt, wird sich durch Ermittlung des Paketanfangs auf die zeitliche Slotstruktur synchronisiert. Nun wird für die Dauer eines Frames DFS ausgeführt. Dabei werden für jeden Slot die freien Frequenzkanäle ermittelt. Aus den freien Kanälen wird dann zufällig ein Kanal für den jeweiligen Slot ausgewählt und zugewiesen. Hierbei besteht die Gefahr, dass die vorhandenen Systeme nicht in jedem Slot Pakete übertragen. In diesem Fall kann ein hinzukommendes System für einen Slot einen bereits reservierten Kanal auswählen, wodurch Paketkollisionen entstehen. Nach Abschluss des Scanvorgangs für die Dauer eines Frames und der damit verbundenen Zuweisung von Kanälen zu Slots beginnt die Nutzdatenübertragung.

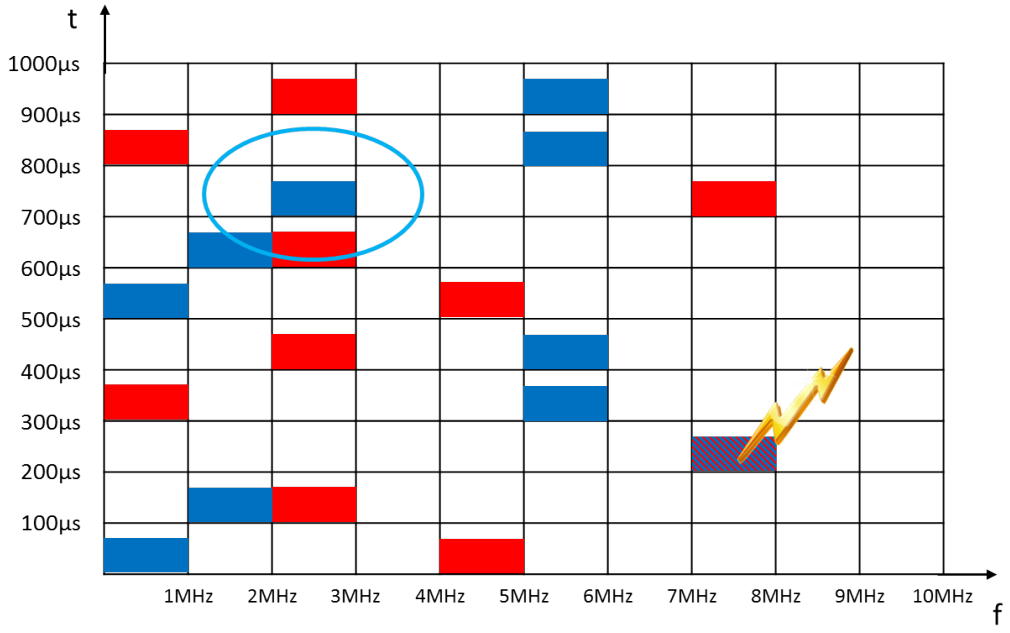

Abb. 60: *Verhalten bei Kollision*

Abbildung 60 zeigt das Verhalten beim Auftreten einer Paketkollision. Bei einer Paketkollision und deren Erkennung durch die betroffenen Systeme wird mit einer Wahrscheinlichkeit von 50 % für diesen Slot ein anderer freier Kanal gewählt. Um die Liste der freien Kanäle aktuell zu halten, scannen die Geräte eines Systems, die in einem Slot nicht senden oder empfangen, das Frequenzband auf freie Kanäle. Die Liste wird unter den Teilnehmern des Systems verbreitet.

7.3.2 Modellbeschreibung

Der Modellansatz ist identisch zu dem von MS-Aloha. Auch der Aufbau des Application und Physical Layer ist gleichgeblieben. Einige wenige Änderungen sind beim DLL von Nöten. Ebenfalls ist die Implementierung eines Modules für die Kanalzuordnung notwendig geworden.
Im DLL existiert neben dem Buffer ein Mechanismus zur Realisierung des Slotkreislaufes. Ein Paket wird nur dann gesendet, wenn ein freier Slot zur Verfügung steht. Pro Slot kann nur ein Paket gesendet werden. Es gilt das FIFO-Prinzip. Der DLL weist jedem Slot den aktuell gültigen Kanal aus der Frequenzliste zu. Die Frequenzliste wird von dem Kanalzuordnungsmodul bereitgestellt und fortlaufend aktualisiert. Steht ein Slot zur Verfügung, kann das Paket an den PHY übergeben werden, in welchem der Medienzugriff stattfindet. Ein vom Physical Layer zurückgegebenes Paket wird auf Interferenzen überprüft und gegebenenfalls einmal wiederholt.
Der DLL bietet ebenfalls die Möglichkeit zur Erzeugung von Beacons. Dieser Modellbestandteil ist zur Synchronisation der einzelnen Systeme auf die gemeinsame Framestruktur notwendig. Ein Beacon wird von jedem System einmal in jedem Frame ausgesendet.

Das Kanalzuordnungsmodul, welches in Abbildung 61 dargestellt ist, dient der Erzeugung und Bearbeitung der bereits erwähnten Frequenzliste. Bevor ein Knoten mit der Übertragung von Paketen beginnt, muss zuvor eine Liste mit Frequenzkanälen aufgebaut werden (Transition „Add_Freq“ und Platz „Gen_Frequencies“). Die Listenlänge ist dabei gleich der Anzahl der verwendeten Slots. Jedem Slot wird ein eigener, zufälliger Frequenzkanal zugewiesen. Die Liste wird dem DLL zur Verfügung gestellt. Jedem Slot kann so der passende Frequenzkanal herausgesucht und zugewiesen werden. Im unteren Teil der Abbildung wird bei jedem Slot des Frames das Medium abgetastet (Transition „Check Free Channels“) und jeder als frei erkannte Kanal wird in eine weitere Liste („freechannels“) übertragen (Transition „Update Free Channels“). Sollte es zu einer Kollision zweier Pakete kommen, so wird mit einer Wahrscheinlichkeit von 50 % der genutzte Frequenzkanal für diesen Slot gewechselt (Transition „Maybe_Delete“ und Transition „full“). Der neue Kanal wird dabei zufällig aus der Liste der noch freien Kanäle ausgewählt. Die interne Frequenzliste wird erneuert.

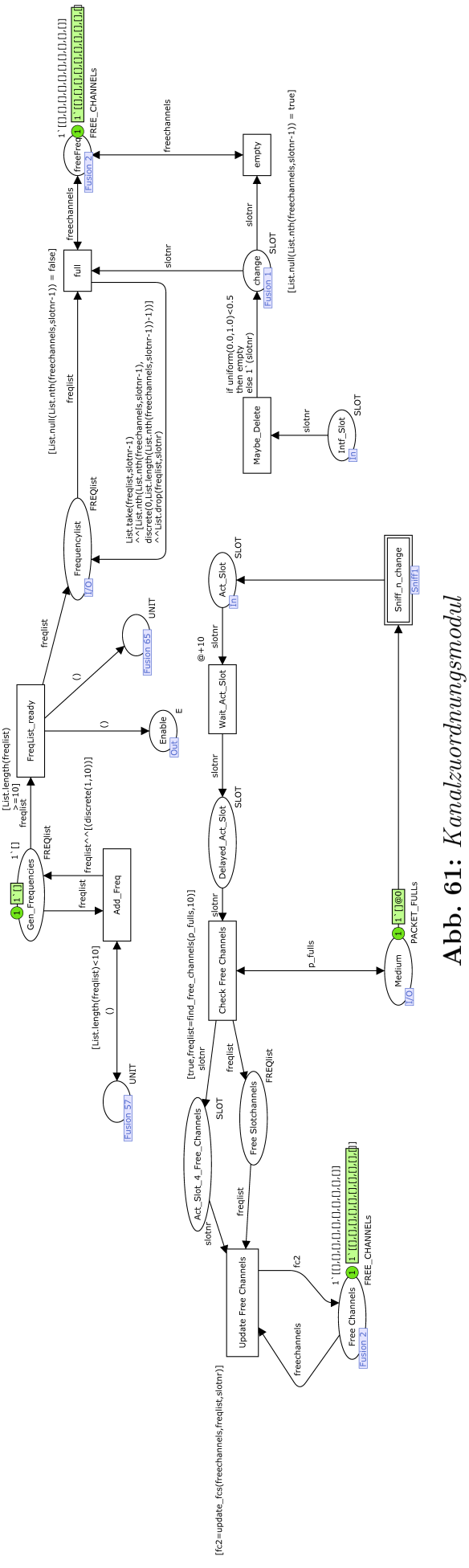

Abb. 61: *Kanalzuordnungsmodul*

Für das System sind zwei verschiedene Modi vorgesehen:

- Zufällige Generierung der Frequenzliste ohne vorherige Analyse der Medienbelegung (ohne Initialisierungszeit)
- Zufällige Generierung der Frequenzliste auf Grundlage einer vorherigen Analyse der Medienbelegung (mit Initialisierungszeit)

Die Auswahl wird in der Subtransition „Sniff_n_Change“ getroffen, welche in Abbildung 62 dargestellt ist.

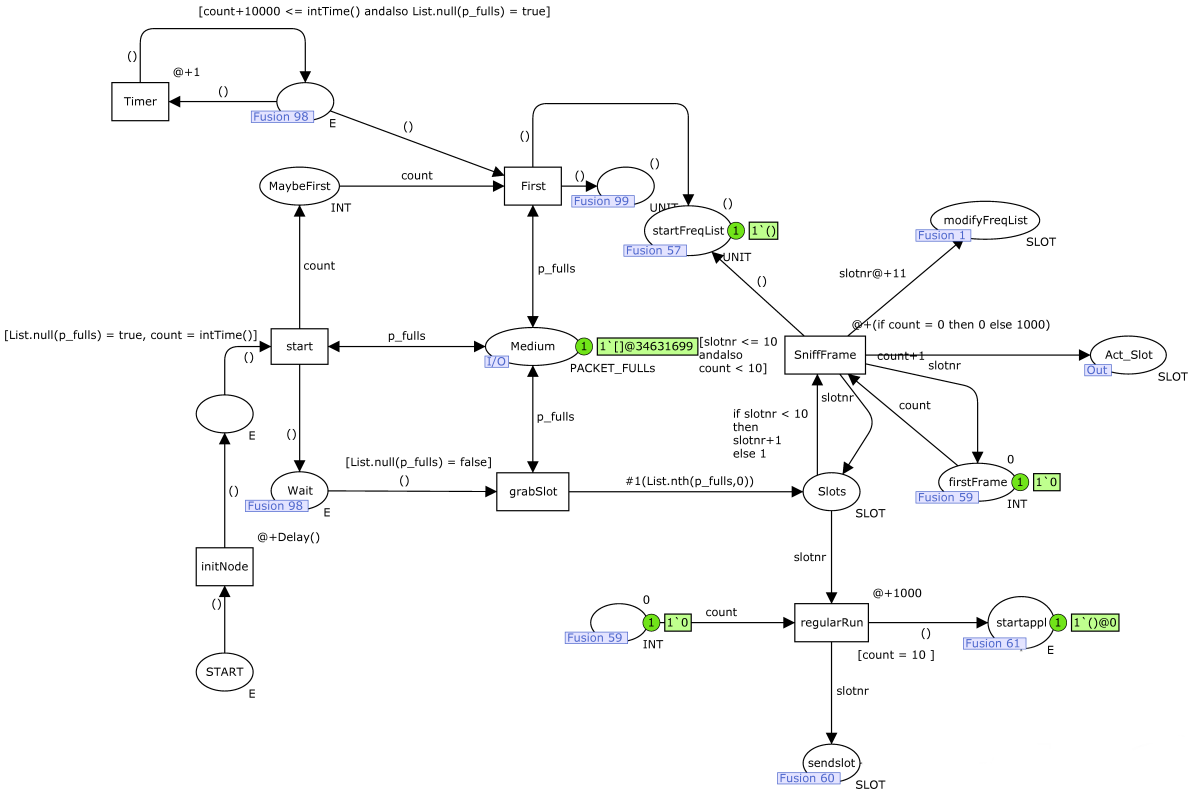

Abb. 62: *Subtransition „Sniff_n_Change“*

Für den erstgenannten Modus wird dem Platz „START“ keine Initialmarke zugeordnet. Stattdessen schaltet die Transition „First“ zum Zeitpunkt Null. Als Konsequenz wird im Petri-Netz des Kanalzuordnungsmoduls die Transition „Add_Freq“ wiederholt aktiviert und geschaltet, solange bis eine zufällige Frequenzliste vorgegebener Länge generiert worden ist. Nach erfolgreicher Fertigstellung beginnt die Verbindung umgehend mit der Erzeugung von Datenpaketen.
Für die Auswahl des zweiten Modus wird eine Initialmarke auf den Platz „START“ gelegt. Diese Marke hat zur Folge, dass für die jeweilige Verbindung eine zufällige Initialisierungszeit festgelegt wird. Nach deren Ablauf beginnt die Verbindung mit der Beobachtung des Mediums. Sollte innerhalb von 10 ms kein Paket auf das Medium gelegt worden sein,

so wird eine zufällige Frequenzliste erstellt und mit der Generierung von Datenpaketen begonnen. Sollte bereits eine Verbindung aktiv sein, wird diese anhand der übertragenen Pakete erkannt. Die Entdeckung eines Paketes führt zum Schalten der Transition „grabSlot“ und zur Weitergabe der im Paket enthaltenen aktuellen Slotnummer an den Platz „Slots“. Der Slot aus dem erkannten Paket gilt als Ausgangspunkt für die Überprüfung der Belegung der nachfolgenden Slots. Dabei wird für jeden Slot untersucht, welche Kanäle unbelegt sind. Aus der Menge der freien Kanäle wird zufällig einer ausgewählt und in die Frequenzliste geschrieben. Die Überprüfung endet nach der Dauer eines Frames. Daraufhin wird die Erzeugung von Datenpaketen freigegeben.

7.3.3 Simulation

Im APPL werden für zehn Verbindungen eines einzelnen Systems Datenpakete generiert. Zu Beginn wird von jeder Verbindung eine zufällige Initialisierungszeit abgewartet. Die Initialisierungszeit liegt dabei in einem Bereich zwischen Null und dem gewählten Sendezeitabstand. Die Medienbelegungszeit beträgt für jeden Slot 700 µs und weitere 300 µs dienen als Gap. Für die Slotlänge ergibt sich dadurch eine Zeit von 1000 µs. Ein Frame umfasst zehn solcher Slots. Bei der Erkennung von Kollisionen wird das betroffene Paket einmalig wiederholt. Dazu wird das zu wiederholende Paket an die letzte Stelle des Zwischenspeichers gelegt. Nach der Zuweisung zu einem freien Slot wird das Paket auf das Medium gelegt. Wird ein bereits kollidiertes Paket nochmal einer Interferenz ausgesetzt, so gilt dieses als zerstört. Sollte für eine Verbindung ein neues Paket generiert werden, obwohl sich noch ein Paket dieser Verbindung im Buffer befindet, wird das veraltete verworfen. In einem Simulationsszenario nutzen zehn Systeme gleichzeitig das Medium. Jedes System kann bis zu zehn Kanäle verwenden. Folgende Simulationsszenarien sollen betrachtet werden:

- Einschwingverhalten für eine Initialisierung der Systeme innerhalb des Sendezeitabstandes
- Einschwingverhalten für eine Initialisierung der Systeme innerhalb von 100 ms
- Übertragungszeitverhalten im eingeschwungenen Zustand
- Reaktion eines eingeschwungenen Systems auf einen schmalbandigen Störer

Im letzten Simulationsszenario wird der Störer zufällig initialisiert. Der Störer weist ebenfalls eine Slotstruktur mit 700 µs Medienbelegungszeit und weiteren 300 µs als Gap-Zeit auf. Kollisionen werden vom Störer nicht erkannt. Zudem sendet der Störer nur auf einem Kanal und kann diesen auch nicht ändern. Der Sendezeitabstand der Störpakete beträgt zwischen 1 ms und 32 ms. Die Nutzdatensysteme verwenden zehn Kanäle. Allerdings steht noch ein 11. Kanal zum Ausweichen zur Verfügung. Die Verbindungen der Nutzdatensysteme sind auf einen Sendezeitabstand von 10 ms fest eingestellt.

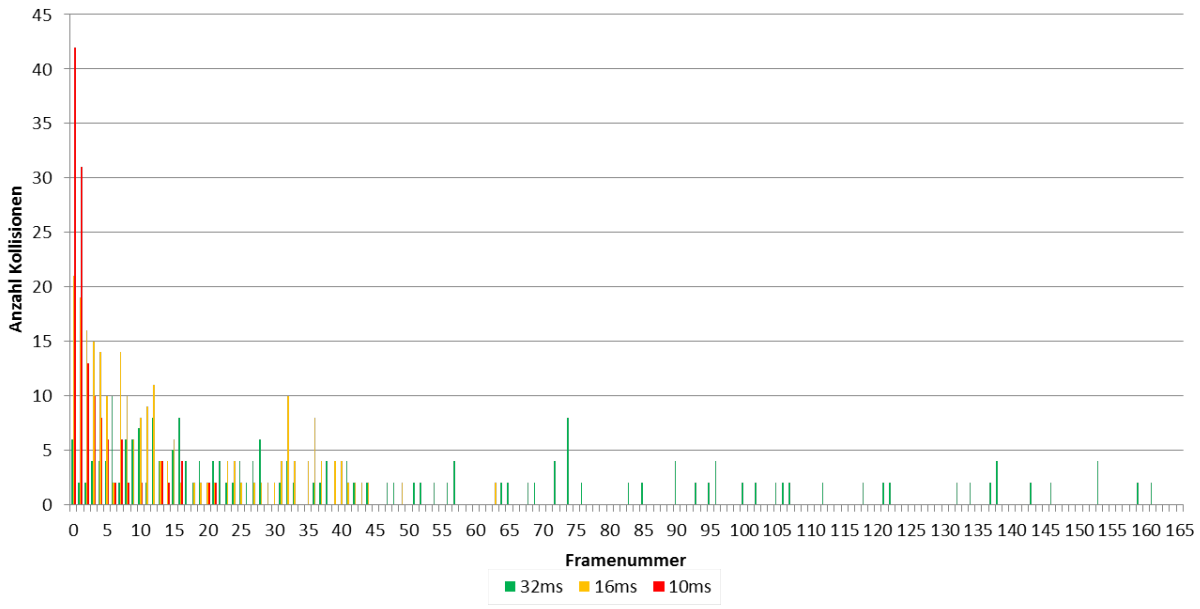

Abb. 63: *Kollisionsverteilung für eine zufällige Initialisierungszeit innerhalb des Sendezeitabstandes*

Abbildung 63 stellt für jedes Frame die Anzahl an Paketkollisionen dar. Das Interval für die Initialisierungszeit der Systeme entspricht dem eingestellten Sendezeitabstand der Verbindungen. Für die zehn Systeme betragen die Sendezeitabstände der Verbindungen 32 ms, 16 ms oder 10 ms. Es ist zu erkennen, dass die Kollisionswahrscheinlichkeit innerhalb eines Frames mit sinkendem Sendezeitabstand zunimmt. Durch den Wechsel eines Kanals nach einer Kollision mit einer Wahrscheinlichkeit von 50 % schwingen sich die Systeme selbstständig ein. Je kleiner der Sendezeitabstand ist, desto weniger Frames werden benötigt, damit der eingeschwungene Zustand erreicht wird. Für einen Sendezeitabstand von 10 ms werden etwa 20 Frames benötigt, für 16 ms ca. 65 und für 32 ms etwa 160. Wenn sich die Systeme eingeschwungen haben, treten keine weiteren Kollisionen auf.

Abbildung 64 zeigt bei einem Initialisierungszeitinterval von 100 ms die Kollisionsverteilung für unterschiedliche Sendezeitabstände. Auch in diesem Diagramm ist zu erkennen, dass der eingeschwungene Zustand mit kleinerem Sendezeitabstand schneller erreicht wird. Im Vergleich zu Abbildung 63 ist auch die Menge der Kollisionen geringer. Die Ursache liegt in der geringeren Wahrscheinlichkeit, dass mehrere Systeme gleichzeitig das Medium auf Belegung prüfen, bevor die Übertragung aktiviert wird.

Abbildung 65 zeigt die Verteilung der Übertragungszeit für unterschiedliche Sendezeitabstände im eingeschwungenen Zustand. Das Diagramm zeigt, dass die Pakete bei einem größeren Sendezeitabstand schneller übertragen werden. Bei einem kleinen Sendezeitabstand ist die Wahrscheinlichkeit höher, dass mehrere Pakete gleichzeitig im Buffer liegen. Diese werden dann nacheinander gemäß des FIFO-Prinzips übertragen. Die Pakete am Ende der Warteschlange verweilen eine längere Zeit im Buffer.

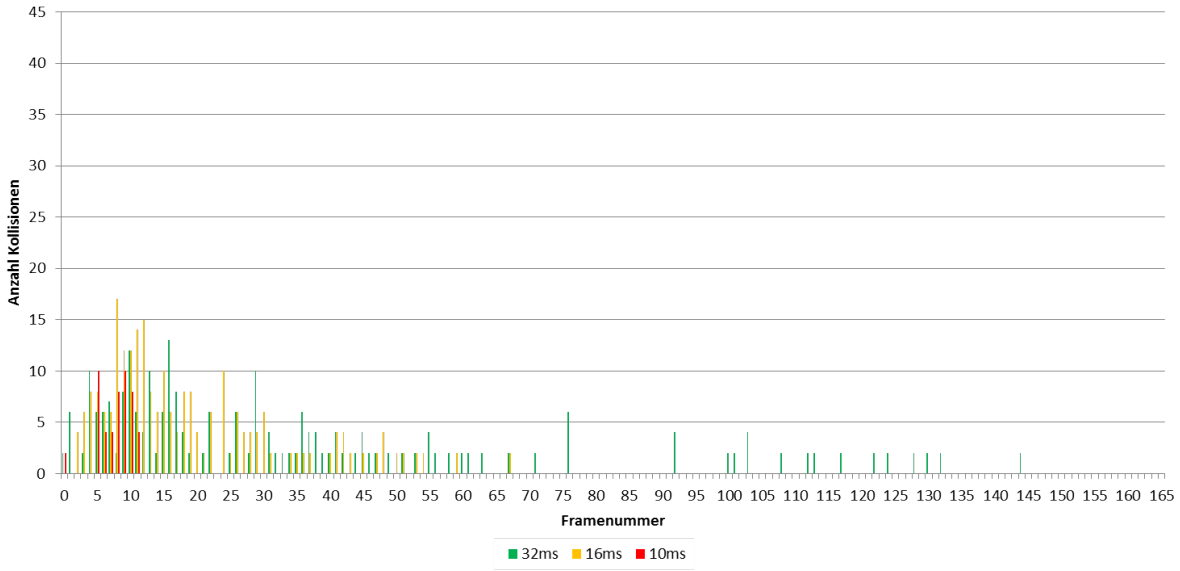

Abb. 64: *Kollisionsverteilung für eine zufällige Initialisierungszeit innerhalb von 100 ms*

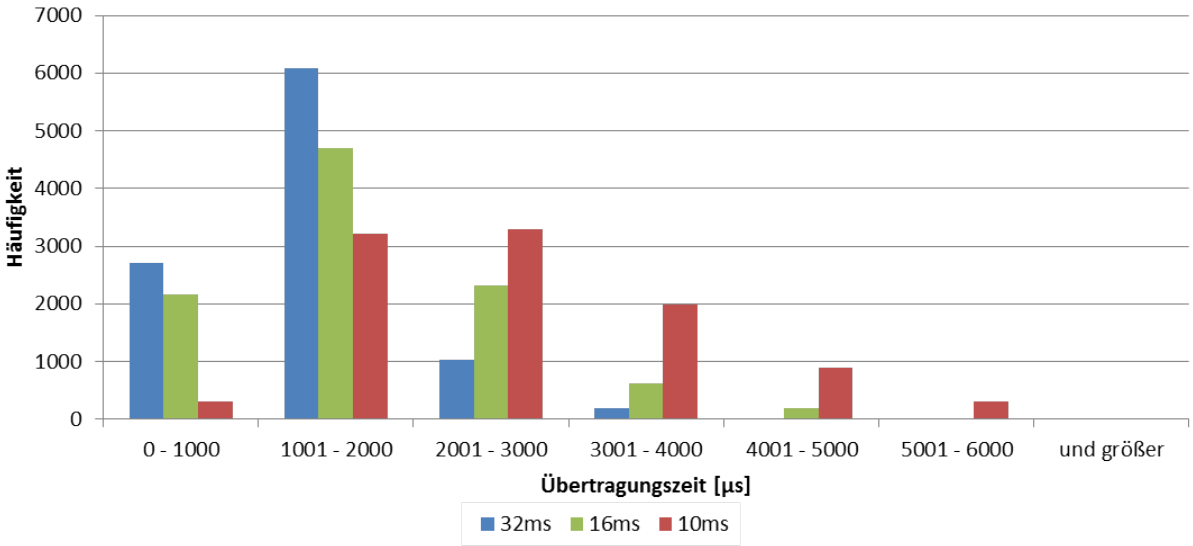

Abb. 65: *Verteilung der Übertragungszeit in Abhängigkeit von Sendezeitabstand*

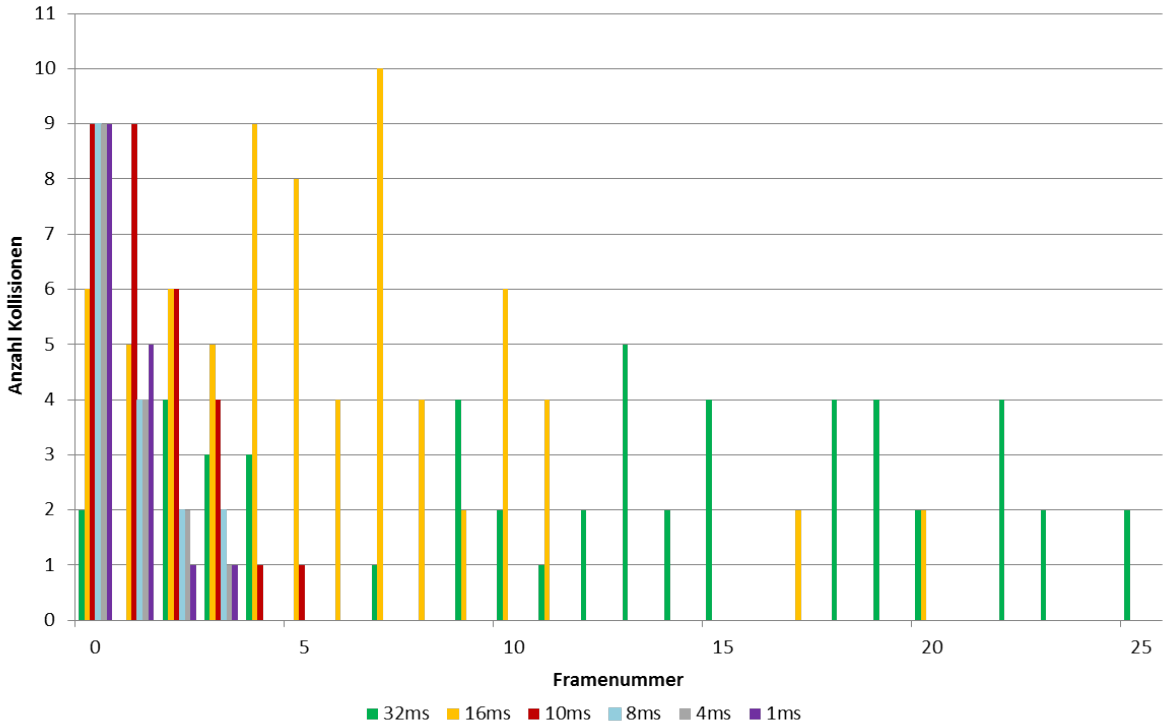

Abb. 66: *Kollisionsverteilung in Abhängigkeit vom Sendezeitabstand eines schmalbandigen Störers*

Abbildung 66 zeigt die Kollisionsverteilung von bereits eingeschwungenen Systemen in Abhängigkeit vom Sendezeitabstand eines schmalbandigen Störers. Auch hier ist zu erkennen, dass der neue stabile Zustand der Nutzdatensysteme umso schneller erreicht wird, je kleiner der Sendezeitabstand des Störers ist. Die Abbildung verdeutlicht aber auch, dass das Medienzugriffsverfahren dem Störer nach Erkennung der Kollisionen aus dem Weg geht. Die Nutzdatensysteme wechseln auf einen Ausweichkanal. Der Kanal des Störers wird nicht mehr verwendet. Im neuen eingeschwungenen Zustand treten keine weiteren Kollisionen auf.

7.4 Zusammenfassung

Dieses Kapitel hat zwei neue Ansätze für Medienzugriffsmechanismen der industriellen Automation untersucht. Beide Verfahren verwenden TDMA als Grundlage. Als wesentlicher Unterschied zu den Zugriffsmechanismen der EN 300 328 wird der Zugriff nicht aus Sicht der jeweiligen Geräte, sondern aus Systemsicht realisiert.
Mit MS-Aloha wurde ein Zugriffsmechanismus analysiert, welcher aus der Car-to-Car-Kommunikation stammt. Diesem liegt ein dezentraler Ansatz zu Grunde. Die Simulationsergebnisse für MS-Aloha haben gezeigt, dass sich das Zeit- und Fehlerverhalten für solche industriellen Automatisierungsanwendungen eignet, die keine niedrigen Latenzen benötigen. Leider arbeitet MS-Aloha nur auf einem Kanal. Dieser wird auch nicht selbstständig gewechselt, wenn beispielsweise ein schmalbandiger Störer denselben Kanal nutzt.

Die Ergebnisse haben aber weiterhin gezeigt, dass MS-Aloha ein weitestgehend robustes Übertragungsverhalten bei einem vorhandenen Störer aufweist. Ein Vorteil von MS-Aloha ist die Vermeidung des Hidden-Node-Problems.
Des Weiteren wurde vom Autor in dieser Arbeit ein eigener Ansatz für ein Medienzugriffsverfahren entwickelt. Dieses verwendet als Ausgangspunkt den Zugriffsmechanismus von DECT. Allerdings ist das neue Medienzugriffsverfahren nicht auf das Frequenzband von DECT beschränkt. Es nutzt mehrere Frequenzkanäle, sodass mehrere Systeme parallel betrieben werden können. Im Falle eines schmalbandigen Störers wird der betroffene Kanal gewechselt. Mit dem nachgewiesenen Zeit- und Fehlerverhalten eignet sich der vorgestellte Ansatz ausgezeichnet für die industrielle Automation. Er ist allerdings stark idealisiert. So wurde der Managementverkehr, welcher innerhalb eines Systems notwendig ist, bisher nicht berücksichtigt. Vor einem realen Einsatz sind noch eine Reihe weiterer Probleme zu lösen. Im Modell kann beispielsweise eine Verbindung zur Übertragung den nächstmöglichen Slot verwenden, ohne eine vorherige Reservierung vornehmen zu müssen. Für eine reale Umsetzung könnte mit einem CCA-Mechanismus, welcher nicht auf dem Energiesondern auf dem Codelevel arbeitet, eine unterschiedliche Priorisierung der Teilnehmer in den verschiedenen Slots umgesetzt und so das Problem gelöst werden. Eine leichte Änderung des Zeit- und Fehlerverhaltens wäre die Folge. Auch könnte das selbstentwickelte Medienzugriffsverfahren weiter verbessert werden. Zum Beispiel könnten in den Beacons Informationen für die Slot-Kanal-Zuordnung eines Systems enthalten sein. Hinzukommende Systeme könnten diese Informationen bei der Generierung ihrer Sprungfolge berücksichtigen. Dadurch wird die Anzahl an Kollisionen weiter reduziert.
Eine gerätetechnische Umsetzung der beiden vorgestellten Ansätze ist bei Verwendung einer Software Defined Radio (SDR)-Platform mit relativ geringem Zeit- und Kostenaufwand möglich.

8 Untersuchung der Koexistenz zwischen WLAN und WSAN-FA

8.1 Allgemein

Im Jahr 2012 wurde WSAN-FA (Wireless Sensor Actor Network - Factory Automation) neben den für die drahtlose Übertragung von PROFINET-Telegrammen zugelassenen WLAN und Bluetooth von der PROFIBUS-Nutzerorganisation (PNO) als Funkstandard für die Fertigungsindustrie spezifiziert. Seine Abstammung findet diese Spezifikation für die Fabrikautomation in WISA (Wireless Interface for Sensors and Actuators), eine Funktechnologie, die von der Firma ABB entwickelt und in einen produktreifen Zustand überführt wurde. WSAN-FA dient vor allem der drahtlosen Verbindung von Sensoren und Aktoren mit einer Maschinensteuerung. Da in der Fertigungsindustrie die Sensorsignale überwiegend binär sind, existiert bei WSAN-FA eine Begrenzung der Nutzdaten von 2 Byte. Dies ermöglicht WSAN-FA allerdings eine echtzeitfähige Übertragung, wie sie beispielsweise in einer Roboterzelle benötigt wird. WSAN-FA basiert grundlegend auf dem IEEE 802.15.1 Standard und ist damit sehr schmalbandig. Es nutzt wie Bluetooth Frequenzhopping, wodurch eine gute Robustheit erzielt wird. WSAN-FA ist ein Vertreter der TDMA-Systeme.

Anders als WSAN-FA besitzt WLAN Nachteile im Zeitverhalten, da es CSMA/CA als Medienzugriffsverfahren verwendet. Bei CSMA/CA wird vor der Übertragung eines Datenpaketes eine zufällige Wartezeit generiert. Auch kann die Überprüfung auf ein freies Medium mit CCA, welche während dieser Wartezeit durchgeführt wird, leicht durch andere Funksysteme und -teilnehmer gestört werden. Eine fehlerfreie Übertragung innerhalb einer vorhersagbaren maximalen Zeitdauer kann mit WLAN aus diesen Gründen nicht gewährleistet werden. WLAN ist damit ohne zusätzliche Modifikationen durch die Hersteller für eine echtzeitfähige Kommunikation, wie sie von der industriellen Automation gefordert wird, wenig geeignet.

In diesem Abschnitt sollen WSAN-FA und WLAN als Beispiele für reale Funksysteme, die bereits in der industriellen Automation zur drahtlosen Kommunikation eingesetzt werden (WSAN-FA in Form von WISA), modelliert und in ausgewählten Koexistenzszenarien simuliert werden. Dabei werden TDMA und CSMA/CA als die vorherrschenden Medienzugriffsverfahren in einem direkten Vergleich bewertet. Bei WSAN-FA wird sich auf das FH02-Frequenzsprungverfahren beschränkt.

Die Inhalte dieses Kapitels stammen aus [50].

8.2 Medienzugriffsverhalten

8.2.1 WSAN-FA

Ein WSAN-FA System besteht aus einer Basisstation (BS) und mindestens einem Sensor/Aktor (SA). Die BS übernimmt die Rolle des Masters und das SA-Gerät die des Slaves. Eine Gruppe, die sich aus einer BS und den dazugehörigen SA-Geräten zusammensetzt, bildet eine sogenannte Zelle. Eine Überlappung mehrerer WSAN-FA-Zellen und damit die gemeinsame Nutzung des gleichen Frequenzbandes ist ebenfalls möglich. Abbildung 67 veranschaulicht den Aufbau und die räumliche Überlappung grafisch.

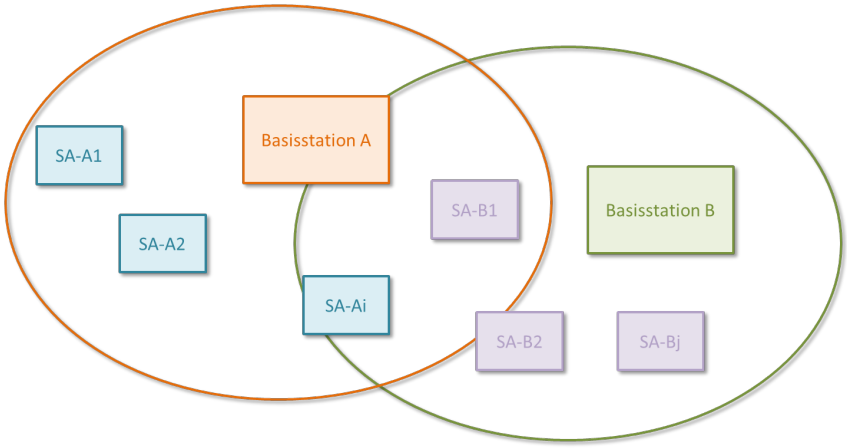

Abb. 67: *WSAN-FA-Zelle*

Die Übertragungsrichtung von der BS zu den SA-Geräten wird als Downlink und die entgegengesetzte Richtung als Uplink bezeichnet (siehe Abbildung 68).

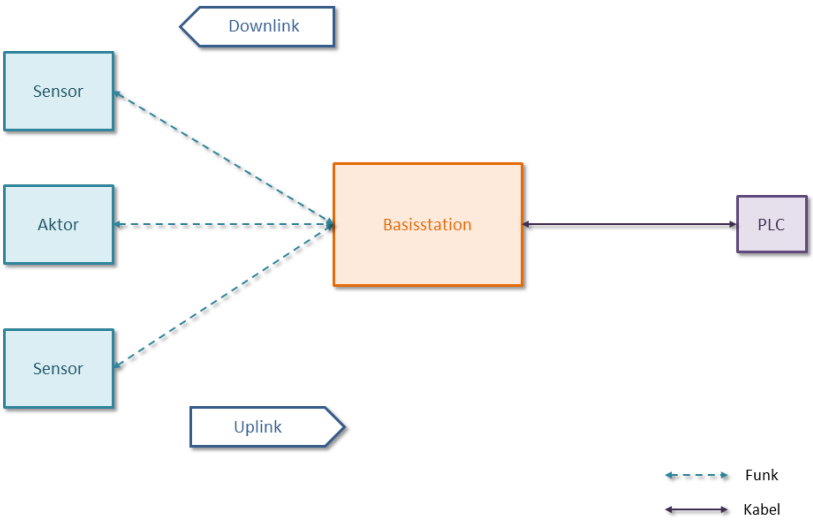

Abb. 68: *Up- und Downlink*

Innerhalb einer Zelle erhält jedes SA-Gerät eine einmalige Adresse. Auf deren Grundlage

erfolgt die Zuweisung zu Sendeslots. WSAN-FA nutzt als Medienzugriffsverfahren TDMA. Dabei übertragen die Teilnehmer die Pakete in ihren fest zugewiesenen Zeitschlitzen. Die Datenübertragung kann sowohl synchron als auch asynchron sein. Ein TDMA-Frame ist 2304 µs lang und wird zyklisch wiederholt. Es besteht aus 16 Slots mit doppelter Länge (Doubleslots/Dslots) beziehungsweise 32 Slots mit einfacher Länge (Singleslots/Sslots). Die Slots 0-14 bzw. 0-29 dienen der Datenübertragung. Im Dslot 15 findet keine Übertragung statt. Diese Sendepause dient stattdessen der Frequenzänderung des Downlink-Frames. Für den Uplink existieren bis zu vier parallele Kanäle. Jeder Uplink-Kanal besitzt eine andere Frequenz. Jedes SA-Gerät ist einem dieser Kanäle zugeordnet. Pro Zelle gibt es nur einen Downlink-Kanal. Jede Zelle nutzt bis zu vier Uplink-Gruppen, wobei jeder Uplink-Gruppe ein anderer Uplink-Kanal zugeordnet ist. Der korrekte Sendeslot eines SA-Gerätes ist abhängig von der Uplink-Gruppe, welcher dieses Gerät zugehörig ist. Abbildung 69 zeigt die Zuweisung von Adressen zu Slots und Uplink-Gruppen. Diese Abbildung gilt nur, wenn vier Uplink-Gruppen existieren. Auch die Zuweisung von Adressen zu Slots des Downlinks wird dargestellt. Auf eine formale Herleitung bzw. Beschreibung soll an dieser Stelle verzichtet werden.

Sslot	Uplink group number				Downlink sub-Dslot address = SA_index								Dslot
	UL0	UL1	UL2	UL3	0	1	2	3	4	5	6	7	
	Corresponding SA numbers				Corresponding SA numbers								
0	0	1	2	3	48	49	50	51	108	109	110	111	0
1	60	61	62	63									
2	4	5	6	7	52	53	54	55	112	113	114	115	1
3	64	65	66	67									
4	8	9	10	11	56	57	58	59	116	117	118	119	2
5	68	69	70	71									
6	12	13	14	15	0	1	2	3	60	61	62	63	3
7	72	73	74	75									
8	16	17	18	19	4	5	6	7	64	65	66	67	4
9	76	77	78	79									
10	20	21	22	23	8	9	10	11	68	69	70	71	5
11	80	81	82	83									
12	24	25	26	27	12	13	14	15	72	73	74	75	6
13	84	85	86	87									
14	28	29	30	31	16	17	18	19	76	77	78	79	7
15	88	89	90	91									
16	32	33	34	35	20	21	22	23	80	81	82	83	8
17	92	93	94	95									
18	36	37	38	39	24	25	26	27	84	85	86	87	9
19	96	97	98	99									
20	40	41	42	43	28	29	30	31	88	89	90	91	10
21	100	101	102	103									
22	44	45	46	47	32	33	34	35	92	93	94	95	11
23	104	105	106	107									
24	48	49	50	51	36	37	38	39	96	97	98	99	12
25	108	109	110	111									
26	52	53	54	55	40	41	42	43	100	101	102	103	13
27	112	113	114	115									
28	56	57	58	59	44	45	46	47	104	105	106	107	14
29	116	117	118	119									
30	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	15
31	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	

Abb. 69: *Slotzuweisung bei vier Uplink-Gruppen*

Der Physical Layer entspricht in seiner Charakteristik dem von Bluetooth (IEEE 802.15.1). Die Bitrate beträgt 1 MBit/s. Es stehen 83 Kanäle im 2,4 GHz-ISM-Band mit jeweils 1 MHz Bandbreite zur Verfügung. Als Modulationsverfahren wird GFSK (Gaussian Frequency Shift Keying) verwendet.

Die Übertragung im Downlink unterteilt sich in zwei Modi: Permanent- und Quiet-Downlink. Bei den weiteren Untersuchungen wird sich auf den Permanent-Downlink beschränkt. Beim Permanent-Downlink werden kontinuierlich Telegramme gesendet. Wenn keine Daten zum Senden anstehen, werden stattdessen Dummy-Telegramme übertragen.

Im Gegensatz zum Downlink werden beim Uplink nur Pakete übertragen, wenn auch tatsächlich Daten vorhanden sind. Zusätzlich werden in bestimmten Zeitabständen IMA-Telegramme („I'm Alive“) gesendet, die der BS mitteilen, dass das entsprechende SA-Gerät noch aktiv ist und kein Fehlerzustand vorliegt. IMA-Telegramme bestehen in der Regel aus Diagnosedaten. Daten zur Diagnose sind beispielsweise der Zähler für wiederholte Übertragungen oder Kommandos, die vom Hersteller festgelegt werden.

Um eine gewisse Robustheit zu gewährleisten, nutzt WSAN-FA neben einem schmalbandigen Signal auch zwei verschiedene Frequenzsprungverfahren: FH02 und FH07. Das WSAN-Frequenzband, welches in Abbildung 70 dargestellt wird, ist in 7 überschneidungsfreie Unterbänder unterteilt. Jedes Unterband enthält 11 WSAN-FA-Kanäle.

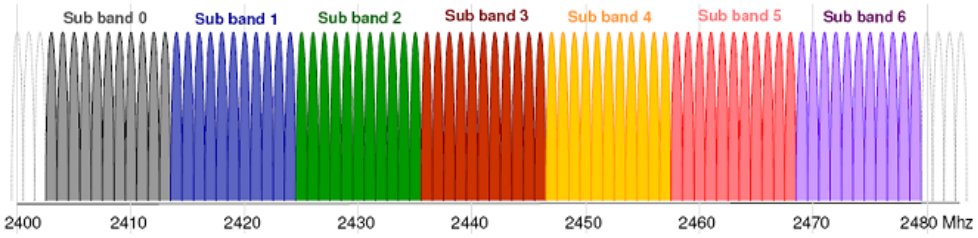

Abb. 70: *WSAN-Frequenzen und Unterbänder*

Im Folgenden wird sich auf das FH02-Frequenzsprungverfahren beschränkt. Das FH02-Frequenzsprungverfahren ist auf den Bereich zwischen 2403 MHz und 2479 MHz beschränkt. Die Sprungfolge ist abhängig von:

- Frame-Nummer,
- Cell-ID,
- Übertragungsrichtung (Downlink/Uplink),
- Uplink-Gruppe.

Für den Downlink aber auch für die 4 Uplinks gibt es 59 individuelle Sprungsequenzen. Der Abstand zwischen der aktuellen Frequenz des Downlinks und einem Satz Uplink-Frequenzen beträgt 22 MHz (Duplex Spacing). Die einzelnen Uplink-Frequenzen unterscheiden sich um 2 MHz. Die Frame-Nummer wird mit jedem Frame von 0 bis 76 inkrementiert.

Die Berechnung der Frequenz eines TDMA-Frames erfolgt in Abhängigkeit von zwei Tabellen, welche die äußere Sprungsequenz W und die innere Sprungsequenz X darstellen. Die äußere Sprungsequenz bestimmt das Unterband und die innere Sprungsequenz den Kanal innerhalb dieses Unterbandes. Abbildung 71 zeigt beispielhaft die Sprungsequenz eines Downlinks. Für die Herleitung und Berechnung wird auf die WSAN-FA Spezifikationen ([100], [101]) verwiesen.

	Zellen-Identifikationsnr.										
Framenr.	1	2	3	4	5	6	7	8	9	10	27
0	2403	2403	2403	2403	2403	2403	2403	2403	2403	2403	
1	2416	2417	2418	2419	2420	2421	2422	2423	2424	2426	
2	2429	2431	2433	2435	2426	2428	2430	2432	2434	2449	
3	2442	2445	2437	2440	2443	2446	2438	2441	2444	2472	
4	2455	2448	2452	2456	2449	2453	2457	2450	2454	2418	
5	2468	2462	2467	2461	2466	2460	2465	2459	2464	2441	
6	2470	2476	2471	2477	2472	2478	2473	2479	2474	2464	
7	2406	2413	2409	2405	2412	2408	2404	2411	2407	2410	
8	2419	2416	2424	2421	2418	2415	2423	2420	2417	2433	
9	2432	2430	2428	2426	2435	2433	2431	2429	2427	2456	
31											2431

Abb. 71: *Beispielhafte Sprungsequenz eines Downlinks*

8.2.2 WLAN

Als WLAN (Wireless Local Area Network) wird eine Funktechnologie bezeichnet, die der IEEE 802.11 Spezifikation entspricht. In der Grundstruktur ist der Access Point (AP) über Ethernet an ein drahtgebundenes Netzwerk angeschlossen. Der AP baut zum Datenaustausch über Funk ein sternförmiges Netzwerk mit anderen WLAN-Geräten, den sogenannten Clients, auf. Diese können z. B. Notebooks oder Drucker sein. Ein AP stellt nun in Form eines Gateways die Datenkommunikation zwischen drahtgebundenen und drahtlosen Teilnehmern her. Ebenfalls dient er als Vermittlungsstelle zwischen den drahtlosen Geräten untereinander. Zum besseren Verständnis ist die gängige Integration von WLAN in Abbildung 72 dargestellt.

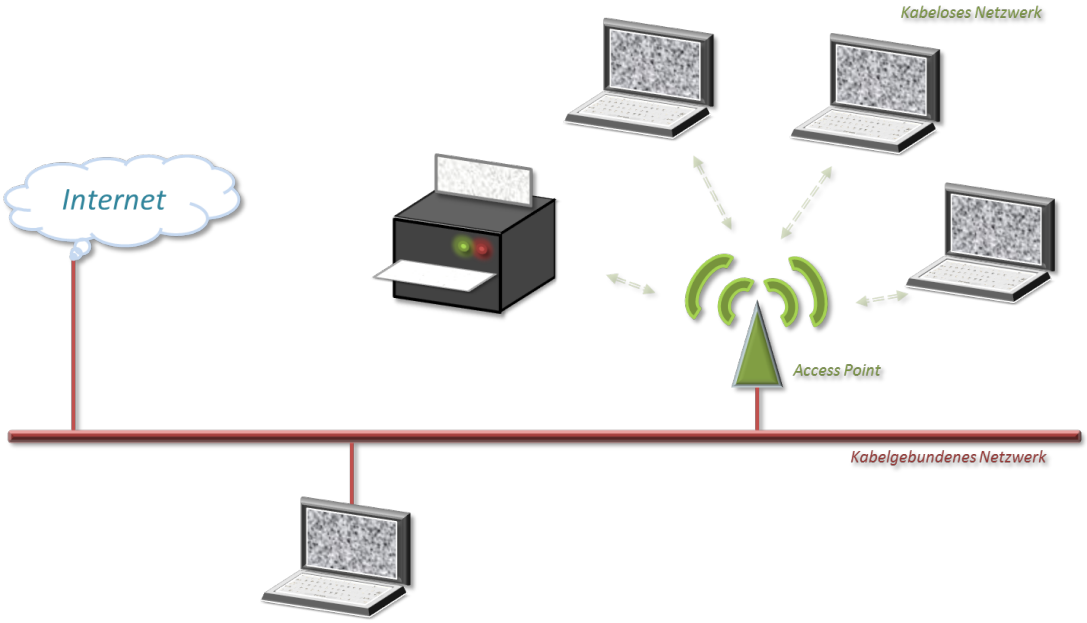

Abb. 72: *Integration von WLAN*

Die IEEE 802.11 Spezifikation umfasst mehrere Ausführungen. So wird zwischen a/h, b/g und n unterschieden. Die Spezifikation IEEE 802.11 a/h sendet im 5 GHz-ISM-Band, das 2,4 GHz-Band wird von IEEE 802.11 b/g genutzt und IEEE 802.11 n unterstützt sowohl das eine als auch das andere der genannten Frequenzbänder. Die Kanalbandbreite beträgt 20 MHz. Eine Ausnahme bildet IEEE 802.11 n, das neben 20 MHz auch 40 MHz unterstützt.

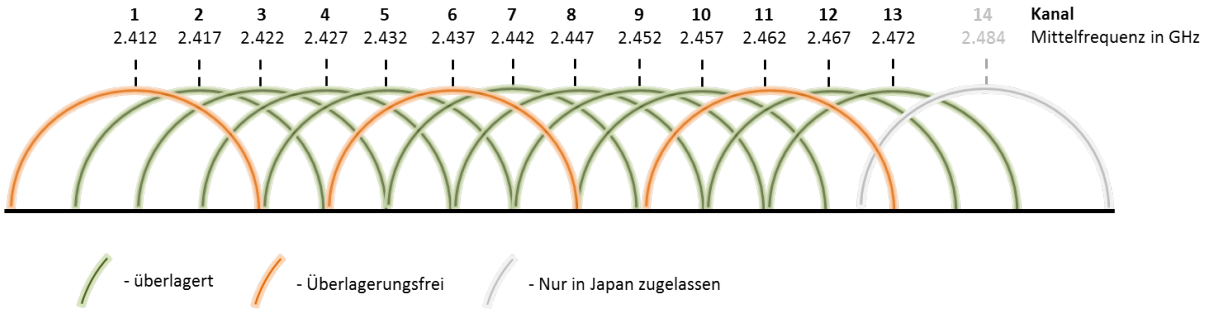

Abb. 73: *WLAN-Kanäle im 2,4 GHz-ISM-Band*

Abbildung 73 zeigt alle WLAN-Kanäle im 2,4 GHz-ISM-Band. Nur die ersten 13 sind in Europa zulässig. Zusätzlich ist in Japan Kanal 14 freigegeben. Von den 13 in Europa zugelassenen Kanälen können jeweils nur drei Kanäle gleichzeitig überlagerungsfrei genutzt werden. Bei WLAN kommt als Modulationsverfahren OFDM (Orthogonal Frequency Division Multiplexing) zum Einsatz. OFDM bedient sich mehrerer orthogonaler Zwischenträger (siehe Abbildung 74), die mittels BPSK, QPSK, 16 QAM oder 64 QAM moduliert sind.

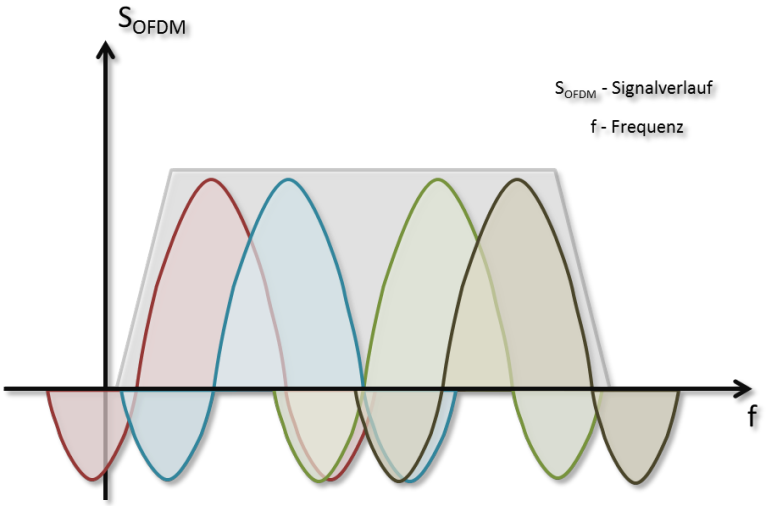

Abb. 74: *Prinzip von OFDM*

Wird ein Träger schmalbandig gestört, findet dennoch die Übertragung auf den anderen Trägern statt. Die Gesamtdatenübertragungsrate nimmt leicht ab.

Das von WLAN verwendete Zugriffsverfahren CSMA/CA (siehe Abschnitt 2.4.3) schreibt jeder sendewilligen Station vor, dass vor einer Übertragung das Funkmedium erst auf Freiheit hin überprüft werden muss. Die Zeit, wie lange das Medium frei sein muss, ist dabei genau definiert. Wird eine Belegung festgestellt, wird die Sendung der Daten zurückgehalten und so lange gewartet, bis das Medium wieder als frei erkannt wird.

Das Carrier Sensing von CSMA/CA basiert auf CCA, welches das Energielevel im interessierenden Frequenzbereich auf Erhöhungen untersucht. Andere Stationen in der Umgebung heben das Energielevel durch das Senden von Daten an. Bei Überschreitung einer festgelegten Schwelle, welche für WLAN bei ca. -76 dB liegt, erkennt das CCA das Medium als belegt und es dürfen keine Daten gesendet werden. Das CCA erfolgt im Physical Layer (PHY), das Ergebnis wird in der MAC-Schicht ausgewertet und verarbeitet.

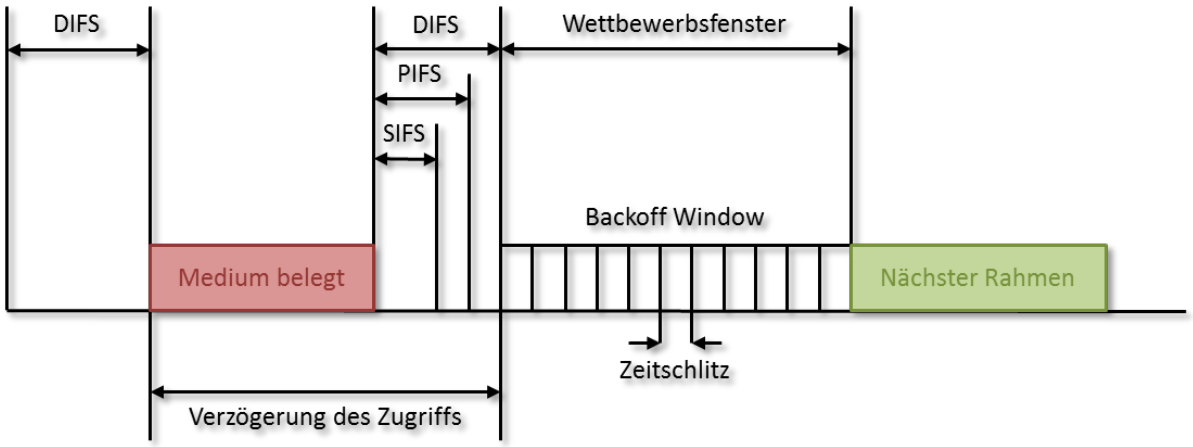

Abb. 75: *Zeitliche Abfolge des WLAN-Medienzugriffs*

Nachdem das Medium nach einer Übertragung wieder als frei erkannt wurde, muss dieses für eine gewisse Zeitspanne ungenutzt bleiben, ehe es wieder belegt werden darf. Die Wartezeiten beim Medienzugriff werden durch unterschiedliche Zeitintervalle definiert, deren Gesamtheit als Inter Frame Spaces (IFS) bezeichnet wird. Abbildung 75 zeigt die unterschiedlichen Wartezeiten und den generellen Ablauf des Medienzugriffs nach dem CSMA/CA-Verfahren.
Folgende Zeitintervalle werden in Abbildung 75 dargestellt:

- SIFS: Short Inter Frame Space ist der kürzeste Abstand zwischen aufeinanderfolgenden Paketen während des Datenaustauschs zweier Stationen. Er tritt überwiegend vor der Übertragung von ACK-Paketen auf.
- PIFS: Point Coordination Inter Frame Space ist die Zeit, in der der Access Point das Vorrecht besitzt, auf das Medium zuzugreifen. Die Zeitdauer beträgt eine SIFS-Periode zuzüglich einer Slotlänge.
- DIFS: Distributed Inter Frame Space ist die Zeit, die vor jeder neuen Datenübertragung abgewartet werden muss. Die Dauer beträgt die einer PIFS-Periode zuzüglich einer Slotlänge. Um Kollisionen zeitgleicher Anfragen zu vermeiden, wird zusätzlich der sogenannte Backoff-Algorithmus durchgeführt. Dabei werden durch zufällig gewählte Wartezeiten Kollisionen unterschiedlicher Sendeanfragen verhindert.

Handelt es sich bei der Datenübertragung um den ersten Sendeversuch, muss nur ein DIFS-Intervall abgewartet werden, andernfalls addiert sich eine weitere zufällige Wartezeit gemäß des Backoff-Algorithmus. Um den korrekten Empfang eines Datenpaketes zu bestätigen, wird ein ACK gesendet. Bleibt der Empfang des ACKs aus, entweder weil der Paketempfang fehlerhaft war oder das ACK selbst verloren ging, wird das entsprechende Paket erneut gesendet. Identisch zu WSAN-FA gibt es auch bei WLAN eine maximale Anzahl an Wiederholungen.

8.3 Modell

8.3.1 Allgemein

Die erstellten Modelle entsprechen dem System- und zu großen Teilen auch dem Verbindungsmodell. Beide wurden bereits bei der Modellierung von Zugriffsverfahren der EN 300 328 erläutert. Auch Application und Physical Layer sind gleich geblieben. Große Veränderungen waren im Data Link Layer (DLL) aufgrund des veränderten Medienzugriffs notwendig. Im Folgenden wird aus diesem Grund nur auf diese notwendigen Anpassungen eingegangen.

8.3.2 WSAN-FA

Das WSAN-FA-Modell besteht aus zwei Teilen, der BS und den SA-Geräten.

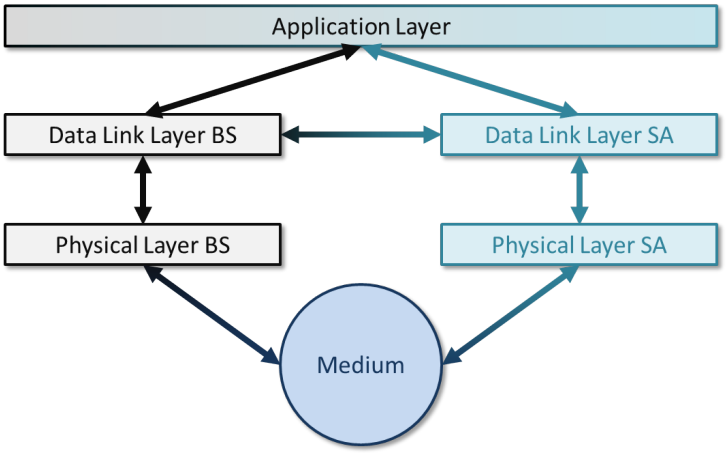

Abb. 76: *Schematischer Aufbau des WSAN-FA-Modells*

Das Modell ist aus mehreren Schichten aufgebaut. Abbildung 76 zeigt eine schematische Darstellung. In dieser Abbildung ist zu erkennen, dass nur ein Application Layer für BS und SA zusammen existiert. Bei einem realen WSAN-FA-System muss der Empfänger eines Datenpaketes die erfolgreiche Übertragung mit einem Acknowledgement (ACK) bestätigen. Da sowohl das Datenpaket selbst als auch das dazugehörige ACK per Funk übertragen werden, können auch beide gestört werden. Für eine einfachere Modellierung wird ein Paket (Dummy, Daten, ACK) von demselben Bestandteil des Modells auf das Medium gelegt, welches es anschließend auch wieder entnimmt. Die Verbindung zwischen den DLLs der BS und der SA dient der Synchronisation. Der DLL des BS übermittelt die derzeit aktuelle Slotnummer an die SAs, damit Uplink-Pakete im richtigen Slot gesendet werden können. Die Verbindung wird aber auch zur Generierung von ACKs benötigt. Nach erfolgreicher Übertragung eines Daten-Paketes legt beispielsweise die BS ein ACK direkt in den Sendebuffer der SAs, welcher dieses schnellstmöglich auf das Medium legt. Der Status dieser Übertragung wird von der SA zur BS über die direkte Verbindung zurückgesendet.

Abbildung 77 zeigt die Umsetzung des Data Link Layers der Basisstation. Dabei weist der DLL zunächst jedem vom APPL kommenden Paket einen passenden Sendeslot und Index zu. Der SA-Index ist für die gleichzeitige Kommunikation der BS mit SAs innerhalb eines Slots notwendig.

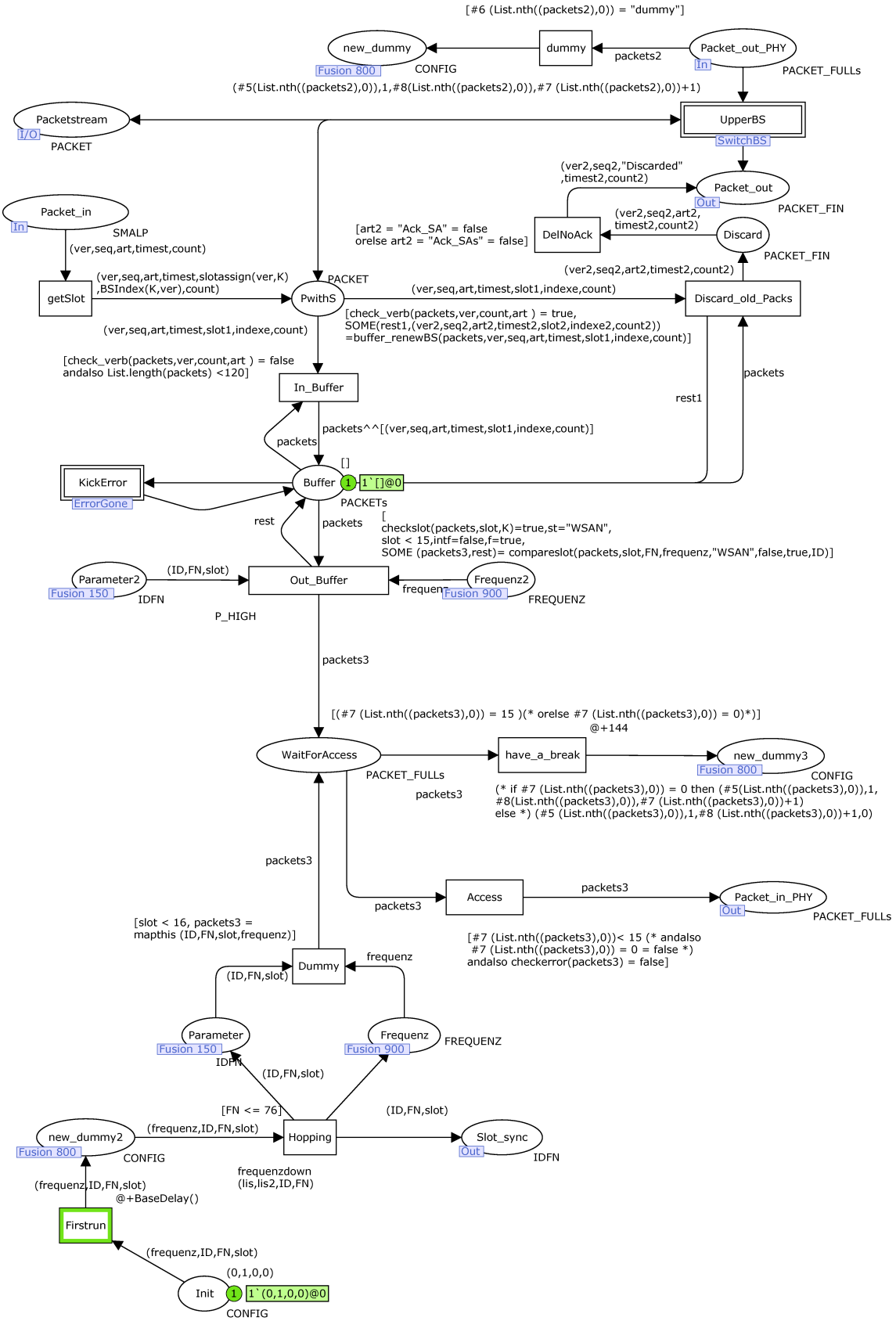

Abb. 77: *Data Link Layer der Basisstation*

Anschließend fließt das um die Slotnummer und den SA-Index erweiterte Paket in den Buffer. Da jeder angebundene Sensor nur seinen aktuellen Messwert vorhält, ist es nicht gestattet, dass sich im Buffer mehrere Datenpakete mit der gleichen Verbindungsnummer be-

finden. Aus diesem Grund besitzen die Transitionen „In_Buffer“ und „Discard_old_Packs“ entsprechende Funktionen, die eine solche Mehrfachbelegung verhindern. Wenn ein aktuelleres Datenpaket den Buffer belegen möchte, dort aber bereits eines mit der identischen Verbindungsnummer liegt, wird das Datenpaket mit dem älteren Messwert verworfen und durch das neue ersetzt. Dagegen dürfen ACKs den Buffer immer ungeprüft betreten. Ein Verwerfen von ACKs ist nicht möglich.

Beim Auftreten eines Fehlers, ist für einen gewissen Zeitraum das Senden von Datenpaketen an das entsprechende SA-Gerät nicht erlaubt. Stattdessen wird eine Error-Marke in den Buffer gelegt. Diese lässt für einen definierten Zeitraum keine Datenpakete mit der Verbindungsnummer, die als Eigenschaft in der Error-Marke enthaltenen ist, passieren. Allerdings dürfen ACKs den Buffer betreten, trotz einer vorliegenden Error-Marke für diese Verbindung. Die Zeitdauer für das Verweilen der Error-Marke im Buffer wird mit Hilfe der „KickError“-Transition realisiert. Beim Feuern dieser Transition wird automatisch die Error-Marke aus dem Buffer entfernt.

In der WSAN-FA-Basisstation werden kontinuierlich Dummy-Pakete generiert, sodass einem im Buffer befindlichen Paket der richtige Sendeslot zugeordnet werden kann. Sollte kein Paket mit aktueller Slotnummer im Buffer liegen, so wird dieses Dummy-Paket zu Synchronisationzwecken auf das Medium gelegt. Als wichtige Attribute enthält das Dummy-Paket die Slotnummer (slot), die Framenummer (FN) und die Identifikationsnummer (ID). Zusätzlich weist ein Dummy-Paket als Eigenschaft die Sendefrequenz auf, welche für das Modell wichtig ist. Diese wird über die Funktion „frequenzdown“ bestimmt. Es kommt zur Bestimmung der Hoppingsequenz das FH02-Frequenzsprungverfahren zum Einsatz. Die Slotnummer wird von 0 bis 15 inkrementiert. Dabei werden die Slotnummern von 0 bis 14 zur Datenübertragung verwendet. Der 15. Slot dient als Gap zur Änderung der Sendefrequenz. Nach Ablauf der Pause wird die Slotnummer auf 0 zurückgesetzt und die Framenummer um eins erhöht. Dieser Vorgang wird fortgesetzt, bis der Wert 76 erreicht wird. Der beschriebene Zyklus beginnt nach 76 Frames von vorn.

Stimmt die Slotnummer eines Paketes mit der des aktuellen Dummy-Paketes überein, wird dieses aus dem Buffer entnommen. Die Überprüfung der Slotübereinstimmung wird mit der Funktion „checkslot“ realisiert. Dabei wird die Slotnummer des Paketes im Buffer mit der des Dummys verglichen. Für eine Verbesserung der Übersichtlichkeit wurden dazu Fusionsplätze verwendet. Fusionsplätze sind eine Menge von Plätzen, die über die gleichen Marken verfügen. Wird eine Marke auf einen dieser gleich bezeichneten Plätze abgezogen, so wird diese auch bei allen anderen Plätzen abgezogen. Mit der Funktion „compareslot“ werden die Pakete aus dem Buffer zur Übertragung über das Medium angepasst. Die Pakete werden anschließend an den Physical Layer übergeben.

Im oberen Bereich von Abbildung 77 ist ein Platz mit der Bezeichnung „Packet_out_PHY“ angeordnet, welcher als Schnittstelle zwischen PHY und DLL fungiert. Über den genannten Platz fließen Pakete zum DLL zurück, die im PHY auf das Medium gelegt und in Bezug auf Interferenz überprüft wurden. Wenn über diese Schnittstelle ein Paket zurück zum DLL gelangt, wird umgehend ein neues Dummy-Telegramm mit darauffolgender Slotnummer generiert.

Hinter der Subtransition „UpperBS“ verbirgt sich ein Petri-Netz, welches die Weiterverarbeitung und Weitergabe von ankommenden Paketen bewerkstelligt. Dazu wird die ankommende Paketliste in ihre Bestandteile aufgelöst. Anschließend wird das Interferenz-Attribut der jeweils einzelnen Pakete überprüft. Falls der Wert des Attributs „true“ ist, wird das Paket wieder zurück in den Buffer geleitet, um erneut gesendet zu werden. Falls das Interferenz-Attribut den Wert „false“ besitzt, wird das zugehörige Paket dem APPL übergeben. Zusätzlich wird ein ACK erzeugt, welches unmittelbar über den Platz „Packetstream“ an das Netz der SA-Geräte übertragen wird.

Neben einer BS gehört zu einem vollständigen WSAN-FA-System mindestens ein SA-Gerät. Die SA-Seite ist in der Lage, mehrere SA-Geräte, die zu derselben WSAN-FA-Zelle gehören, mit nur einem Petri-Netz-Modell nachzubilden. Der Schichtenaufbau gleicht im Wesentlichen dem der BS. Allerdings wurde die Funktionsweise des APPL zu großen Teilen auf die BS ausgelagert.

Das Petri-Netz-Modell des DLLs der SA-Seite ist bis auf einen wesentlichen Unterschied identisch zu dem der BS. Dummy-Pakete, die von der BS generiert und auf das Funkmedium gelegt werden, werden von den SA-Geräten lediglich zur Slotsynchronisation genutzt. Anstatt Dummy-Paketen erzeugt jedes SA-Gerät IMA-Pakete. Diese IMA-Pakete werden nur an die BS gesendet, wenn kein Datenpaket oder ACK zur Übertragung ansteht. Ihre Priorität ist geringer. Die Zeit zur Generierung der IMA-Pakete ist abhängig von dem zeitlichen Abstand zweier Datenpakete. Bei einem zu großen zeitlichen Abstand zwischen aufeinanderfolgenden Datenpaketen werden diese Lücken mit IMA-Paketen aufgefüllt. Identisch zu Datenpaketen und ACKs gibt es IMA-Pakete sowohl für Single- und Doubleslots. Sie werden außerdem in denselben festgelegten Slots wie die anderen Pakettypen gesendet.

8.3.3 WLAN

Für die Koexistenzbetrachtung eines TDMA- und eines CSMA/CA-Systems ist nach der Modellierung des WSAN-FA-Systems auch die Integration des WLAN-Modells notwendig. Das Modell, welches ein WLAN-Funksystem nachbildet, soll in diesem Abschnitt kurz erläutert werden.

Im DLL, welcher das Verbindungsstück zwischen APPL und PHY bildet, wird die Entscheidung getroffen, wann ein zur Übertragung anstehendes Datenpaket gesendet werden darf. Dabei werden die für den Medienzugriff notwendigen festen und zufälligen Zeitintervalle erst abgewartet, wenn das Medium auf der vorgesehenen Sendefrequenz als frei erkannt wurde. Die Überprüfung, ob das Medium frei oder belegt ist, erfolgt durch den CCA-Mechanismus des WLAN-Systems. Dieser ist im PHY angesiedelt. Als Ergebnis liefert das CCA entweder „Medium belegt“ oder „Medium frei“. Der aktuelle Zustand sowie deren Änderung wird kontinuierlich an den DLL übertragen, dort ausgewertet und für den Medienzugriffsmechanismus zugänglich gemacht. Die beiden Teile des DLLs sind in Abbildung 78 und 79 dargestellt.

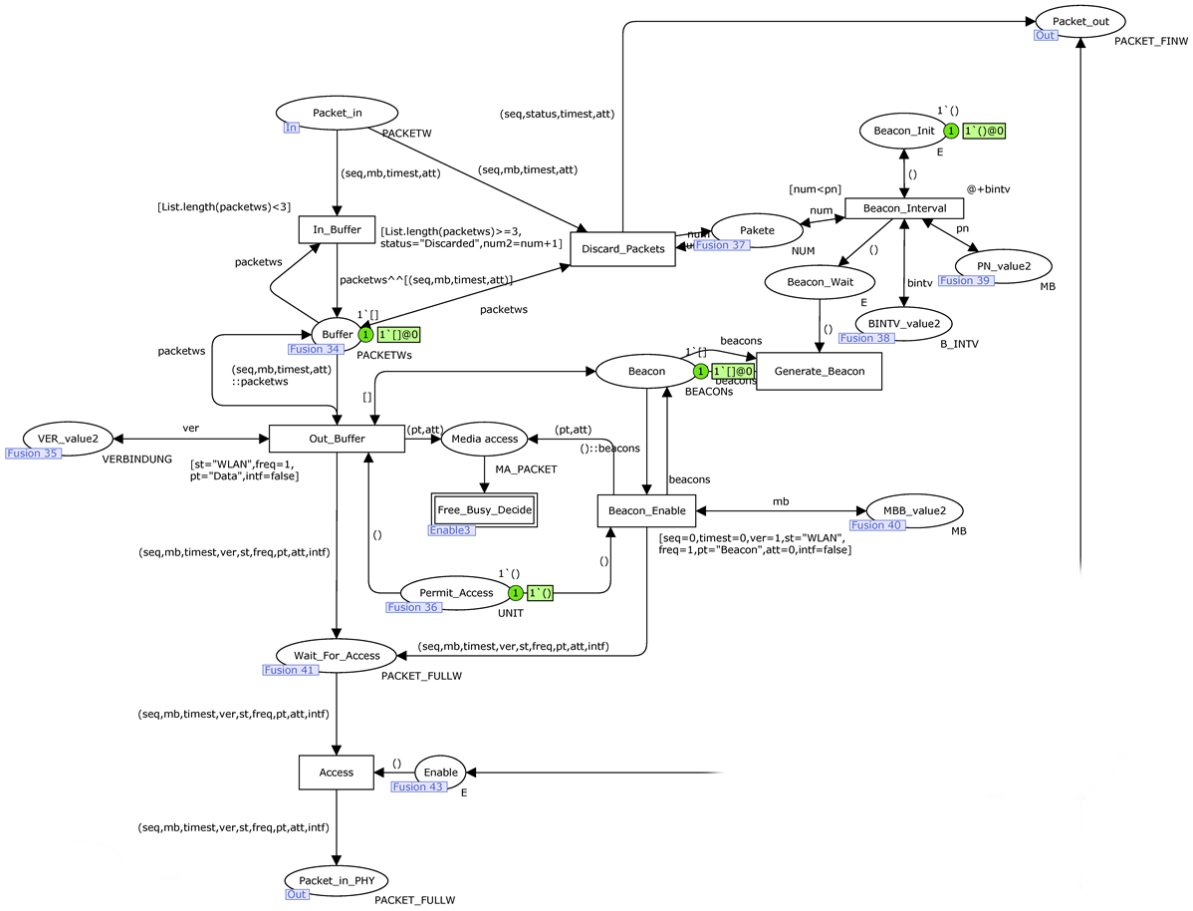

Abb. 78: *Oberer Teil des DLL-Modells für WLAN*

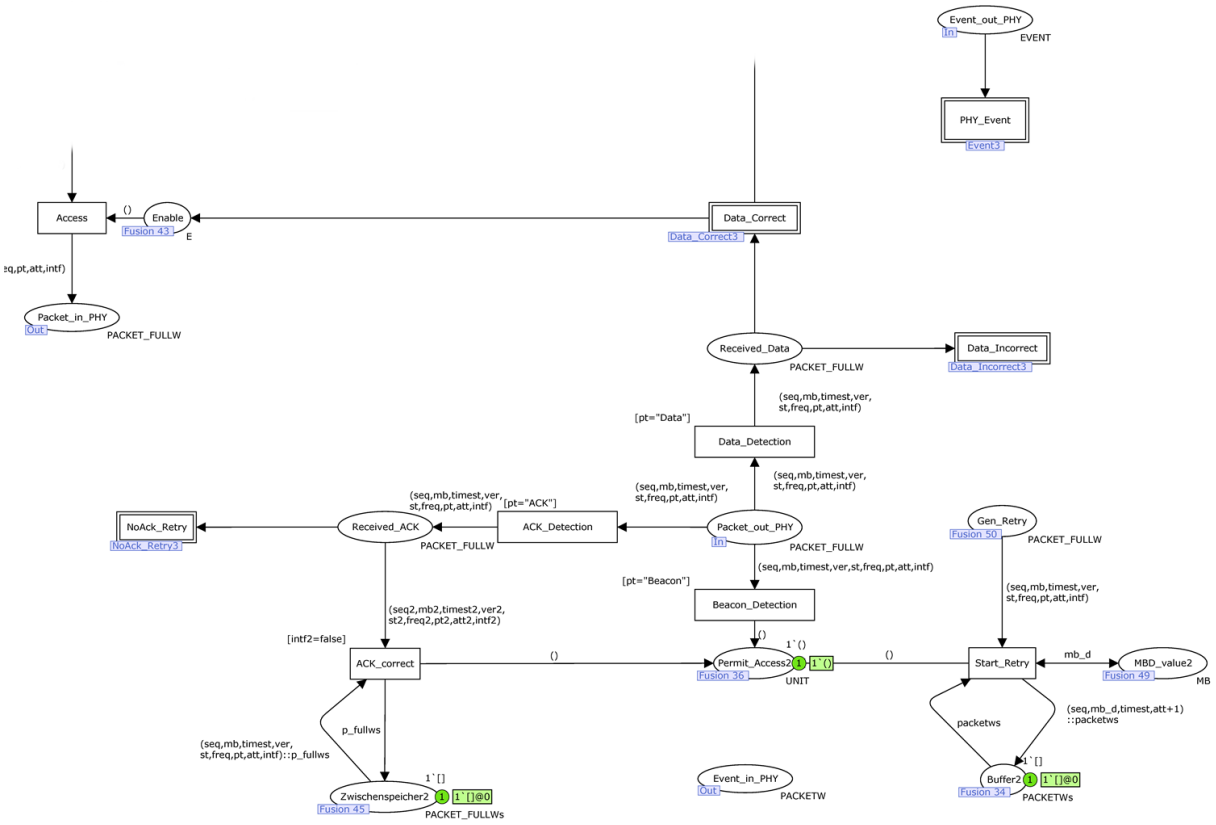

Abb. 79: *Unterer Teil des DLL-Modells für WLAN*

Pakete, die im APPL generiert wurden, werden in Form von Marken über den Platz „Packet_in“ dem DLL zugeführt. Anschließend gelangen die Marken in den „Buffer“, welcher in Form einer Liste die dort abgelegten Pakete zusammenfasst. Die Listenlänge und damit der Speicherumfang des Buffers ist auf drei Pakete begrenzt. Sollte der Buffer komplett belegt sein und ein weiteres vom APPL kommendes Paket versuchen, in den Buffer zu gelangen, wird dieses als verworfen gekennzeichnet und zur Analyse zurück an den APPL gesendet. Der Buffer entspricht einem FIFO. Ein weiterer Bestandteil des DLL ist die Generierung der Beacons, welche nur für APs relevant ist. Die Beacon-Generierung ist im oberen Teil des Layers (siehe Abbildung 78) dargestellt. Beacon-Pakete werden in festgelegten Zeitabständen von typischerweise 100 ms von einem AP ausgesendet, um das Netzwerk mit dem dazugehörigen Netzwerknamen (SSID) in der näheren Umgebung bekannt zu machen. Sie unterstützen dadurch beispielsweise einen schnelleren Verbindungsaufbau. Ein Beacon-Paket muss im Gegensatz zu Datenpaketen nicht den Buffer durchlaufen und dessen Übertragung bekommt damit eine höhere Priorität zugewiesen.

Die Subtransition „Free_Busy_Decide“ benötigt zur Abarbeitung Informationen über die Belegung des Mediums. Die notwendigen Informationen werden von der Subtransition „PHY_Event“ geliefert, welche in Abbildung 79 dargestellt ist. Diese Subtransition überprüft die auf dem Medium befindlichen Pakete auf deren Belegung im Frequenzband und gibt die Information, ob der für die Übertragung interessierende Frequenzbereich frei ist oder nicht an den DLL weiter.

Die von der Subtransition „Free_Busy_Decide“ stammende Information, ob das Medium belegt oder frei ist, wird für den Medienzugriff (siehe Abbildung 80) benötigt. Bei der Durchführung des Medienzugriffs wird grundsätzlich unterschieden, ob entweder ein Beacon- oder ein Datenpaket zur Übertragung ansteht. Bei einem Beacon muss der relevante Frequenzbereich des Mediums für die Zeitdauer einer PIFS unbelegt sein. Der dafür vorgesehene Modellbestandteil ist für die Einhaltung dieses Zeitintervalls verantwortlich. Ist das Medium für die PIFS-Zeit nicht unbelegt, darf die Übertragung nicht freigegeben werden. Bei einem Datenpaket ist die Vorgehensweise ähnlich. Dabei muss das Medium für die DIFS-Periode und einer zufälligen Wartezeit (Wettbewerbsfenster) unbelegt sein. Auch dieses Zeitverhalten muss durch das Petri-Netz für den Medienzugriff sichergestellt werden.

Wird das Medium innerhalb der DIFS-Zeit oder des Wettbewerbsfensters belegt, darf die Übertragung des Datenpaketes nicht freigegeben werden. Bei Freigabe des Sendevorgangs fließt eine Marke auf den Platz „Enable“. Durch diesen Platz warten Datenpakete, Beacons und ACKs so lange auf dem Platz „Wait_for_Access“, bis eine Marke für eine entsprechende Freigabe vorliegt. Bei der Belegung des Platzes „Enable“ mit einer Marke, wird das aktuell zu übertragende Paket an den Physical Layer übergeben.

Als erstes wird ein Paket, welches vom PHY zurückgegeben wird, dahingehend überprüft, ob es von Paketkollision betroffen war. Für diese Überprüfung dient der untere Bereich des DLLs (siehe Abbildung 79). Weiterhin ist festzustellen, um welchen Pakettyp es sich handelt. Dabei wird eine Unterscheidung zwischen Datenpaket, Beacon und ACK getroffen. Da die korrekte Übertragung eines Beacons nicht mit einem ACK bestätigt wird, spielt es keine Rolle, ob dieser Pakettyp Kollisionen ausgesetzt war. Es wird sofort eine neue Freigabe-Marke generiert. Handelt es sich bei den vom PHY zurückgegebenen Paket um ein ACK, wird im Falle einer Kollision die Subtransition „No_Ack_Retry“ ausgeführt.

Daten- und ACK-Pakete beinhalten das Attribut „att“. Dieses gibt die Anzahl der Wiederholungen für dieses Paket wieder. Wenn in einem Paket der Wert dieses Attributes ein festgelegtes Maximum überschritten hat, wird das Paket verworfen. Wenn andererseits der Maximalwert noch nicht erreicht ist, wird nach einer Kollision eine erneute Übertragung des zum ACK gehörigen Datenpaketes initiiert. Bei einem kollisionsfreiem ACK wird umgehend eine neue Freigabe-Marke erzeugt und der Medienzugriff für das nächste Paket gestartet. Auch ein Datenpaket wird identisch zum ACK im Kollisionsfall so oft wiederholt, bis der festgelegte Maximalwert des Attributes „att“ erreicht wird. Nachdem das Datenpaket kollisionsfrei übertragen wurde, wird ein ACK generiert. Dieses ACK wird auf den Platz „Wait_for_Access“ gelegt und erhält dort nach Ablauf der SIFS-Zeit die Sendefreigabe.

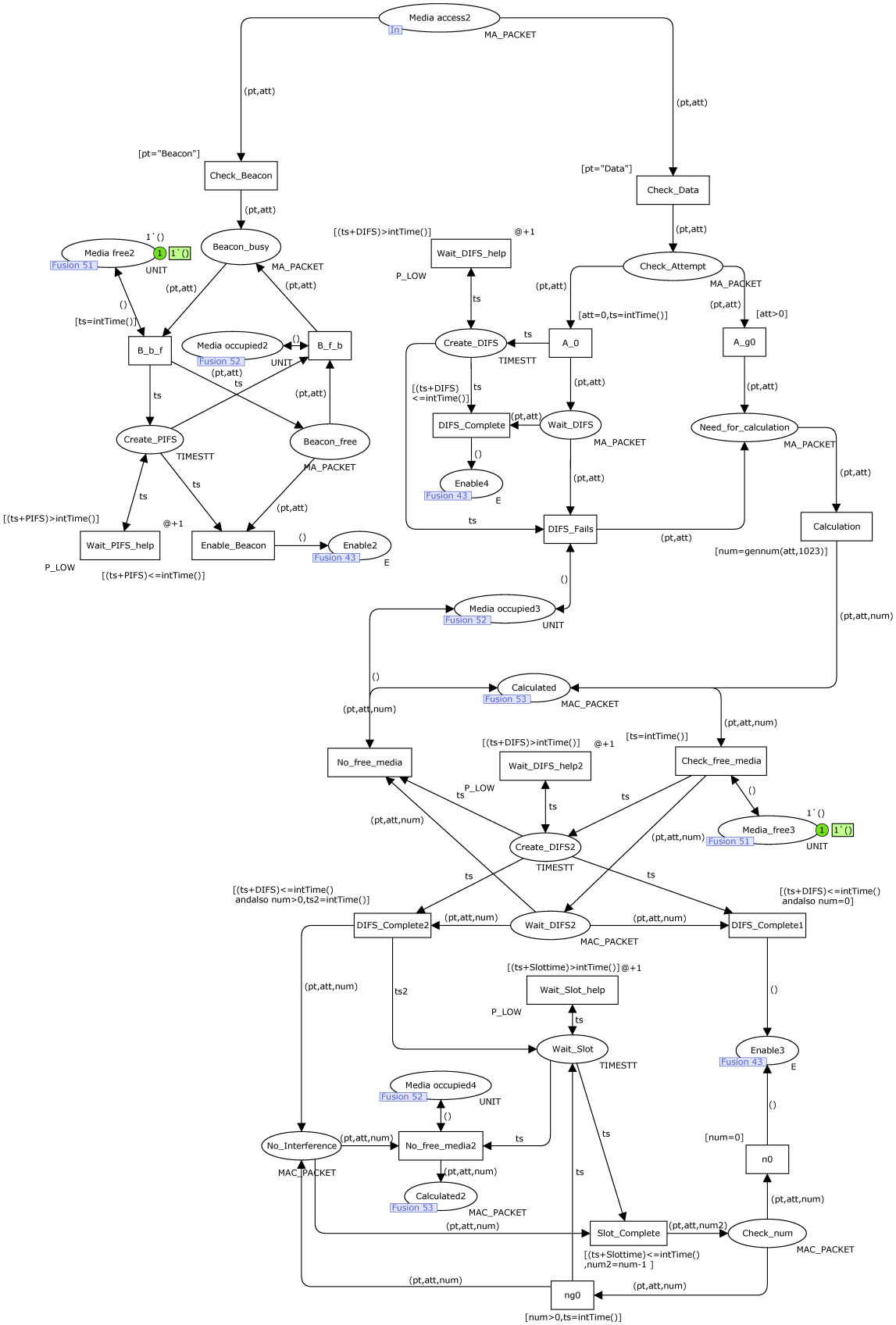

Abb. 80: *Umsetzung des Medienzugriffs*

8.3.4 Interferenz-Check

Ein wesentlicher Bestandteil des Koexistenzmodells, welcher bereits bei der Beschreibung des Systemmodells eingeführt wurde, ist der Interferenz-Check. Dieser ist mit der Transition „Int-Check“ umgesetzt worden. Dabei wird mit jedem Feuern der Transition eine Funktion zur Überprüfung der Interferenz ausgeführt. Das Medium besteht generell aus einem Platz und einer leeren Liste, die als Marke auf diesem Platz liegt. Bei der Übertragung eines Paketes wird dieses vom PHY auf das Medium gelegt. Das Paket wird dabei ein Element der sich auf dem Medium befindenden Liste von Marken. Mit jedem neuen Paket, das auf das Medium gelegt wird, schaltet die Transition „Int-Check“ und die dazugehörige Funktion wird abgehandelt. Besitzt die Liste des Mediums nur ein Paket, ist keine Interferenz feststellbar. Interferenzen können nur bei zwei oder mehr Paketen, die sich gleichzeitig auf dem Medium befinden, auftreten. In diesem Fall muss unterschieden werden, um welchen Systemtyp es sich bei den einzelnen Paketen handelt. Sollten sich nur WSAN-FA- oder nur WLAN-Pakete auf dem Medium befinden, ist nur ein Vergleichen der Kanalnummern notwendig. Stimmen die Kanalnummern mindestens zweier Pakete überein, wird der Wert des Attributes „Intf“ aller betroffener Pakete auf „true“ gesetzt. Sollten sowohl WLAN- als auch WSAN-FA-Pakete auf dem Medium liegen, müssen die unterschiedlichen Bandbreiten beider Funktechnologien berücksichtigt werden. Die WLAN-Bandbreite beträgt für die IEEE 802.11 b/g-Spezifikation 20 MHz, für WSAN-FA dagegen 1 MHz. WLAN deckt somit mehrere WSAN-FA-Kanäle ab. Bei dem zu Grunde liegenden Modellansatz handelt es sich um eine Vereinfachung bei der Interferenz-Überprüfung. Dabei wird die Signaldämpfung bei der Ausbreitung ignoriert. Die Störwahrscheinlichkeit ist entlang der gesamten Bandbreite gleichbleibend. Dadurch wird ein WLAN-Signal durch ein WSAN-FA-Signal am Rand der Spektralmaske genauso gestört wie in der Mitte. Außerdem gelten alle an einer Kollision beteiligten Pakete als zerstört, unabhängig von der zeitlichen Dauer der Überschneidung. Das wird in Abbildung 81 veranschaulicht.

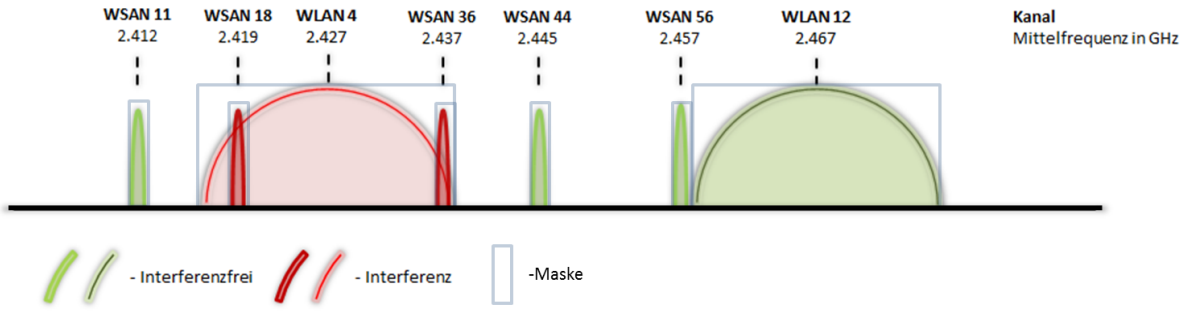

Abb. 81: *Auftreten von Interferenzen*

Zur Interferenz-Prüfung der Pakete beider Funktechnologien wird die Bandbreite als eine Liste von Frequenzen umgesetzt. Wenn gleichzeitig sowohl WLAN- als auch WSAN-FA-Pakete auf dem Funkmedium übertragen werden, müssen beide Frequenzbelegungslisten auf Übereinstimmungen untersucht werden. Wenn bei der Überprüfung gleiche Frequenzen

entdeckt werden, muss bei den dazugehörigen Paketen eine Änderung des Attributes „Intf“ auf „true“ erfolgen.

8.3.5 Bestätigung der Simulationsergebnisse durch Messungen

Verifikation und Validierung der WSAN-FA- und WLAN-Modelle wurden auf identische Weise durchgeführt, wie sie bereits ausführlich bei den Modellen der EN 300 328-Medienzugriffsverfahren beschrieben wurden. Zusätzlich wurde ein Vergleich zwischen Modell und realem System vorgenommen. Da die Funktionsweisen von WSAN-FA und WISA sehr ähnlich sind, lässt sich das WISA-Modell mit geringem Aufwand aus dem WSAN-FA-Modell ableiten. WISA-Funklösungen werden bereits seit einigen Jahren von der Firma ABB vertrieben und daher auch in industriellen Automatisierungsanlagen eingesetzt. Beide Funktechnologien verwenden als Medienzugriffsverfahren TDMA. WSAN-FA und WISA unterscheiden sich zum einen in der Länge des Downlink-Frames. Des Weiteren existieren auch leichte Abweichungen in der Frequenzsprungfolge.

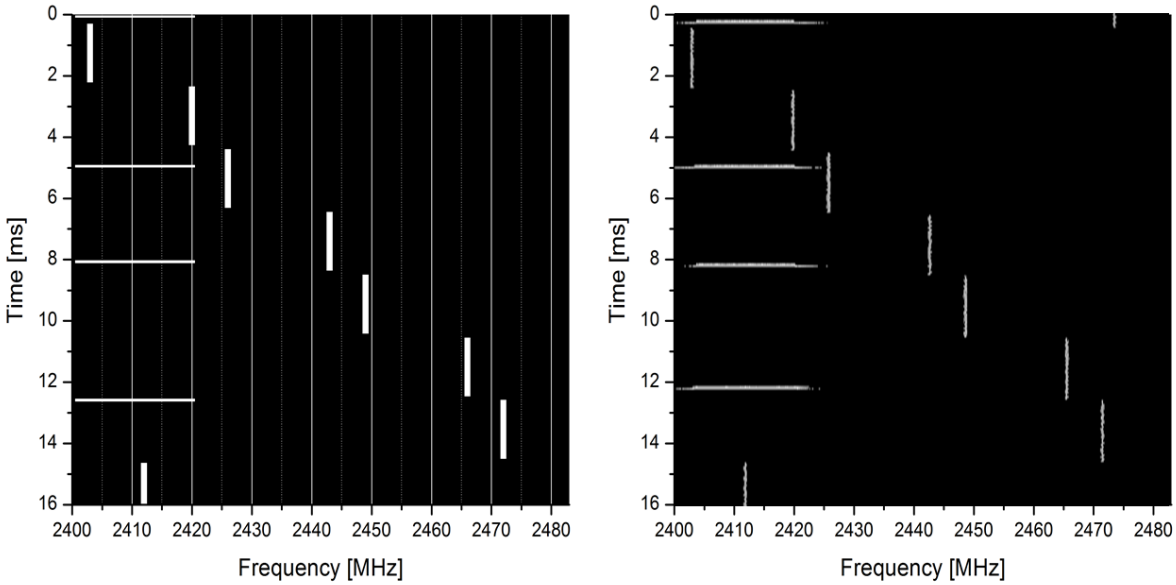

Abb. 82: *Vergleich von Simulation (links) und Messung (rechts)*

Abbildung 82 zeigt sowohl das Spektrogramm von WLAN und WISA als Simulationsergebnis des Petri-Netz-Modells (links) und als Messergebnis, welches von realen Systemen stammt (rechts). Die WLAN-Pakete sind im Spektrogramm links zu sehen. Sie sind zeitlich sehr kurz, haben aber eine große Bandbreite. Die WISA-Pakete des Downlinks sind zeitlich sehr lang, haben eine kleine Bandbreite und hoppen durch das Frequenzband. Die mit den Spektrogrammen gezeigte Übereinstimmung von realer Messung und Simulation beweist zusätzlich die Richtigkeit des erstellten WSAN-FA-Modells.

8.4 Simulation

8.4.1 Allgemein

Nach erfolgreicher Erstellung und Verifikation des WSAN-FA- und WLAN-Modells kann eine Koexistenzbetrachtung mit Hilfe von Simulationen durchgeführt werden. Um aussagekräftige Ergebnisse erzielen zu können, wurden folgende Fälle betrachtet:

- WSAN-FA/WLAN isoliert
- Ein WSAN-FA und drei, das ISM-Band abdeckende WLAN-Systeme
- Ein WLAN und drei unabhängige WSAN-FA-Systeme

Bei den Simulationsszenarien liegt die Variation in der Anzahl und Frequenz der Funkteilnehmer. Auf Seiten des WSAN-FA-Modells werden auch unterschiedliche Zellen-Identifikationsnummern (Cell-IDs) verwendet, welche einen Einfluss auf die Frequenzsprungfolge und somit auch auf das Koexistenzverhalten haben.

In allen Simulationsszenarien besteht jedes verwendete WSAN-FA-Funksystem aus einer Basisstation und sechs angebundenen SA-Geräten. Die SA-Adressen innerhalb der einzelnen Systeme sind aus Gründen der Vergleichbarkeit fest eingestellt. Sie wurden allerdings im Vorfeld zufällig bestimmt. In Tabelle 12 sind die genutzten Adressen der Sensoren/Aktoren aufgeführt.

Tab. 12: *Zuordnung der SA-Adressen*

WSAN-FA-Zelle	SA-Adressen
WSAN-FA 1	82, 109, 80, 104, 44, 31
WSAN-FA 2	78, 72, 73, 37, 35, 108
WSAN-FA 3	11, 34, 9, 16, 111, 105

Jedes WSAN-FA-Datenpaket wird maximal dreimal wiederholt. Nach der dritten fehlgeschlagenen Wiederholung gilt es als zerstört. Bei der Generierung eines aktuelleren Datenpaketes für eine SA-Adresse durch den Application Layer wird ein vorhandenes Paket, welches sich im Buffer befindet und für dieselbe SA-Adresse vorgesehen ist, verworfen. Dabei kann es sich auch um ein Retry-Paket handeln. Das neu erstellte Paket geht anstelle des alten Telegramms in den Buffer.

Beim WLAN-Modell ist festzuhalten, dass dieses immer aus einem Access Point und einem Client besteht. Der Access Point sendet im Gegensatz zu dem Client neben den

Datenpaketen (P) auch noch Beacon-Pakete (B). Die Medienbelegungszeiten für die einzelnen Pakettypen des WLANs sind in Tabelle 13 aufgelistet. Die angegebenen Werte der Belegungszeiten stammen von Messreihen mit unterschiedlichen Bitraten. Die für die Simulation ausgewählte Bitrate beträgt 1 MBit/s. Diese wird in der industriellen Automation häufig eingesetzt, da sie als besonders robust gilt.

Tab. 13: *Medienbelegungszeit des WLAN-Systems*

Pakettyp	Medienbelegungszeit [µs]
Datenpaket	849
Acknowledgement	307
Beacon	945

Auch ein WLAN-Datenpaket wird maximal dreimal wiederholt. Sollte auch bei der dritten Wiederholung die Übertragung fehlschlagen, gilt es als zerstört. Jedes WLAN-System besteht aus zwei Übertragungsrichtungen, die separat modelliert wurden; eine vom Access Point zum Client und die andere in die entgegengesetzte Richtung. Jede Übertragungsrichtung hat einen Buffer für Datenpakete mit einer Größe von drei. Sind bereits drei Pakete im Buffer und wird ein weiteres vom Application Layer erzeugt, wird das älteste Paket im Buffer verworfen.

8.4.2 WSAN-FA/WLAN isoliert

Bevor das Koexistenzverhalten von WSAN-FA und WLAN untersucht wird, werden die beiden Technologien einzeln ohne gegenseitige Beeinflussung betrachtet. Die mit diesem Simulationsszenario gewonnenen Ergebnisse stellen das ideale Übertragungsverhalten beider Systeme dar. Tabelle 14 listet die Parameter auf, die für die Simulation verwendet wurden.

Tab. 14: *Parameter des Simulationsszenarios: "WSAN-FA/WLAN isoliert"*

System	Cell-ID	t_{TI} [ms]	Init [ms]	Kanal	Pakete
WSAN-FA	1	10	random(t_{TI})	zufällig	12.000
WLAN	-	32	random(t_{TI})	1	500
WLAN-AP	-	32 P/100 B	random($t_{TI}(P)$)	1	500

Die Abbildungen 83 und 84 zeigen die Häufigkeitsverteilungen der Übertragungszeit für beide Technologien. Da der Sendezeitabstand ausreichend groß gewählt wurde und kein

anderes System Interferenzen hervorrufen konnte, wurden alle Pakete korrekt übertragen. Die Verteilungen der Übertragungszeit sind somit die einzigen darstellbaren Ergebnisse dieses Simulationsszenarios.

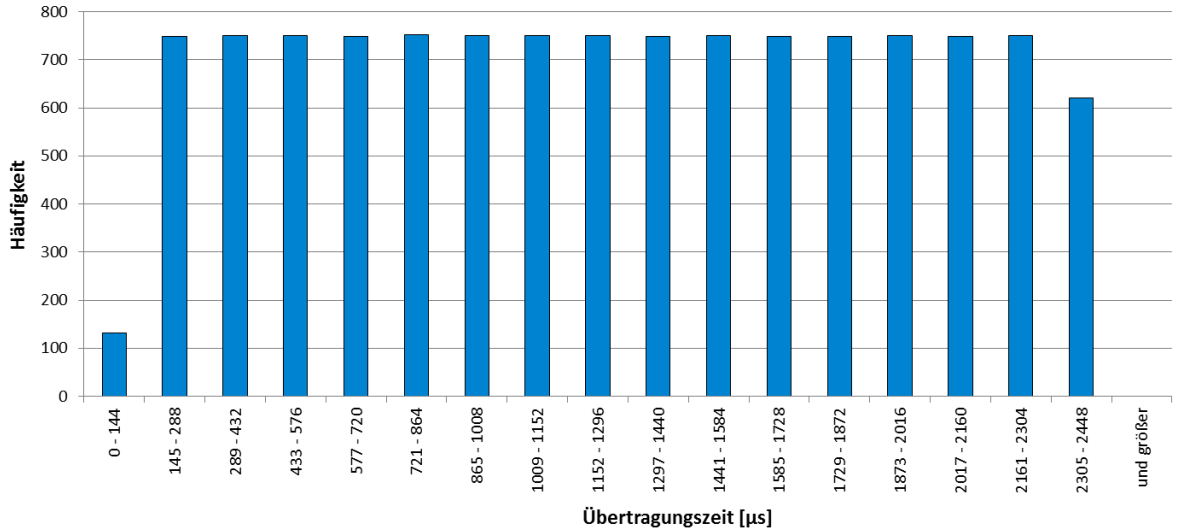

Abb. 83: *Übertragungszeiten von WSAN-FA im Szenario: "WSAN/WLAN isoliert"*

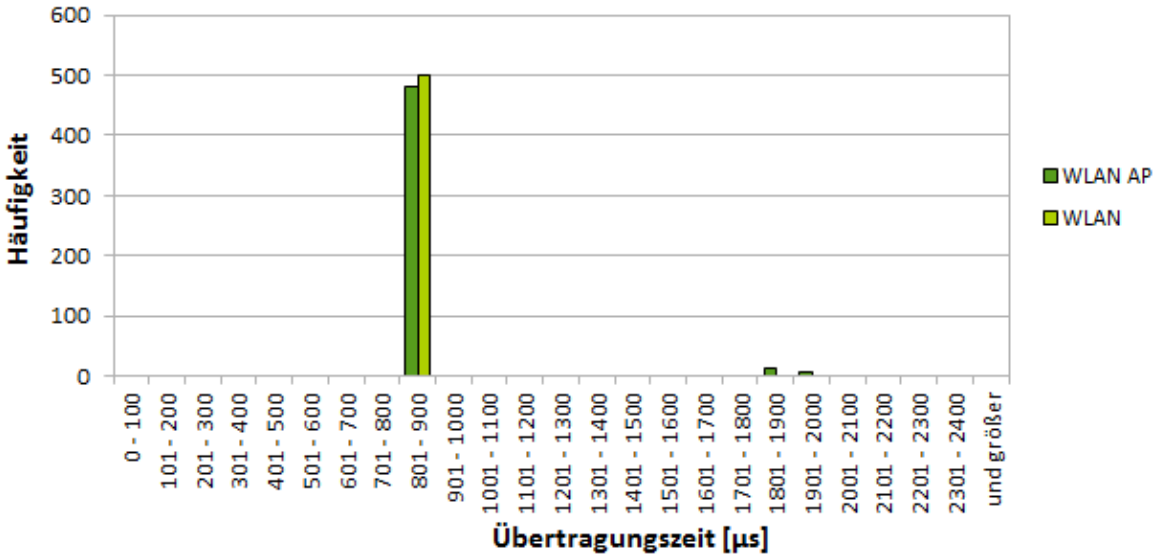

Abb. 84: *Übertragungszeiten von WLAN im Szenario: "WSAN/WLAN isoliert"*

Die Verteilung der Übertragungszeit des WSAN-FA-Systems in Abbildung 83 zeigt den reinen Einfluss des Medienzugriffsverfahrens. Andere Einflussgrößen spielen keine Rolle. Das von WSAN-FA genutzte TDMA weist jeder Verbindung einen festen Sendeslot zu. Da sich der Zyklus der ständig wiederkehrenden Übertragungsslots vom Zyklus der Paketgenerierung im APPL unterscheidet, müssen die Pakete unterschiedlich lange im Zwischenspeicher verweilen. Dieser Effekt wird als „Asynchronität zyklischer Prozesse" bezeichnet und ist typisch für TDMA-basierte Funksysteme. Diese Asynchronität führt

zu einem sägezahnartigen Verlauf von aufeinanderfolgenden Übertragungszeiten (siehe Abbildung 85). Im Histogramm ergibt sich dadurch eine Gleichverteilung. Diese hat ein Fenster von 144 µs bis 2448 µs. Die Breite des Fensters entspricht genau der Länge des TDMA-Frames (2304 µs). Der Minimalwert der Verteilung ist identisch zur maximalen Medienbelegungszeit eines Paketes (144 µs).

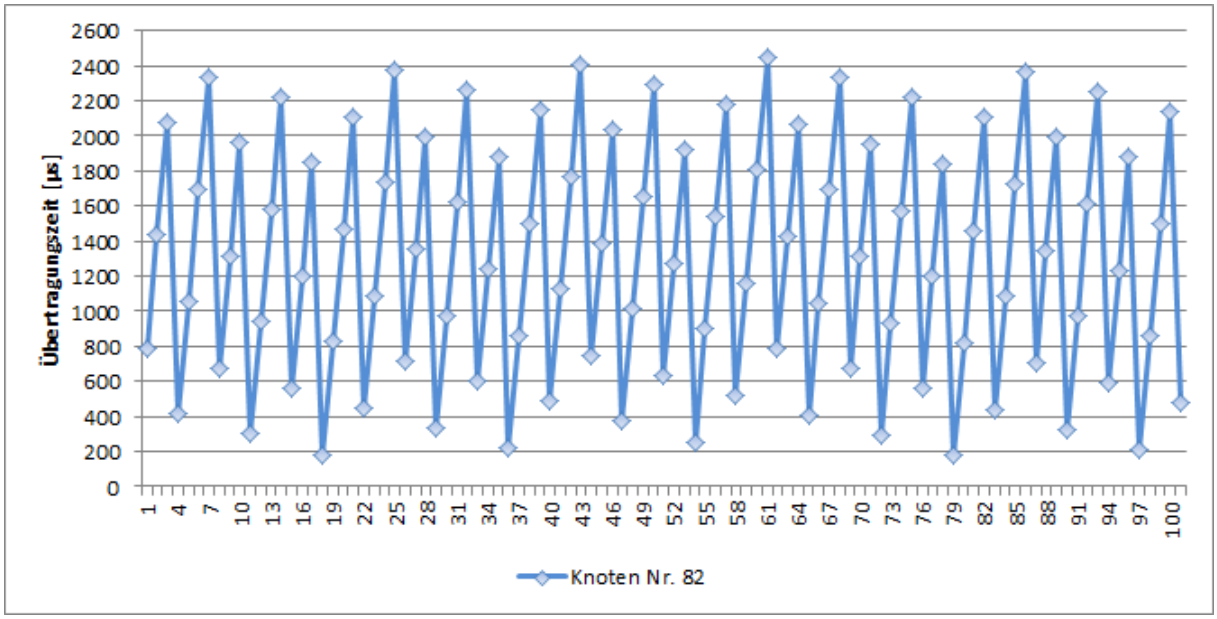

Abb. 85: *Sägezahnförmiger Verlauf der Übertragungszeiten von WSAN-FA*

Abbildung 84 zeigt das Zeitverhalten des WLAN-Systems. Alle Übertragungszeiten des Clients haben einen festen Wert von 899 µs. Dieses entspricht der Summe aus Medienbelegungszeit eines Datenpaketes und der DIFS-Zeit. Die Übertragungszeiten des APs befinden sich in einem Bereich zwischen 899 µs und 1943 µs. Die Ursache für teilweise größere Übertragungszeiten liegt in der Aussendung von Beacons. Da Beacons eine höhere Priorität als Datenpakete haben, müssen Datenpakete bei der Übertragung von Beacons länger im Zwischenspeicher warten.

8.4.3 Ein WSAN-FA und drei, das ISM-Band abdeckende WLAN-Systeme

Tab. 15: *Parameter des Szenarios: "Ein WSAN-FA und drei, das ISM-Band abdeckende WLAN-Systeme"*

System	Cell-ID	t_{TI} [ms]	Init [ms]	Kanal	Pakete
WSAN-FA 1	1	10	random(t_{TI})	zufällig	13.000
WLAN 1	-	32	random(t_{TI})	2	500
WLAN-AP 1	-	32 P/100 B	random($t_{TI}(P)$)	2	500
WLAN 2	-	32	random(t_{TI})	6	500
WLAN-AP 2	-	32 P/100 B	random($t_{TI}(P)$)	6	500
WLAN 3	-	32	random(t_{TI})	11	500
WLAN-AP 3	-	32 P/100 B	random($t_{TI}(P)$)	11	500

Dieses Simulationsszenario dient der Überprüfung der Robustheit des WSAN-FA-Systems. Dabei werden die drei WLAN-Systeme über die Einstellung der Funkkanäle so angeordnet, dass eine vollständige Belegung des 2,4 GHz-ISM-Bandes vorliegt. Weiterhin soll untersucht werden, inwieweit die vier Anforderungen, die schon an die EN 300 328 gestellt wurden (siehe Abschnitt 6.5), erfüllt werden können. Der Fokus liegt dabei auf der gleichmäßigen Spektrenaufteilung zwischen den Teilnehmern der verschiedenen Funksysteme. Die genaue Einstellung der Parameter ist Tabelle 15 zu entnehmen.

Die Ergebnisse sind in den Abbildungen 86, 87 und 88 dargestellt.

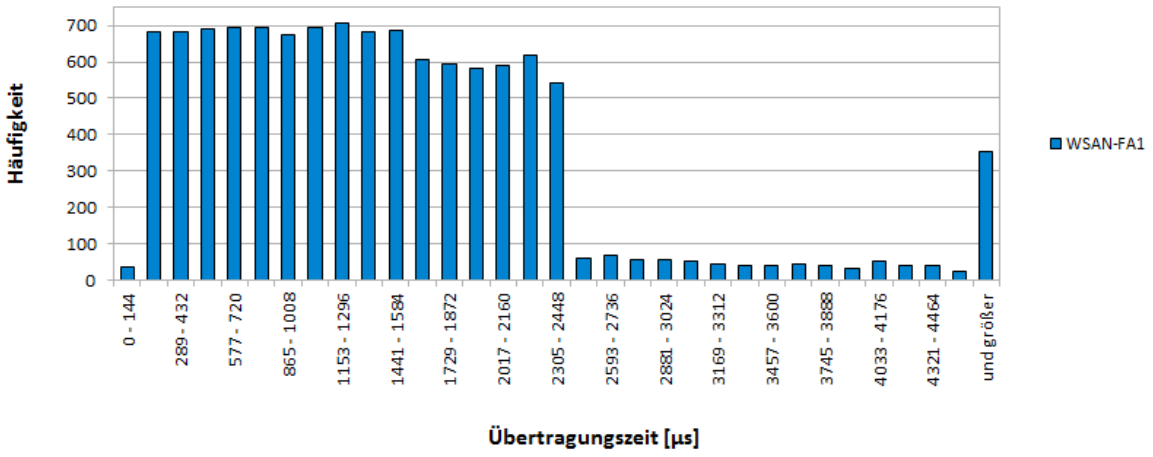

Abb. 86: *Übertragungszeiten von WSAN-FA im Szenario: "1xWSAN-FA/3xWLAN"*

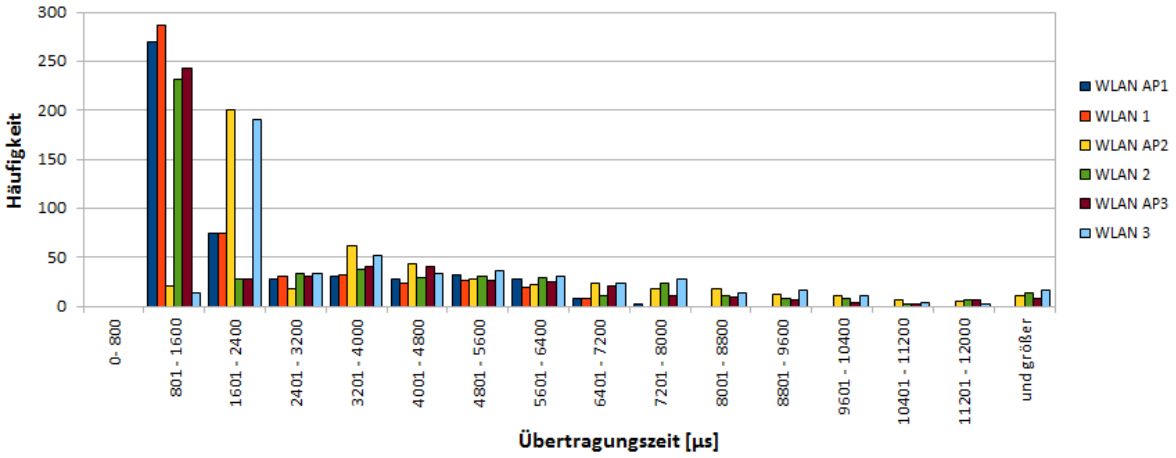

Abb. 87: *Übertragungszeiten von WLAN im Szenario: "1xWSAN-FA/3xWLAN"*

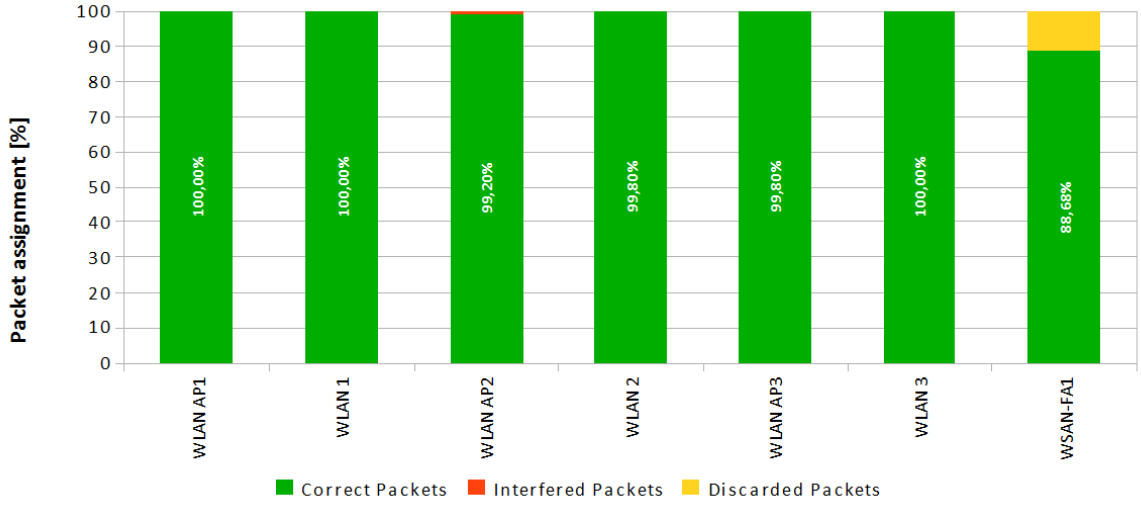

Abb. 88: *Paketverteilung im Szenario: "1xWSAN-FA/3xWLAN"*

Die Ergebnisse zeigen, dass trotz vollständiger Belegung des 2,4 GHz-Bandes durch drei parallele WLAN-Systeme ein WSAN-FA-System mehr als 88 % der Pakete korrekt übertragen kann. Die Übertragungszeiten des WSAN-FA-Systems liegen zwischen 72 und 11534 µs. Allerdings wird der größte Teil der Pakete schon innerhalb des ersten Downlink-Frames erfolgreich übertragen. Bei den WLAN-Systemen gehen weniger als 1 % der Pakete verloren. Die Übertragungszeiten liegen zwischen 899 und 21035 µs. Aber auch bei WLAN wird ein großer Teil der Pakete innerhalb der ersten 2,4 ms übertragen. An dieser Stelle sei nochmal darauf hingewiesen, dass die Interferenzbetrachtung einem Worst Case entspricht. Jeder Teilnehmer „sieht" jeden anderen. Zwei Pakete müssen sich nur minimal zeit- und frequenzmäßig überschneiden, damit sie als gestört markiert und anschließend erneut übertragen werden. In der Realität wird das schmalbandige WSAN-FA-Signal weitaus weniger gestört. Auch ist die Beeinflussung von WLAN-Systemen durch dieses Signal geringer. Bezogen auf die Anforderungen der EN 300 328 muss festgehalten werden, dass das Spektrum effizient genutzt und gleichmäßig aufgeteilt wird. Allerdings wird das Medi-

um auch nicht an der Leistungsgrenze betrieben. Die Vorhersehbarkeit des Zeitverhaltens von WLAN ist eingeschränkt, da der Medienzugriff eine zufällige Wartezeitkomponente besitzt und durch WSAN-FA-Pakete verzögert wird. WSAN-FA ist durch die Verwendung von TDMA im Zeitverhalten weitestgehend stabil und vorhersehbar.

8.4.4 Ein WLAN und drei unabhängige WSAN-FA-Systeme

Tab. 16: *Parameter des Szenarios: "Ein WLAN und drei unabhängige WSAN-FA-Systeme"*

System	Cell-ID	t_{TI} [ms]	Init [ms]	Kanal	Pakete
WSAN-FA 1	4	10	random(t_{TI})	zufällig	13.000
WSAN-FA 2	12	10	random(t_{TI})	zufällig	13.000
WSAN-FA 3	38	10	random(t_{TI})	zufällig	13.000
WLAN 1	-	32	random(t_{TI})	1	500
WLAN-AP 1	-	32 P/100 B	random($t_{TI}(P)$)	1	500

Dieses Szenario soll untersuchen, wie sich ein WLAN-Funksystem verhält, wenn sich in näherer Umgebung drei WSAN-FA-Zellen befinden. Die Einstellungen der Systeme ähneln dabei denen aus dem vorherigen Simulationsfall. Der Kanal des WLANs ist auf einen Wert von eins eingestellt. Die Cell-IDs der WSAN-FA-Systeme variieren. In Tabelle 16 sind die einzelnen Parameter aufgelistet.
Die Ergebnisse sind in den Abbildungen 89, 90 und 91 dargestellt.

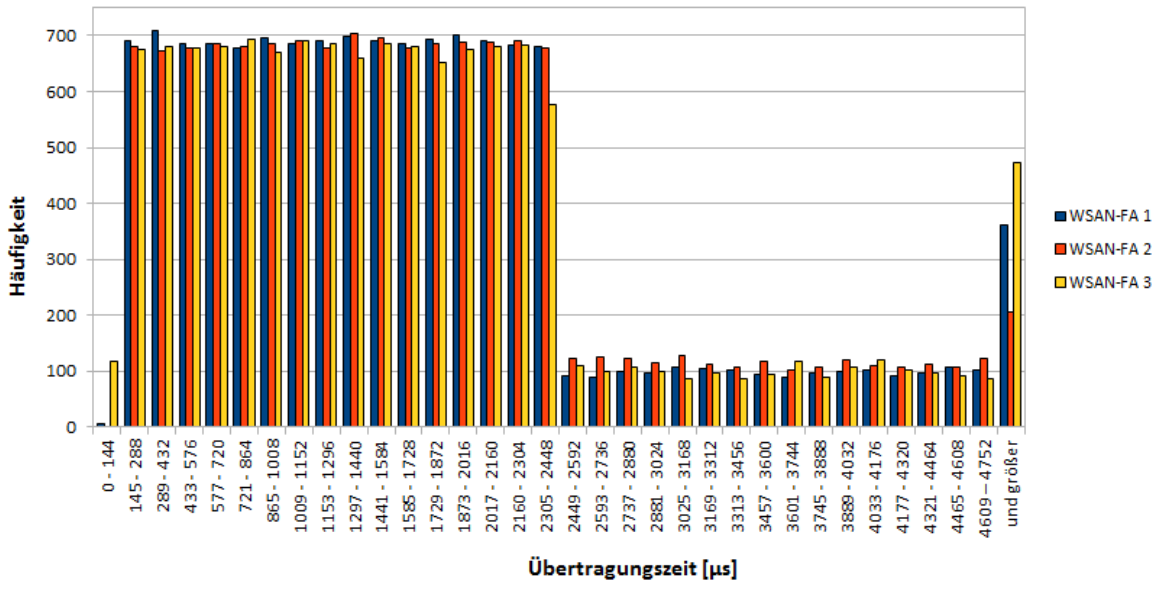

Abb. 89: *Übertragungszeiten von WSAN-FA im Szenario: "3xWSAN-FA/1xWLAN"*

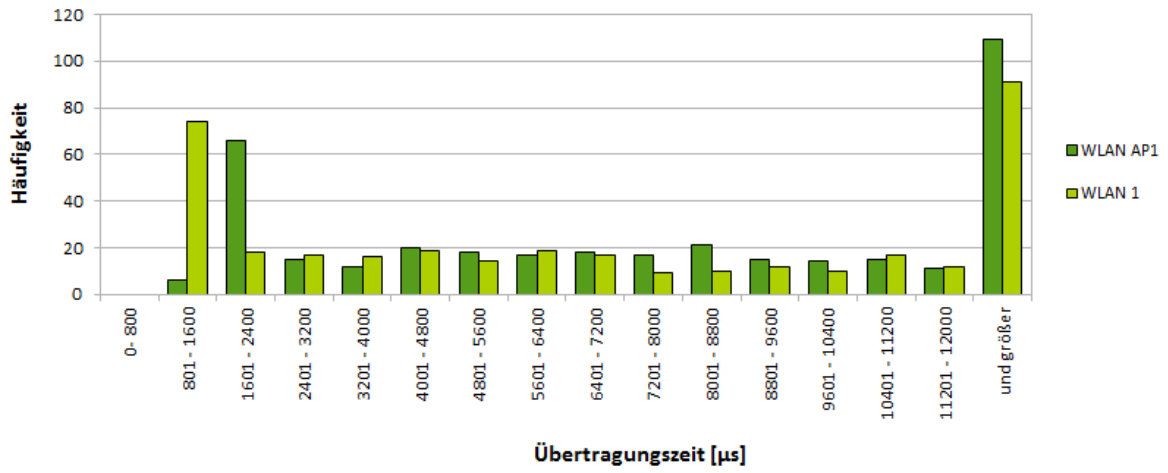

Abb. 90: *Übertragungszeiten von WLAN im Szenario: "3xWSAN-FA/1xWLAN"*

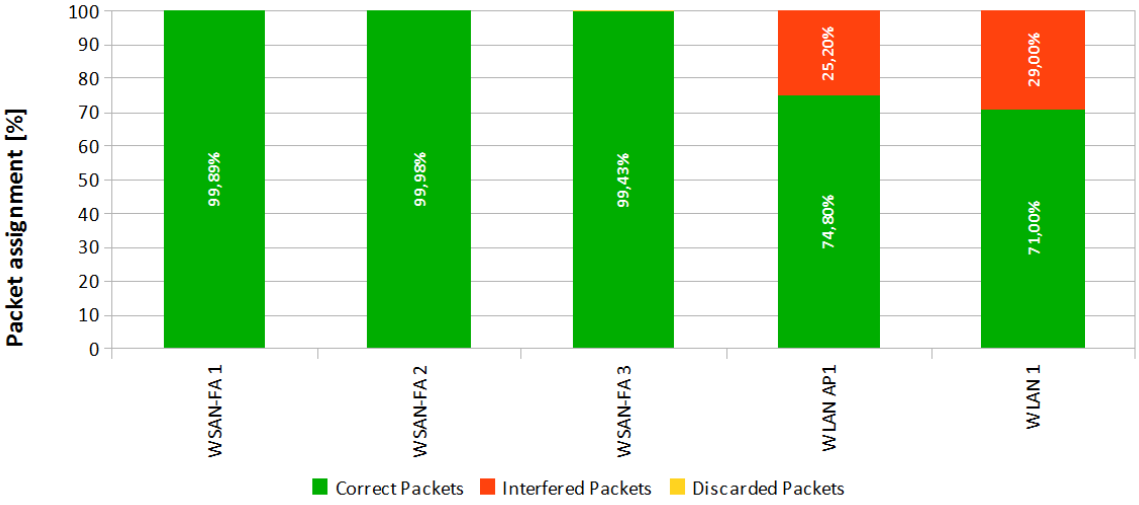

Abb. 91: *Paketverteilung im Szenario: "3xWSAN-FA/1xWLAN"*

Die Ergebnisse zeigen, dass das WLAN-System stark von den drei WSAN-FA-Systemen gestört wird, da mehr als 25 % der Pakete verloren gehen. Diese Beeinflussung zeigt sich auch in den WLAN-Übertragungszeiten. Ein Großteil der Pakete wird um mehr als 12 ms verzögert. Die WSAN-FA-Systeme werden von dem WLAN-System nur sehr wenig beeinflusst. Es werden kaum Pakete zerstört. Die Übertragungszeiten befinden sich überwiegend innerhalb der Länge eines Downlink-Frames.

8.5 Zusammenfassung

Mit dem kolorierten Petri-Netz-Modell wurden erfolgreich Koexistenzbetrachtungen zwischen WSAN-FA und WLAN durchgeführt. WLAN und WSAN-FA sind Beispiele für Funksysteme, die für einen realen Einsatz in der industriellen Automation ausgelegt sind. Die durchgeführten Simulationen und deren Auswertungen zeigen, dass das WSAN-FA-Funksystem durchaus in der Lage ist, die drahtlose industrielle Automation sicherer

und robuster zu gestalten. Das Frequenzsprungverfahren des WSAN-FA-Systems und das schmalbandige Signal sind optimal, um möglichst viele Sensoren ansteuern zu können. Die geschaffene Robustheit verhindert, dass zu viele Pakete neu übertragen werden müssen. Durch die störungsfreie Übertragung ist die Echtzeitfähigkeit gesichert und Fehlfunktionen durch zu lange Übertragungszeiten werden vermieden. Die Simulationsszenarien haben aber auch gezeigt, dass die Kommunikation von WSAN-FA nur ideal funktioniert, wenn freie Bereiche im verwendeten Frequenzband existieren. Es wurde zudem festgestellt, dass WLAN-Systeme durch WSAN-FA beeinflusst werden. Dabei werden nicht nur Pakete zerstört, sondern auch der Medienzugriff verzögert. Das geschieht, wenn beispielsweise ein Downlink-Frame im WLAN-Frequenzbereich aktiv ist. Das WLAN erkennt dies als Belegung. Die Verzögerungen können dadurch mehr als 2 ms betragen. Das WLAN-System könnte in einem Anwendungsszenario die Aufgabe zugeteilt bekommen, die durch die WSAN-FA-Sensoren/Aktoren entstehenden Datenmengen zu sammeln und weiterzuleiten. Wie die Simulationen gezeigt haben, wurde das WLAN durch WSAN-FA zwar beeinträchtigt, die Daten konnten dennoch ausreichend zuverlässig übertragen werden.

Eine weitere Möglichkeit der Verbesserung wäre, anstelle des FH02- das FH07-Frequenzsprungverfahren einzusetzen. Dieses Sprungverfahren unterstützt das Ausschließen von Bereichen innerhalb des verwendeten Frequenzbandes. Der Vorteil dabei ist, dass bei Wissen über die Frequenzbereiche von bereits in der Umgebung befindlichen Funksystemen, diese für den Frequenzsprung des WSAN-FA ausgelassen werden können. Die gegenseitigen Beeinträchtigungen mit anderen Funksystemen könnten damit weiter reduziert werden.

9 Integration eines Kanalmodells

9.1 Allgemein

In den vorangegangenen Simulationen wurden bisher die räumliche Verteilung der Funkteilnehmer und die damit verbundenen Übertragungsverluste aufgrund von verschiedenen Ausbreitungswegen des Nutzsignals nicht berücksichtigt. Die Modellierung des Funkmediums erfolgte nach dem Prinzip: „Jedes Funkgerät sieht jedes andere, wenn beide denselben Frequenzbereich verwenden." In diesem Abschnitt soll das bisherige Modell um diese Übertragungsverlustbetrachtung erweitert werden. Die Entscheidung über das Auftreten von Interferenzen erfolgt damit auf der Grundlage von Nutz- und Störleistung am entsprechenden Empfänger. Mit der Berücksichtigung von möglichen Effekten im Übertragungskanal werden die Simulationsergebnisse noch realistischer.

Die Übertragung von Informationen via Funk ist gleichzeitig die Übertragung von elektromagnetischer Energie. Aufgrund verschiedener Effekte bei der Ausbreitung der Funkwellen kommt nur ein Bruchteil der Sendeleistung auch am Empfänger an. Bei einer direkten Sichtverbindung (Line-Of-Sight, LOS) ist der einzige leistungsmindernde Effekt die Dämpfung infolgedessen, dass sich die Sendeenergie kugelförmig im Raum verteilt und deshalb nicht komplett auf den Empfänger trifft. Dieser Fall trägt die Bezeichnung Freifelddämpfung oder auch Free-space path loss.

Bei der Ausbreitung im freien Raum gibt es keine Hindernisse zwischen der Sende- und der Empfangsantenne, trotzdem wird hier die Strahlungsleistung entlang des Ausbreitungsweges gedämpft. Die folgende Herleitung der Freiraumdämpfung stammt aus [22].
Eine Kugel mit dem Radius r gleich der Verbindungsstrecke d zwischen Sender und Empfänger wird von der kompletten abgestrahlten Sendeleistung P_S durchdrungen. Die Leistungsdichte S ergibt sich damit durch

$$S = \frac{P_S}{4\pi d^2} \tag{69}$$

Im Abstand d entnimmt die Empfangsantenne dem elektrischen Feld die Leistung

$$P_E = SA_E \tag{70}$$

A_E ist dabei die Apertur (effektive Antennenwirkfläche)

$$A_E = \frac{\lambda^2}{4\pi} \tag{71}$$

mit der Wellenlänge λ. Unter Berücksichtigung der Verstärkung von sowohl Sende- (G_S) als auch Empfangsantenne (G_E) ergibt sich nun die Friis-Gleichung oder auch Freiraumformel

$$P_E = P_S G_S G_E \left(\frac{\lambda}{4\pi d}\right)^2 \tag{72}$$

die nur im Fernfeld, d. h. für $d \geq 2\lambda$, Gültigkeit besitzt. Hier transportiert die elektromagnetische Welle reine Wirkleistung, da elektrisches und magnetisches Feld in Phase sind. Häufig ist es zweckmäßig, das Verhältnis P_S/P_E als Pfadverlust in dB anzugeben.

$$L_F[\text{dB}] = \log\frac{P_S}{P_E} = 32{,}4 + 20\log\frac{d}{\text{km}} + 20\log\frac{f}{\text{MHz}} - 10\log G_S - 10\log G_E \qquad (73)$$

Weiterhin lässt Gleichung 72 erkennen, dass die Empfangsleistung quadratisch mit dem Abstand d abnimmt. Verdoppelt sich also die Entfernung, so erhöht sich die Dämpfung um 6 dB.

In den seltensten Fällen breitet sich eine Funkwelle im freien Raum aus. Meist wird der direkte Ausbreitungsweg durch Hindernisse wie Gebäude im Außenbereich oder Rohre, Leitungen, Maschinen und Fahrzeuge in einer industriellen Umgebung gestört. An diesen Objekten können weitere Effekte bei der Signalausbreitung auftreten. Im Folgenden werden diese Effekte aufgelistet und genauer erläutert ([22], [40], [59], [36]):

- **Absorption:** Zusätzlich zu der im Freiraum beschriebenen Dämpfung werden Funkwellen ebenfalls beim Durchdringen von verlustbehafteten Medien wie Wänden durch Absorption abgeschwächt. Während eines solchen Durchdringens wird dabei die absorbierte Energie in Wärme überführt und sorgt für einen exponentiellen Abfall der Wellenamplitude entlang des Ausbreitungswegs. Bei den später beschriebenen Berechnungen kann ein Verlust an normalen Gebäudeaußenwänden von $L_{wall} = 10$ dB angenommen werden.

- **Reflexion:** Trifft eine homogene ebene Welle auf eine ebene Grenzschicht zweier Medien, so wird sie reflektiert. Dabei ist der Einfallwinkel gleich dem Ausfallwinkel.

- **Brechung:** Beim Durchgang einer homogenen, ebenen Welle durch eine ebene Grenzschicht zweier Medien erfährt die Welle eine Richtungsveränderung. Diese wird als Brechung bezeichnet.

- **Beugung:** An den Kanten von Hindernissen treten außerdem noch Beugungseffekte auf. Dabei dringt die Welle in den in Ausbreitungsrichtung hinter dem Hindernis liegenden geometrischen Schattenbereich ein. Dieses Phänomen kann mit dem Huygensschen Prinzip veranschaulicht werden, d. h. alle Punkte einer Wellenfront können als Ausgangspunkte neuer Elementarwellen betrachtet werden, die sich dann wiederum zu einer resultierenden Welle überlagern.

- **Streuung:** Wenn elektromagnetische Wellen auf mehrere Objekte, die auch unterschiedlich sein können, treffen, treten die eben beschriebenen Ausbreitungseffekte in vielfacher Weise auf. Das gesamte Wellenfeld ergibt sich dann aus der Überlagerung des einfallenden und des durch z. B. Beugung und Reflektion gebildeten Feldes. Diese Erscheinung wird als Streuung bezeichnet.

9.2 Kanalmodelle

9.2.1 Übersicht

Abgesehen vom einfachen Fall der Freiraumausbreitung ist es schwierig, die Ausbreitung einer Welle physikalisch exakt zu beschreiben. Dies liegt zum einen an den äußert komplexen Berechnungen und zum anderen ist eine Beschreibung der Umgebung mit der dazu nötigen Genauigkeit kaum möglich. Allerdings sind bereits verschiedene empirische oder semi-empirische Modelle vorhanden, die für praktisch relevante Fälle trotz teilweise starker Vereinfachungen ausreichend gute Näherungslösungen liefern.
Es folgt eine tabellarische Übersicht über ausgewählte Kanalmodelle. Anschließend werden mit dem Extended Hata und Extended Hata - Short Range Device (SRD) zwei Kanalmodelle beschrieben, mit denen sich der Ausbreitungsverlust der Sendeleistung in Abhängigkeit von Frequenz, Entfernung, Antennenparametern und vorliegenden Umweltbedingungen berechnen lässt. Dabei wird die Berechnungsvorschrift für den Ausbreitungsverlust bei Freiraumausbreitung zu Grunde gelegt. In Abhängigkeit von den Ausbreitungsbedingungen wurden in den Modellen Konstanten bestimmt, mit denen der Freiraumausbreitungsverlust angepasst wird. Es wurden auch zusätzliche Terme eingeführt.
Bei Anwendung eines Modells ergibt sich der Mittelwert des Ausbreitungsverlustes, der bei den vorherrschenden Bedingungen den Erwartungswert für die Nutzsignaldämpfung bildet. Zusätzlich lässt sich eine Standardabweichung angeben, aus der normalverteilt [Verlust in dB] bzw. lognormalverteilt [Verlust in W] eine additive Variation berechnet wird.

Tabelle 17 zeigt eine Übersicht der Kanalmodelle und ihren Anwendungsbereichen.

9.2.2 Extended Hata Modell

Das Extended Hata Modell wird in [42] beschrieben. Ausbreitungsverluste nach diesem Modell berechnen sich wie folgt:

$$L_{XHata} = L(f, h_1, h_2, d, env) + T(G(\sigma)) \tag{74}$$

Der erste Summand ist der mittlere Verlust L. Dieser ergibt sich je nach Frequenz f, Abstand d und den Bedingungen, die aus der Umgebung resultieren (env), aus verschiedenen Gleichungen in Abhängigkeit von f, d und den Antennenhöhen h_1 und h_2. Dabei sei im Folgenden:

$$H_m = \min(h_1, h_2) \tag{75}$$

$$H_b = \max(h_1, h_2) \tag{76}$$

Tab. 17: *Übersicht der Kanalmodelle*

Modell	Antennenhöhen	Frequenz	Entfernung	In-/Outdoor	Bemerkungen
Free Space Loss	nicht begrenzt	$f \geq$ 30 MHz	LOS begrenzt	indoor-indoor- bzw. outdoor-outdoor-Übertragungen bei direkter Sichtverbindung (LOS)	Systeme, bei denen direkte Sichtverbindung angenommen werden kann
Extended Hata	$h_1 = [1\text{ m}; 10\text{ m}]$ $h_2 =$ $[30\text{ m}; 200\text{ m}]$	30 MHz – 3 GHz	$d \leq 40$ km	automatisches Hinzuaddieren von Verlusten an Wänden bei indoor-outdoor- bzw. indoor-indoor-Übertragungen (mehrere Räume/Gebäude)	mobile Dienste und andere Dienste, ohne direkte Sichtverbindung, nur im Bereich 2000 – 3000 MHz implementiert
Extended Hata-SRD	$h = [1{,}5\text{ m}; 3\text{ m}]$	30 MHz – 3 GHz	$d \leq 0{,}3$ km	automatisches Hinzuaddieren von Verlusten an Wänden bei indoor-outdoor- bzw. indoor-indoor-Übertragungen (mehrere Räume/Gebäude)	Kurzstreckenverbindungen, bei denen wenigstens nahezu LOS angenommen werden kann, wichtig: Antennenhöhe max. 3m
WINNER II Szenario B3 - Indoor Hotspot	$h \leq 6$ m	2 GHz – 6 GHz	$d =$ $[5\text{ m}; 100\text{ m}]$	indoor-indoor-Übertragungen bei LOS oder NLOS	Anwendung in großen, offenen Räumen bei hohem Datenverkehr (z. B. Konferenzhalle, Fabrikgebäude)

Die Gleichungen (75) und (76) ordnen der geringeren Antennenhöhe die mobile Station und der größeren Antennenhöhe die Basisstation zu. Die Übertragung ist unabhängig davon in beide Richtungen möglich. Es wird empfohlen, dass bei der Verwendung des Extended Hata Modells die Bedingungen 1 m $\leq H_m \leq$ 10 m und 30 m $\leq H_b \leq$ 200 m zutreffen.
Die Umweltbedingungen umfassen Angaben zur generellen Umgebung (Urban, Suburban, Rural), zur lokalen Umgebung der jeweiligen Antennen (Indoor/Outdoor) und zur Ausbreitungsumgebung (Above Roof/Below Roof).

- Für $d \leq 0,04$ km:

$$L = 32,4 + 20\log(f) + 10\log\left[d^2 + \frac{(H_b - H_m)^2}{10^6}\right] \tag{77}$$

  (Bei kleinen Abständen zwischen Sender und Empfänger gilt für den Ausbreitungsverlust die Berechnungsvorschrift für Freiraumausbreitung.)

- Für $0,04$ km $< d < 0,1$ km:

$$L = L(0,04) + \frac{\log(d) - \log(0,04)}{\log(0,1) - \log(0,04)} \cdot [L(0,1) - L(0,04)] \tag{78}$$

- Für $d \geq 0,1$ km:
  - urban:

$$\begin{aligned} L = 46,3 + 33,9\log(2000) + 10\log\left(\frac{f}{2000}\right) - 13,82\log(\max\{30,H_b\}) \\ +[44,9 - 6,55\log(\max\{30,H_b\})](\log(d))^{\alpha} - a(H_m) - b(H_b) \end{aligned} \tag{79}$$

    (nur gültig für 2000 MHz $< f \leq$ 3000 MHz)

  - suburban:

$$L = L(urban) - 2\left[\log\left(\frac{\min\{max\{105,f\},2000\}}{28}\right)\right]^2 - 5,4 \tag{80}$$

  - rural:

$$\begin{aligned} L = L(urban) - 4,78\left[\log\left(\min\{max\{105,f\},2000\}\right)\right]^2 \\ +18,33\log\left(\min\{max\{105,f\},2000\}\right) - 40,94 \end{aligned} \tag{81}$$

Für die Parameter $a(H_m)$ und $b(H_b)$ gilt:

$$\begin{aligned} a(H_m) = (1,1\log(f) - 0,7) \cdot \min\{10,H_m\} \\ -(1,56\log(f) - 0,8) + \max\left\{0,20\log\left(\frac{H_m}{10}\right)\right\} \end{aligned} \tag{82}$$

$$b(H_b) = \min\left\{0,20\log\left(\frac{H_m}{30}\right)\right\} \tag{83}$$

Der Exponent α besitzt einen Wert von 1 für Distanzen unter 20 km.

Neben dem mittleren Verlust L existiert ein Term, welcher gemäß der Gaußschen Normalverteilung eine zufällige Abweichung nachbildet ($T(G(\sigma))$). Die Streuung des zufälligen Terms ist abhängig vom Abstand zwischen Sender und Empfänger und den Ausbreitungsbedingungen (Above Roof, Below Roof). Die Werte für die Standardabweichung variieren von $\sigma = 3,5$ bei Entfernungen von $d \leq 0,04$ km über $\sigma = 17$ bei Entfernungen von $0,1 \text{ km} < d \leq 0,2$ km (Below Roof Ausbreitung) bis hin zu $\sigma = 9$ bei Entfernungen von $d > 0,6$ km.

Zusätzlich muss der Ausbreitungsverlust bei indoor-outdoor- und indoor-indoor-Ausbreitung weiter angepasst werden. Bei indoor-outdoor-Ausbreitung geschieht dies durch Hinzuaddieren eines vorher definierten Verlustwertes L_{we}, der beim Durchdringen einer Außenwand entsteht. Meist wird dafür ein Wert von $L_{we} = 10$ dB angenommen. Die Standardabweichung ist wie folgt anzupassen:

$$\sigma_{indoor-outdoor} = \sqrt{\sigma^2_{outdoor-outdoor} + \sigma^2_{add}} \tag{84}$$

wobei $\sigma_{outdoor-outdoor}$ die oben beschriebene Grundstandardabweichung darstellt und σ_{add} wiederum ein vordefinierter Wert ist, der in der Regel als $\sigma_{add} = 5$ dB angegeben wird.
Im Fall der indoor-indoor-Ausbreitung wird zunächst überprüft, ob sich Sende- und Empfangsantenne im selben oder in verschiedenen Gebäuden befinden. Wenn die Antennen in verschiedenen Gebäuden angeordnet sind, so ist das Vorgehen ähnlich, wie oben bei der indoor-outdoor-Ausbreitung beschrieben. Allerdings wird vom Durchdringen zweier Wände ausgegangen und daher $2L_{we}$ hinzuaddiert. Anstatt σ^2_{add} geht $(2\sigma_{add})^2$ in die Standardabweichung ein.

Befinden sich beide Antennen in einem Gebäude, kommt folgende Gleichung zur Berechnung des mittleren Ausbreitungsverlustes zur Anwendung:

$$L_{indoor-indoor} = -27,6 + 20\log(d) + 20\log(f) + \text{fix}(\frac{d}{d_{room}}) \cdot L_{wi} + k_f^{\left[\frac{k_f+2}{k_f+1} - b\right]} \cdot L_f \tag{85}$$

Für k_f gilt:

$$k_f = \text{fix}\left(\frac{|h_2 - h_1|}{h_{floor}}\right) \tag{86}$$

Weitere Parameter sind:

h_{floor}	Höhe der Räume (standardmäßig 3 m)
d_{room}	Größe der Räume (standardmäßig 4 m)
b	Empirischer Parameter (standardmäßig 0,46)
L_{wi}	Verlust bei Innenwänden (standardmäßig 5 dB)
L_f	Verlust zwischen angrenzenden Etagen (standardmäßig 18,3 dB)

Die Funktion fix(x) ergibt die größte ganze Zahl, die kleiner oder gleich dem Argument x ist. Aus fix(d/d_{room}) ergibt sich die Anzahl der Innenwände und k_f liefert die Anzahl der Böden zwischen den Etagen, die durchdrungen werden müssen.
Um Einzelheiten der Gebäudearchitektur und andere Unbestimmtheiten nicht zu vernachlässigen, wird auch hier ein normalverteilter Wert hinzuaddiert. Dabei wird die Standardabweichung typischerweise auf 10 dB gesetzt.

9.2.3 Extended Hata - SRD Modell

Bei Funkkomponenten in der Automatisierungstechnik handelt es sich überwiegend um SRDs. Die Antennen solcher SRDs befinden sich sowohl beim Sender als auch beim Empfänger meist auf Höhen von üblicherweise rund 1,5 m. Die festgelegte Mindesthöhe der Basisstationsantenne von 30 m bei dem im vorherigen Abschnitt beschriebenen Extended Hata Modell würde zu überhöhten Ausbreitungsverlusten führen. Um dies zu vermeiden, wird beim Extended Hata - SRD Modell ([42]) die Gleichung für $b(H_b)$ ersetzt:

$$\begin{aligned} b(H_b) &= (1,1\log(f)) - 0,7) \cdot \min\{10, H_b\} \\ -(1,56\log(f) - 0,8) &+ max\left\{0,20\log\left(\frac{H_b}{10}\right)\right\} \end{aligned} \tag{87}$$

Die Antennenhöhen sollten dabei den Bereich zwischen $1,5$ und 3 Metern nicht verlassen.

9.2.4 WINNER II Modell

Ein weiteres Modell ist das WINNER II ([51]), welches verschiedene Szenarien beinhaltet. Gleichung (88) zeigt die grundlegende Formel zur Berechnung von Ausbreitungsverlusten.

$$L_{Winner} = A\log(d[m]) + B + C\log\left(\frac{f_c[GHz]}{5}\right) + X \tag{88}$$

Dabei ist Parameter A ein Ausbreitungsexponent, gefolgt von einem additiven Parameter B. Parameter C beschreibt die Abhängigkeit des Ausbreitungsverlustes P_{Winner} von der Frequenz f_c. In einigen Szenarien, die das WINNER II Modell beinhaltet, wird auch dem Parameter X ein Wert zugewiesen, beispielsweise geht über diesen Parameter bei einer indoor-outdoor-Ausbreitung der Verlust an Wänden in die Berechnung ein. Wie schon in den vorangegangenen Modellen wird eine Streuung der Ausbreitungsverluste mit dem Parameter σ ausgedrückt, wobei diese Variation gemäß der Gaußschen Normalverteilung zu der Gleichung (88) hinzuaddiert wird.
In den einzelnen Szenarien des Modells werden den genannten Parametern feste Werte zugewiesen. Für Szenario B3 - Indoor Hotspot gilt:

Tab. 18: *Parameterwerte für ausgewählte Szenarien*

	A	B	C	σ
LOS	13,9	64,4	20	3
NLOS	37,8	36,5	23	4

9.2.5 Interferenzkriterium

Um auftretende Interferenzen in einem Kanalmodell zu erkennen, muss ein gewisses Entscheidungskriterium festgelegt werden. Dazu soll die Carrier-to-Interference-Ratio als Kenngröße Anwendung finden. Der Wert dieser Kenngröße ist abhängig von der verwendeten Modulation des Nutzsignals und variiert typischerweise zwischen 9 dB (z. B. für QPSK) und mehr als 26 dB (z. B. für 64QAM). Wegen der Anerkennung von Ergebnissen aus der Simulationsumgebung SEAMCAT in Arbeitsgruppen der CEPT, CENELEC und ETSI sowie der möglichen Vergleichbarkeit der Resultate wird der Default-Wert für die CIR direkt aus SEAMCAT übernommen. Zur Interferenzfeststellung wird dann überprüft, ob die Bedingung

$$\frac{C}{I} < 19 \text{ dB} \tag{89}$$

erfüllt ist, wobei C (Carrier bzw. Träger) die Signalstärke des Nutzsignals und I die Interferenzsignalstärke ist.

9.3 Umsetzung

Bisher erfolgte die gegenseitige Beeinflussung der Funkverbindungen über den Platz „Medium“. Dabei nimmt jede Funkverbindung jede andere wahr, wenn diese denselben Frequenzbereich verwendet. Da durch die Reduzierung der Nutzsignalleistung durch beispielsweise Pfadverluste eine räumliche Trennung von Funkverbindungen möglich ist, bildet der bisherige Ansatz die Realität nur unzureichend ab. Aus diesem Grund wird der bislang verwendete „Interferenz-Check“ durch eine Java-Applikation ersetzt, welche die Petri-Netz-Modelle der Funkverbindungen über ein empirisches Kanalmodell verknüpft. Abbildung 92 zeigt die grundlegende Struktur der Anwendung.
Die Java-Applikation verbindet sich über TCP/IP mit den einzeln in „CPN Tools“ geöffneten Petri-Netz-Verbindungsmodellen. Diese wurden um eine Subtransition zur externen Kommunikation erweitert. Während der Laufzeit kann die Anwendung gleichzeitig mit mehreren gleichartigen oder unterschiedlichen Verbindungsmodellen Daten austauschen. Die Java-Applikation besteht aus drei Teilen: der Konfigurationseinheit, dem Synchronisationsteil und dem eigentlichen Kanalmodell.

Der Konfigurationsteil übergibt nicht nur den Verbindungsmodellen die relevanten Parameter, sondern stellt auch dem Kanalmodell alle nötigen Werte für die durchzuführenden Berechnungen bereit. Zum Beispiel sind für das Kanalmodell Antennenparameter (Position, Verstärkung), die Sendefrequenz sowie -leistung und die Ausbreitungsbedingungen zwischen den Antennen notwendig.

Die Synchronisationseinheit (Bestandteil von Simkontroll in Abb. 92) ist dafür verantwortlich, dass die Abarbeitung der einzelnen Verbindungsmodelle in chronologischer Reihenfolge erfolgt. Dieser Teil der Java-Applikation ist notwendig, da die Petri-Netz-Verbindungs-

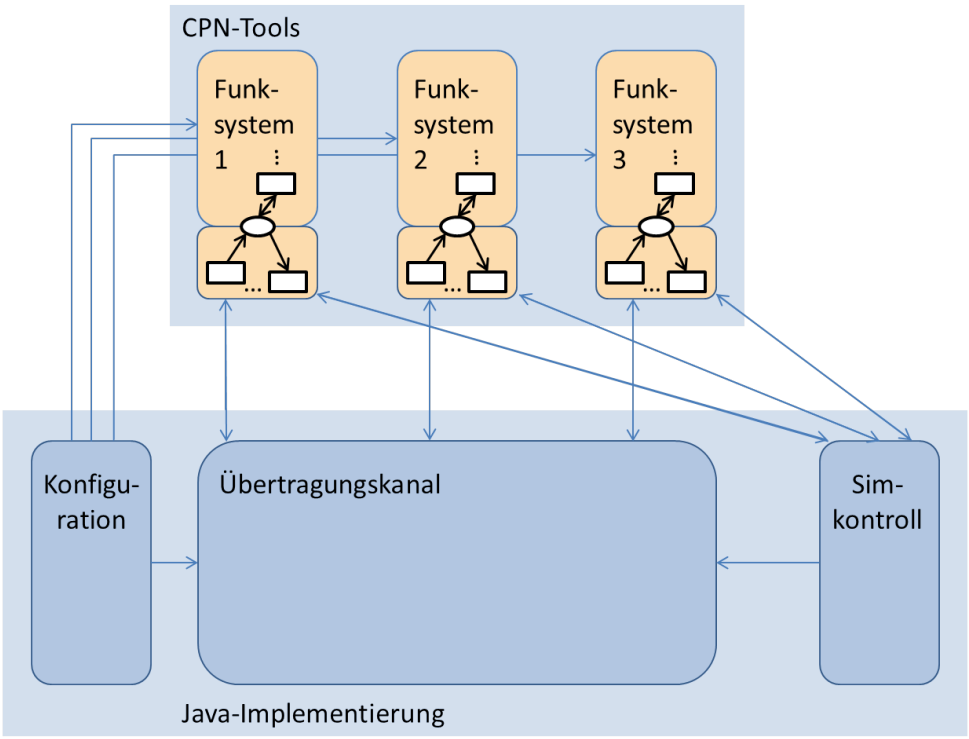

Abb. 92: *Struktur zur Implementierung eines Kanalmodells*

modelle (Funksysteme in Abb. 92) unabhängig voneinander simuliert werden.

Das Kanalmodell bildet das Medium für die Übertragung von Paketen nach. Bei einer gleichzeitigen Übertragung mehrerer Funkverbindungen wird anhand der Empfangsleistung des Nutzsignals und der Leistung des Störsignals überprüft, ob die Nutzsignalübertragung gestört wurde oder nicht. Zur Berechnung der Ausbreitungsverluste auf sämtlichen Wegen zwischen allen positionierten Antennen der verschiedenen Funkverbindungen wurden aus den oben beschriebenen Kanalmodellen die passenden festgelegt. Für Ausbreitungswege, die außschließlich innerhalb einer Industriehalle liegen, wird das WINNER-II-Modell mit den Koeffizienten aus dem Indoor-Hotspot-Szenario angenommen. Für außerhalb der Fabrikhalle liegende Ausbreitungswege, für die außerdem NLOS besteht, sowie für Verbindungen, bei denen mindestens eine Wand druchdrungen wird, ist der Verlust mit dem Extended Hata - SRD Modell zu berechnen. Der Verlust an einer Wand soll dabei $L_{wall} = 10$ dB betragen. Für die übrigen Verbindungsstrecken, die komplett im Freien liegen und LOS angenommen werden kann, soll die Freiraumausbreitung (Free Space Loss Modell) gelten.

Mit Hilfe des Kanalmodells wird ermittelt, wieviel der Leistung des Senders am gewollten Empfänger eintrifft. Es wird aber auch bestimmt, wieviel Störleistung durch den Sendevorgang bei allen anderen Geräten vorliegt. Eine entsprechende Frequenzabhängigkeit wird dabei berücksichtigt. Mit Hilfe des ermittelten Störpegels gegenüber des Nutzsignalpegels kann dann eine Aussage über eine mögliche Interferenz getroffen werden. Am Ende

jeder Übertragung erhalten die Funkverbindungen die Information über möglicherweise aufgetretene Interferenzen. Der Störpegel wird auch bei CCA-Anfragen verwendet, um eine Aussage über die Belegung des Mediums treffen zu können.

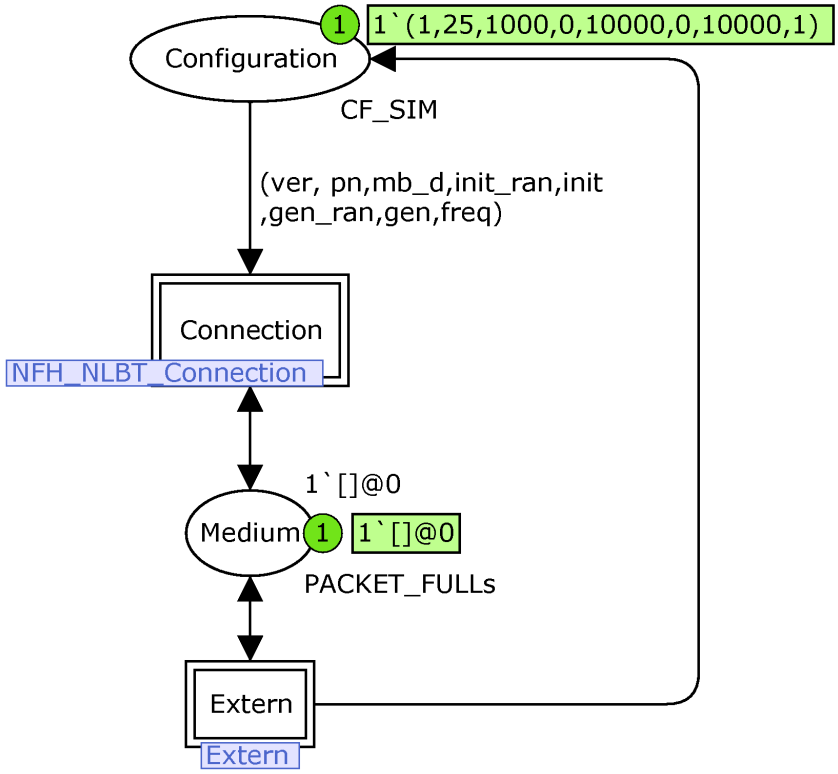

Abb. 93: *Erweitertes Systemmodell*

Abbildung 93 zeigt das angepasste Petri-Netz-Modell, welches nur noch eine Funkverbindung beinhaltet. Die Subtransition „Extern“, welche die externe Kommunikation mit der Java-Applikation realisiert, ist sowohl mit dem Platz „Medium“ als auch mit „Configuration“ verbunden. Zur Konfiguration wird von der Subtransition „Extern“ eine Marke mit den notwendigen Parametern auf den Platz „Configuration“ gelegt. Diese fließt anschließend in das Petri-Netz, welches sich hinter „Connection“ verbindet, und nimmt dort die entsprechenden Einstellungen vor.

Wie bereits erwähnt, übernimmt die Subtransition „Extern“ auch die Synchronisation der einzelnen Petri-Netz-Verbindungsmodelle. Dazu senden die einzelnen Verbindungen, wenn ein Paket zur Übertragung über das Medium ansteht, eine Anfrage mit der aktuellen Zeit des Netzes an die Java-Applikation. Die Java-Anwendung wartet nun so lange, bis alle Verbindungen eine Anfrage gestellt haben. Erst dann bekommt das Netz mit der niedrigsten Zeitangabe die Erlaubnis fortzusetzen und mit der Übertragung zu beginnen. Alle anderen Netze, deren Anfragen abgelehnt wurden, verweilen so lange in einer Schleife, bis sie selbst den niedrigsten Stand der Zeit aufweisen und an der Reihe sind. Die Synchronisation ist auch notwendig, da die Netze zur Umsetzung der unterschiedlichen Medienzugriffsmechanismen unterschiedlich viele Abarbeitungsschritte benötigen.

Tab. 19: *Parameter des Simulationsszenarios zur Untersuchung des Einflusses eines Kanalmodells*

System	t_{onmax} [ms]	n_{Con}	t_{TImin} [ms]	t_{TI} [ms]	Req [%]	Init [ms]	Packets
NFH-NA	5	3	50	80	18,75%	rand(t_{TI})	1.000
NFH-NLBT	5	3	5,05	80	18,75%	rand(t_{TI})	1.000
NFH-FB	5	3	6,25	80	18,75%	rand(t_{TI})	1.000
NFH-LB	5	3	5,33	80	18,75%	rand(t_{TI})	1.000
					75%		

Sie stellt die richtige Reihenfolge für die Benutzung des Mediums sicher.

Die Umsetzung des CCA-Mechanismus erfolgt durch eine Verbindung über einen zusätzlichen Platz zwischen den Subtransitionen „Connection“ und „Extern“, der nicht in Abbildung 93 dargestellt ist. Wenn in einem Verbindungsmodell ein CCA durchgeführt wird, erhält die Java-Applikation eine entsprechende Mitteilung. Sollte das aktuelle Störpotential an dem Teilnehmer, der das CCA vornimmt, so groß sein, dass eine Belegung des Mediums erkannt wird, legt die Subtransition „Extern“ für die Dauer des CCA ein Dummy-Paket auf den Platz „Medium“. Dadurch wird auch von dem Physical Layer des Teilnehmers das Medium als belegt erkannt und die Übertragung entsprechend verzögert.

9.4 Simulation

In einem Simulationsszenario soll untersucht werden, wie sich das implementierte empirische Kanalmodell auf das Zeit- und Fehlerverhalten von Funksystemen auswirkt. Es ist weiterhin zu überprüfen, ob Auswirkungen auf die Koexistenz festzustellen sind. Dazu wird auf das Szenario „Alle Geräte teilen sich das Spektrum gleichmäßig auf“ (siehe Abschnitt 6.5.3) zur Analyse der Anforderungen an die EN 300 328 zurückgegriffen. Bei einer Simulation wird das Medium wie bisher mit einem Platz, der alle Funksysteme miteinander verbindet, nachgebildet. Bei der zweiten Durchführung kommt das implementierte empirische Kanalmodell zum Einsatz. Die relevanten Parameter sind für beide Fälle gleich und in Tabelle 19 aufgelistet.
Aus Performancegründen enthält jedes Funksystem nur drei Verbindungen, wodurch die Gesamtanforderung an das Medium statt 100 % nur noch 75 % beträgt. Auch die Anzahl der Pakete wurde auf 1.000 gesenkt. Die Initialisierung der einzelnen Verbindungen ist zwar zufällig, allerdings aus Gründen der Vergleichbarkeit für beide Fälle identisch. Abbildung 94 zeigt eine willkürliche, räumliche Anordnung der Funkkomponenten, zwischen denen die einzelnen Verbindungen bestehen. Entsprechend dieser Anordnung wird das empirische Kanalmodell parametrisiert. Abbildung 94 stellt zwei Industriehallen dar.

Die Gebäudegrenzen werden durch schwarze Linien markiert. Die verschiedenen Funkverbindungen kommunizieren nicht nur innerhalb und außerhalb der Gebäude, sondern auch zwischen den Gebäuden sowie von innen nach außen.

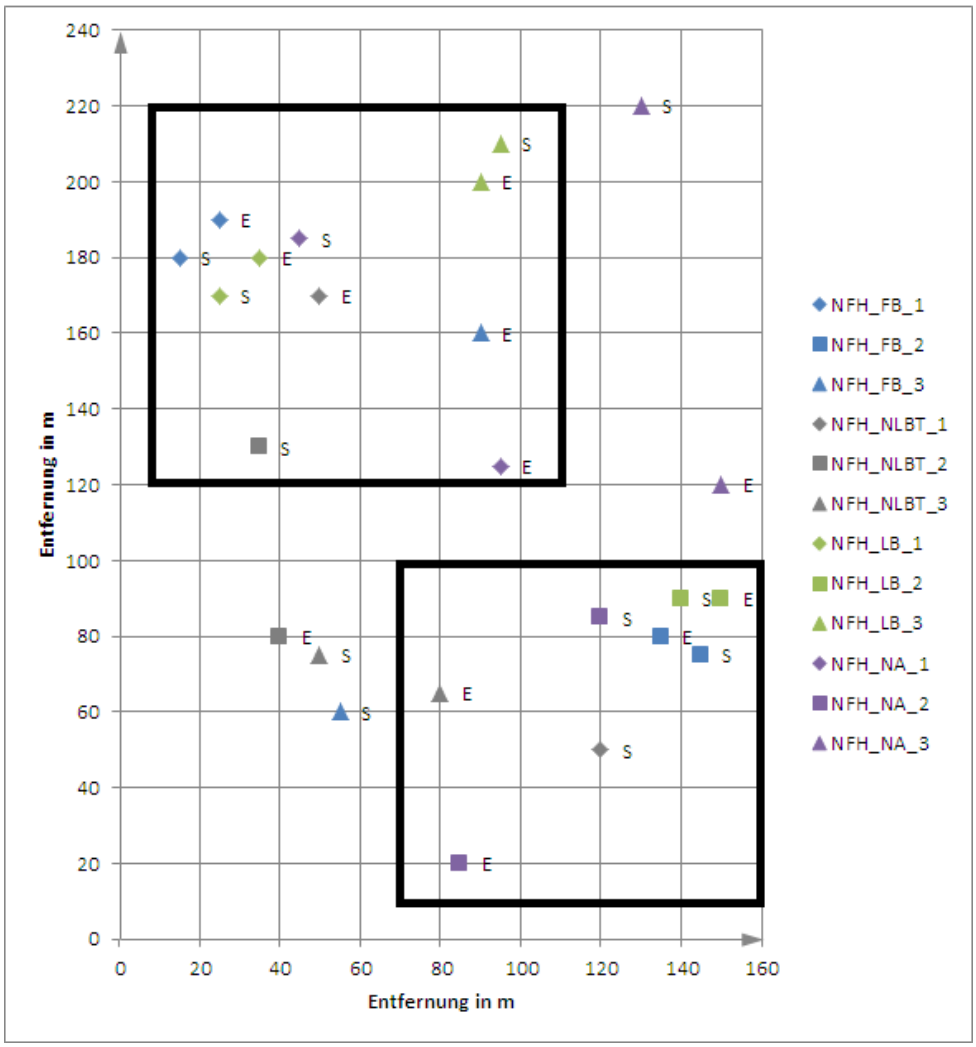

Abb. 94: *Anordnung der Funksysteme*

Abbildung 95 zeigt die Verteilung der Pakete für die beiden beschriebenen Simulationsfälle.

Wie schon in Abschnitt 6 sind grün die korrekt übertragenen Pakete, rot sind die zerstörten und gelb sind die Pakete, die durch einen Speicherüberlauf verworfen wurden. Wieder ist in den Ergebnissen der Trend erkennbar, dass LBT-Systeme gegenüber Nicht-LBT-Systemen bevorteilt sind. Der überwiegende Teil der Pakete des NFH-NA-Systems wird durch Interferenzen zerstört. Die Mehrheit der Pakete des NFH-NLBT-Systems wird durch die Kanalsperrzeit von 1 s verworfen. Beim NFH-FB-System dominieren die korrekt übertragenen Pakete. Allerdings weisen die Verbindungen des NFH-LB-Systems viele Paketkollisionen auf. Eine ausgiebige Analyse der einzelnen Paketkenngrößen hat ergeben, dass die zufällige Initialisierung dazu geführt hat, dass die NFH-NA-Verbindungen genau dann anfangen zu senden, wenn die Übertragung der NFH-LB-Verbindungen gerade aktiv ist.

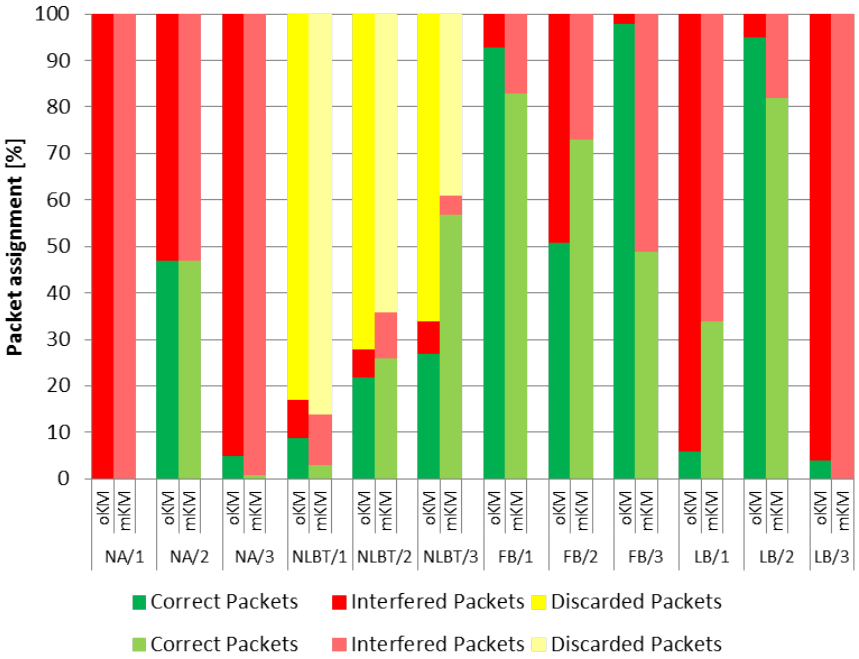

Abb. 95: *Paketverteilung mit (mKM) und ohne (oKM) Verwendung eines Kanalmodells*

Die gewonnenen Ergebnisse zeigen zwei Auswirkungen, die durch die Verwendung eines empirischen Kanalmodells entstehen. Es ist zum einen möglich, dass während des CCA von LBT-Systemen die Störleistung, die durch andere aktive Funkverbindungen erzeugt wird, an dem sendewilligen Gerät unterhalb der für CCA festgelegten Energiepegelgrenze liegt. Daraufhin wird eine Übertragung gestartet. Am Empfänger kann die Störleistung allerdings so groß sein, dass die Demodulation des Paketes unmöglich ist. Eine weitere mögliche Auswirkung ist, dass die Verwendung des empirischen Kanalmodells zu einer räumlichen Trennung von parallelen Funkverbindungen führt. Dadurch können Geräte gleichzeitig kommunizieren, ohne dass sie sich gegenseitig stören.

Bei dem Zeitverhalten in Abbildung 96 ist eher die Tendenz zu erkennen, dass die Übertragungszeiten der Pakete geringer werden. Auch hier spielt die Beeinflussung des CCA-Mechanismus eine entscheidende Rolle. Wenn die Störleistung an der sendewilligen Station zu gering ist, wird das Medium als frei erkannt und die Übertragung wird unverzüglich gestartet.

9.5 Zusammenfassung

In diesem Abschnitt wurde ein Ansatz beschrieben, bei dem die Petri-Netz-Modelle der Funksysteme mit einem empirischen Kanalmodell verknüpft werden. Die Ergebnisse eines damit durchgeführten Simulationsszenarios haben gezeigt, dass die Verwendung eines Kanalmodells einen Einfluss sowohl auf das Fehler- als auch auf das Zeitverhalten hat.

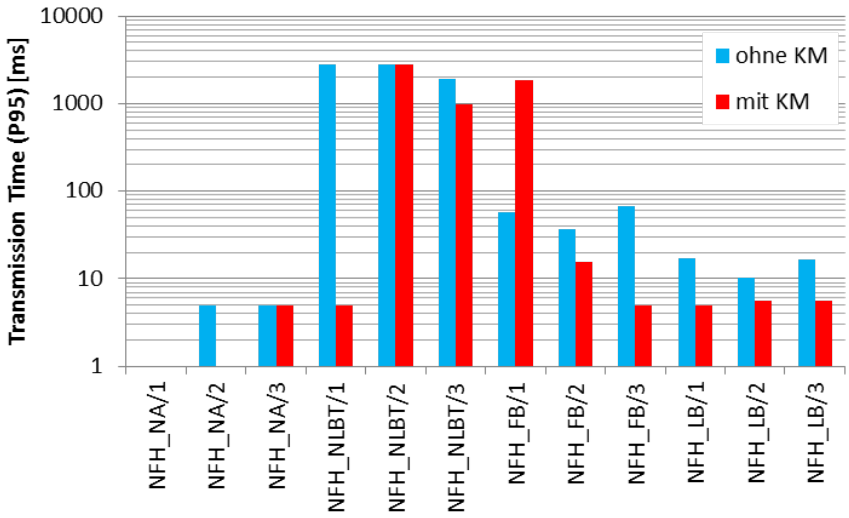

Abb. 96: *Perzentil 95 der Übertragungszeit mit und ohne Verwendung eines Kanalmodells*

Das Zeitverhalten verbessert sich überwiegend. Beim Fehlerverhalten ist kein klarer Trend zu erkennen. Im Einzelnen verbessern oder verschlechtern sich die Verbindungen. Einige bleiben unverändert. Zudem ist keine Abhängigkeit vom verwendeten Medienzugriffsmechanismus festzustellen. Damit belegen die Ergebnisse des Simulationsszenarios, dass die Modellierung des Mediums mit nur einem Platz für die Untersuchung des grundlegenden Koexistenzverhaltens von verschiedenen Medienzugriffsmechanismen ausreichend ist. Die Betrachtung von Ort und Signalleistung ist in diesem Zusammenhang vernachlässigbar. In einem realen Anwendungsfall sollten die Ausbreitungsbedingungen mit einem empirischen Kanalmodell berücksichtigt werden, um eine hohe Genauigkeit der Simulationsergebnisse gewährleisten zu können.

Die Inhalte dieses Kapitels stammen aus [39].

10 Zusammenfassung und Ausblick

10.1 Zusammenfassung

Die Funkkommunikation ist heutzutage ein fester Bestandteil industrieller Automatisierungsanlagen. Die Ursache liegt in den verschiedenen Vorteilen, die bei der Verwendung der Funkkommunikation entstehen. Beispiele sind die Einsparung von oftmals teuren Kabeln und deren Verlegung sowie die Erhöhung der Mobilität und Flexibilität von Sensoren und Aktoren. Je nach Anwendungsfall werden unterschiedliche Anforderungen bezüglich Echtzeitverhalten, Robustheit und Zuverlässigkeit an die Funkkommunikation gestellt. Allerdings werden diese Anforderungen von den verschiedenen Funktechnologien mehr oder weniger gut erfüllt, sodass jede Funktechnologie ihre Daseinsberechtigung besitzt. Der gleichzeitige Einsatz der Funkkommunikation für unterschiedliche Anwendungsfälle führt zu einer immer stärkeren Belegung der knappen Frequenzressource. Resultierend daraus entsteht eine gegenseitige Beeinflussung der auf dasselbe Medium zugreifenden Funklösungen. Diese Beeinflussung wirkt sich negativ auf das Zeit- und Fehlerverhalten aus. Eine mögliche Folge kann sein, dass die Anforderungen der Automatisierungsanwendung nicht mehr erfüllt werden. Ein Ausfall des Produktionsprozesses und die damit verbundenen Kosten wären die mögliche Konsequenz. Es stellt sich die Frage der Koexistenz zwischen den einzelnen Funklösungen und die effiziente Nutzung der kostbaren Spektrumressource, sodass eine Einhaltung der Anwendungsanforderungen gewährleistet werden kann.

Es existieren eine Reihe von Werkzeugen, welche den Fokus auf die Simulation und Untersuchung der Koexistenz von Funklösungen legen. Allerdings betrachten diese Tools jeweils einzelne Aspekte der Koexistenz wie beispielsweise die Interferenzwahrscheinlichkeit oder Signalausbreitung. Es existiert kein Ansatz, welcher Koexistenzuntersuchungen aus Sicht der industriellen Automation ermöglicht. Dazu gehört auch die Ermittlung von Kenngrößen entsprechend der VDI/VDE-Richtlinie 2185, um Rückschlüsse auf die Einhaltung der Anwendungsanforderungen ziehen zu können.
Aus diesem Grund wurde in dieser Arbeit ein Ansatz basierend auf höheren Petri-Netzen entwickelt, um das Zeit- und Fehlerverhalten von Funklösungen bei Beeinflussung durch andere Funksysteme ermitteln zu können. Dabei spielt vor allem der zu Grunde liegende Medienzugriffsmechanismus eine entscheidende Rolle. Höhere Petri-Netze erlauben die formale Modellierung von Funklösungen. Zudem unterstützen sie die ausgiebige Validierung und Verifikation der erstellten Modelle. Mit höheren Petri-Netzen können ereignisdiskrete Systeme und Nebenläufigkeiten modelliert werden. Ebenfalls können parameterbehaftete Ereignisse und Zeitbedingungen nachgebildet werden. Damit erfüllen höhere Petri-Netze sämtliche Anforderungen, die zur Modellierung und Simulation von Koexistenzszenarien in der industriellen Automation notwendig sind.
Der Ansatz der höheren Petri-Netze wurde auf unterschiedliche Medienzugriffsmechanis-

men angewendet:

- Nichtfrequenzhopper, welche in der EN 300 328 spezifiziert sind;
- neue Medienzugriffsmechanismen für die industrielle Automation (MS-Aloha, verbesserter DECT-Ansatz);
- Medienzugriffsmechanismen realer Systeme (WLAN, WSAN-FA).

Koexistenzuntersuchungen an den in der EN 300 328 spezifizierten Medienzugriffsmechanismen haben ergeben, dass das Spektrum nicht effizient genutzt wird. Auch wird das Medium bei Überbelegung von den verschiedenen Medienzugriffsmechanismen unterschiedlich stark genutzt. Damit werden die Anforderungen, welche an die Überarbeitung der EN 300 328 gestellt wurden, nicht erfüllt. Hinzu kommt eine starke Abhängigkeit gegenüber den Anfangsbedingungen und ein unvorhersehbares Zeitverhalten einzelner Zugriffsmechanismen. Aus diesem Grund sind die in der EN 300 328 spezifizierten Verfahren für die industrielle Automation ungeeignet. Das grundlegende Problem ist, dass Koexistenz aus Gerätesicht erreicht werden soll.

Dagegen wird ein Systemansatz beispielsweise von MS-Aloha verwendet. Dieses Zugriffsverfahren stammt aus der Car-to-Car-Kommunikation und nutzt TDMA. Koexistenzuntersuchungen mit höheren Petri-Netzen haben gezeigt, dass MS-Aloha ein robustes Übertragungsverhalten aufweist. Leider verwendet MS-Aloha nur einen Kanal. Dieser wird auch nicht selbstständig gewechselt, wenn beispielsweise ein schmalbandiger Störer denselben Kanal verwendet.
Diesen Mangel beseitigt ein selbstentwickelter Ansatz basierend auf DECT. Der Ansatz verwendet mehrere Frequenzkanäle. Im Falle eines schmalbandigen Störers wird der betroffene Kanal gewechselt. Mit dem nachgewiesenen Zeit- und Fehlerverhalten ist dieser Ansatz ausgezeichnet für die industrielle Automation geeignet. Der Ansatz ist bisher stark idealisiert, sodass für eine mögliche Umsetzung tiefergehende Überlegungen notwendig sind.

Koexistenzuntersuchungen mit höheren Petri-Netzen haben weiterhin gezeigt, dass eine gegenseitige Beeinflussung bei der gleichzeitigen Verwendung von WLAN und WSAN-FA besteht. Beispielsweise können WLAN-Pakete bei der Anwesenheit von WSAN-FA um mehr als 2 ms verzögert werden. Auch WSAN-FA ist durch die Zerstörung von Paketen während der Übertragung betroffen. Allerdings haben die durchgeführten Simulationen auch gezeigt, dass die Kommunikation von WLAN und WSAN-FA zwar beeinträchtigt wird, die Daten dennoch zuverlässig übertragen werden können.

In einem weiteren Abschnitt wurden die aus höheren Petri-Netzen bestehenden Funksystemmodelle mit einem empirischen Kanalmodell verknüpft. Die Simulationsergebnisse

aus einem Vergleich mit und ohne Kanalmodell haben gezeigt, dass die Genauigkeit bei Verwendung eines empirischen Kanalmodells gegenüber einem realen Anwendungsfall erhöht wird. Für die Untersuchung des grundlegenden Koexistenzverhaltens von verschiedenen Medienzugriffsmechanismen ist ein solches jedoch nicht vonnöten.

10.2 Ausblick

In dieser Arbeit wurde versucht, anhand der Kenngröße Übertragungszeit eine vergleichende Bewertung unterschiedlicher Medienzugriffsmechansismen aus Sicht der industriellen Automation vorzunehmen. Allerdings wird das Zeitverhalten der verschiedenen Medienzugriffsmechanismen durch eine Reihe von einstellbaren Parametern bestimmt, sodass ein störungsfreier Vergleich keinen Sinn machen würde. Als zukünftige Forschungsaufgabe ergibt sich allerdings die Parameteroptimierung. Als Folge könnten die einzelnen Medienzugriffsmechanismen für unterschiedliche Automatisierungsanwendungen optimiert werden. Ein anderer Aspekt ist die Bewertung des Koexistenzverhaltens von Medienzugriffsmechanismen bezüglich Spektreneffizienz, Instabilität, gleichmäßige Spektrenaufteilung und unvorhersehbares Zeitverhalten (siehe Abschnitt 6.5). Dies sind qualitative Anforderungen. Quantitative Definitionen dieser Anforderungen, um Aussagen anhand von Messungen und Simulationen gewinnen zu können, existieren bisher noch nicht. Auch ist die systematische, gegenseitige Beeinflussung der verschiedenen Medienzugriffsmechanismen zur Ermittlung des Koexistenverhaltens schwierig. Die Erarbeitung der Anforderungsdefinitionen sowie die vergleichende Bewertung des Koexistenzverhaltens von Medienzugriffsmechanismen bezogen auf die genannten Anforderungen und aus Sicht der industriellen Automation sind Gegenstand des aktuellen Forschungsprojektes „Wireless Regulierung für die industrielle Automation (WiRiA)“ ([107]).

Mit der Simulation von Koexistenzszenarien auf Basis höherer Petri-Netze und aus Sicht der industriellen Automation wurde ein wesentlicher Bestandteil für ein vereinfachtes Koexistenzmanagement erarbeitet. Mit diesem können Automatisierungsingenieure den Zustand der Koexistenz ohne Zuhilfenahme von Funkexperten herstellen. Für die Umsetzung des vereinfachten Koexistenzmanagements sind allerdings noch weitere Schritte erforderlich.
Ein Beispiel für solche weiterführenden Arbeiten ist die Verknüpfung des dynamischen Petri-Netz-Modells mit einem Parametermodell. Dies ist für die Realisierung eines vereinfachten Koexistenzmanagements notwendig, da die eingestellten Parameterwerte einen direkten Einfluss auf das Zeit- und Fehlerverhalten haben ([72]). Zudem ist eine Datenbank basierend auf dem Parametermodell erforderlich, in welcher die geplanten und vorhandenen Funklösungen aufgenommen werden. Diese Datenbank enthält auch die aktuell eingestellten Parameter. Weitere wichtige Informationen sind die Anforderungsgrenzen der einzelnen Automatisierungsanwendungen, welche auf die verschiedenen Funklösungen

zugreifen. Ein anderer wichtiger Aspekt, welcher in der Datenbank aufgeführt sein muss, ist die Position der Funkkomponenten. Anhand der in der Datenbank vorliegenden Daten können nun Koexistenzsimulationen basierend auf Petri-Netz-Modellen durchgeführt werden. Mit diesen wird überprüft, ob die Anwendungsanforderungen unter den gewählten Einstellungen eingehalten werden können. Bei Nichteinhaltung könnte ein Alarm oder eine Warnung ausgelöst werden.
Ein anderer Ansatz, welcher weiterer Betrachtungen bedarf, ist die automatische Optimierung der Geräteparameter in Abhängigkeit von den Anwendungsanforderungen. Industrielle Funkgeräte, welche derzeit am Markt erhältlich sind, bieten eine Vielzahl einstellbarer Parameter. Diese haben nicht nur Einfluss auf das Zeit- und Fehlerverhalten, sondern auch auf die Koexistenz gegenüber anderen Funksystemen. Mit Hilfe des dynamischen Petri-Netz-Modells kann nun eine Art Regelkreis zur Optimierung des Übertragungs- und Koexistenzverhaltens gebildet werden. Das Petri-Netz-Modell stellt dabei die zu regelnde Strecke dar. Die Istwerte sind die Kenngrößen des Zeit- und Fehlerverhaltens. Als Sollwerte dienen die Anwendungsgrenzen. Als Stellgrößen werden die koexistenzrelevanten Geräteparameter verwendet. Mit diesem Ansatz wäre auch ein automatisches Koexistenzmanagement denkbar.
Für die Umsetzung eines automatischen Koexistenzmanagements wäre auch die Anbindung eines genaueren Kanalmodells vonnöten. Auf Basis des Grundrisses der Automatisierungsanlage könnte ein Dominant-Path-Kanalmodell implementiert werden. In dem Grundriss ist die Position der Funkgeräte festzulegen. Aus dem Grundriss und der Position kann dann die an den Empfängern eintreffende Leistung, die in Abhängigkeit vom Sender entweder nutzend oder störend sein kann, errechnet werden. Mit Hilfe eines genaueren Kanalmodells kann auch Einfluss auf die Positionierung der Geräte genommen werden.
Ein funktionierendes Koexistenzmanagement spielt auch während des Betriebs einer Automatisierungsanlage eine wichtige Rolle. Wenn unbekannte Funksysteme wie z. B. das Bluetooth eines Handys in die Anlage gebracht werden, kann die geplante Koexistenz beeinträchtigt werden. Mit aktiven Diagnosesystemen können solche Fremdsysteme aufgespürt und identifiziert werden. Ein Ansatz für ein solches System wurde im Projekt „inWiDia“ entwickelt ([104]). Durch Anbindung dieses Funkdiagnosetools könnten bei Erkennung eines schädigenden Funksystems die Parameter der Nutzdatenfunksysteme angepasst werden, um den Zustand der Koexistenz zu wahren. Sollte dies nicht möglich sein, müsste der Koexistenzmanager automatisch informiert werden. Dieser muss dann das störende Funksystem ausfindig machen und anschließend beseitigen.

Anlage

Programmcodes

Die folgenden Funktionen kommen in den kolorierten Petri-Netz-Modellen zur Anwendung. Innerhalb von CPN Tools kommt dabei die Sprache StandardML zum Einsatz.

Quelltext 1: *Die Funktion „check_first“*

```
fun check_first((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls) = let
    val a =List.length((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls)
    val zwischen = ref true
in
    if (#9 (List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,a-1)))=
       true then zwischen:= !zwischen else zwischen:= false;
    !zwischen
end
| check_first([]) = false;
```

Quelltext 2: *Die Funktion „change_first“*

```
fun change_first((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls) = let
   val a = List.length((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls)
   val seqzw = ref 0 : int ref
   val mbzw = ref 0 : int ref
   val timestzw = ref 0 : int ref
   val verzw = ref 0 : int ref
   val stzw = ref "" : string ref
   val freqzw = ref 0 : int ref
   val ptzw = ref "" : string ref
   val intfzw = ref false : bool ref
   val fzw = ref false : bool ref
   val seqzw2 = ref 0 : int ref
   val mbzw2 = ref 0 : int ref
   val timestzw2 = ref 0 : int ref
   val verzw2 = ref 0 : int ref
   val stzw2 = ref "" : string ref
   val freqzw2 = ref 0 : int ref
   val ptzw2 = ref "" : string ref
   val intfzw2 = ref false : bool ref
   val fzw2 = ref false : bool ref
   val pzw = ref []
   val ifreq = ref 0 : int ref
   val i = ref 0: int ref
   val zwischen = ref false: bool ref
   val position = ref 0: int ref

in
         seqzw := #1(List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
            p_fulls,a-1));
         mbzw := #2(List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls
            ,a-1));
         timestzw := #3(List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
            p_fulls,a-1));
         verzw := #4(List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
            p_fulls,a-1));
         stzw := #5(List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls
            ,a-1));
         freqzw := #6(List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
            p_fulls,a-1));
```

```
34          ifreq:=  !freqzw;
35          ptzw := #7(List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls
                ,a-1));
36          intfzw := #8(List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
                p_fulls,a-1));
37          fzw:=false;
38
39  if a>1 then
40  pzw:= List.take((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,a-1)^^[(!
        seqzw,!mbzw,!timestzw,!verzw,!stzw,!freqzw,!ptzw,!intfzw,!fzw)] else pzw
        :=
41  [(!seqzw,!mbzw,!timestzw,!verzw,!stzw,!freqzw,!ptzw,!intfzw,!fzw)];
42
43  if a>1 then while !i < (a-1) do (
44          if !ifreq = (#6 (List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
                p_fulls,!i))) then (zwischen:= true; position := !i) else
                zwischen:= !zwischen;
45          i:= !i+1
46  ) else zwischen:=false;
47
48  if !zwischen=true then (intfzw:=true;
49   pzw:= List.take((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,a-1)^^[(!
         seqzw,!mbzw,!timestzw,!verzw,!stzw,!freqzw,!ptzw,!intfzw,!fzw)];
50
51  if !position = 0 then
52  (
53          seqzw2 := #1(List.nth(!pzw,!position));
54          mbzw2 := #2(List.nth(!pzw,!position));
55          timestzw2 := #3(List.nth(!pzw,!position));
56          verzw2 := #4(List.nth(!pzw,!position));
57          stzw2 := #5(List.nth(!pzw,!position));
58          freqzw2 := #6(List.nth(!pzw,!position));
59          ptzw2 := #7(List.nth(!pzw,!position));
60          intfzw2 := true;
61          fzw2 := #9(List.nth(!pzw,!position));
62          pzw := List.drop(!pzw,!position+1)^^[(!seqzw2,!mbzw2,!timestzw2,!
                verzw2,!stzw2,!freqzw2,!ptzw2,!intfzw2,!fzw2)]
63  )
64  else
65  (
66          seqzw2 := #1(List.nth(!pzw,!position));
67          mbzw2 := #2(List.nth(!pzw,!position));
68          timestzw2 := #3(List.nth(!pzw,!position));
69          verzw2 := #4(List.nth(!pzw,!position));
70          stzw2 := #5(List.nth(!pzw,!position));
71          freqzw2 := #6(List.nth(!pzw,!position));
72          ptzw2 := #7(List.nth(!pzw,!position));
73          intfzw2 := true;
74          fzw2 := #9(List.nth(!pzw,!position));
75          pzw := List.take(!pzw,!position)^^[(!seqzw2,!mbzw2,!timestzw2,!
                verzw2,!stzw2,!freqzw2,!ptzw2,!intfzw2,!fzw2)]^^List.drop(!pzw
                ,!position+1)
76  )
77
78
79  )
80  else zwischen := true;
81
82  SOME(!pzw)
83  end
```

```
84 | change_first([]) = NONE;
```

Quelltext 3: *Die Initialisierungsfunktion des Monitors*

```
fun init (APPL2'Sammel_mark : PACKET_FIN2 tms) = "Time;Seq;Status;
    Transmission Time;Update Time; Media occupation\n"
```

Quelltext 4: *Die Predicate-Funktion des Monitors*

```
fun pred (bindelem, APPL2'Sammel_mark : PACKET_FIN2 tms) =
let
  fun predBindElem (APPL2'Calculate_Characteristics (1,
                                {mb,mbsum,mbsum2,seq,status,tempt,
                                tempt2,timest,timest2,tt,ut})) = true
      | predBindElem _ = false
in
  predBindElem bindelem
end
```

Quelltext 5: *Die Observer-Funktion des Monitors*

```
fun obs (bindelem, APPL2'Sammel_mark : PACKET_FIN2 tms) =
let
  fun obsBindElem (APPL2'Calculate_Characteristics (1,
                                {mb,mbsum,mbsum2,seq,status,tempt,
                                tempt2,timest,timest2,tt,ut})) =
Int.toString timest2^";"^Int.toString seq^";"^status^";"^Int.toString tt
    ^";"^Int.toString ut^";"^Int.toString mbsum2 ^";\n"
      | obsBindElem _ = ""
in
  obsBindElem bindelem
end
```

Quelltext 6: *Die Stop-Funktion des Monitors*

```
fun stop (APPL2'Sammel_mark : PACKET_FIN2 tms) =
  "Simulationsende:;"^Int.toString (intTime())^";\n"
```

Quelltext 7: *Die Funktion „buffer_oldest“*

```
fun buffer_oldest((seq,mb,timest)::packets, seq2 : int, mb2 : int, timest2
    : int) = let
    val seqzw = ref 0 : int ref
    val mbzw = ref 0 : int ref
    val timestzw = ref 0 : int ref
    val pzw = ref []
in
    seqzw := #1(List.nth((seq,mb,timest)::packets,0));
    mbzw := #2(List.nth((seq,mb,timest)::packets,0));
    timestzw := #3(List.nth((seq,mb,timest)::packets,0));
    pzw := List.drop((seq,mb,timest)::packets,1)^^[(seq2,mb2,timest2)];
SOME(!pzw,!seqzw,!mbzw,!timestzw)
end
| buffer_oldest([],_,_,_) = NONE;
```

Quelltext 8: *Die Funktion „check_token“*

```
fun check_token((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,vern) = let
    val i = ref 0
    val a = List.length((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls)
    val izwischen = ref false
```

```
in
    while !i < a do (
          if vern = (#4 (List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
              p_fulls,!i))) then izwischen:= true else izwischen:= !izwischen;
          i:= !i+1
);
!izwischen
end
| check_token ([],_) = false;
```

Quelltext 9: *Die Funktion „compare_token“*

```
fun compare_token((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,vern) =
    let
    val i = ref 0
    val a = List.length((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls)
    val izwischen = ref 0
in
    while !i < a do (
          if vern = (#4 (List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
              p_fulls,!i))) then izwischen:= !i else izwischen:= !izwischen;
          i:= !i+1
);
SOME (List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,!izwischen),
    List.take((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,!izwischen)
^^List.drop((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,!izwischen+1))
end
| compare_token ([],_) = NONE;
```

Quelltext 10: *Die Funktion „check_same_freq“*

```
fun check_same_freq((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,fre) =
    let
     val i = ref 0
     val a =List.length((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls)
     val zwischen = ref false
in
    while !i < a do (
          if fre = (#6 (List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
              p_fulls,!i))) then zwischen:= true else zwischen:= !zwischen;
          i:= !i+1
);
!zwischen
end
| check_same_freq([],_) = false;
```

Quelltext 11: *Die Funktion „no_same_freq“*

```
fun no_same_freq((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls,fre) = let
     val i = ref 0
     val a =List.length((seq,mb,timest,ver,st,freq,pt,intf,f)::p_fulls)
     val zwischen = ref true
in
    while !i < a do (
          if fre = (#6 (List.nth((seq,mb,timest,ver,st,freq,pt,intf,f)::
              p_fulls,!i))) then zwischen:= false else zwischen:= !zwischen;
          i:= !i+1
);
!zwischen
end
| no_same_freq([],_) = false;
```

Quelltext 12: *Beispiel zur Verifikation der Zeit, in der das Medium uneingeschränkt genutzt werden darf*

```
fun check1 n = Mark.DLL't_seq_started 1 n != empty;
fun check2 n = Mark.DLL'Wait_Gap 1 n != empty;
val name1 = "DLL't_seq_started";
val name2 = "DLL'Wait_Gap";
val testzeit:time =5000;

val listnode1 = PredAllNodes (check1);
val listnode2 = PredAllNodes (check2);
val charname1 = String.explode name1;
val charname2 = String.explode name2;
val explstring = (NodeDescriptor(List.nth(listnode1,0)));
fun testMB (expl,compchar)=
let
val (pref,suff) = Substring.position compchar (Substring.full expl)
val (s,i,n) =  Substring.base suff
val leerbegin = ref i val explode = ref expl
val element = ref 0 val zaehlers = ref 0
val timelist = ref [] val comp = ref compchar
val timelist2 = ref []
in
if i = String.size expl then NONE else SOME i;
while (!zaehlers < 90 ) do (
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if (List.nth(String.explode(!explode),!leerbegin) = #"\n") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if !element = 1 then (timelist := !timelist^^[List.nth(String.explode(!
    explode),!leerbegin)])
else leerbegin := !leerbegin;
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = false
then leerbegin := !leerbegin+1 else leerbegin := !leerbegin;
zaehlers := !zaehlers+1
);String.implode(!timelist)
end;
testMB (explstring,name1);

fun final (listefinal,comparelist) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(listefinal)
val listef = ref listefinal val comp = ref comparelist
val zielend = ref []
in
while (!counter <= listenlaenge-1) do
                (counter := !counter+1;zielend:= !zielend^^[NodeDescriptor(
                   List.nth(!listef,!element))];
ergebnis:= !ergebnis^^[testMB((List.nth(!zielend,!element)),(!comp))];
                element := !element + 1);rev(!ergebnis) end;

fun remove(_, [])     = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;
fun mehrfach []      = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));
fun sort (nil) = nil
|   sort (l) = let
                  val p:time = List.nth(l, length(l) div 2)
```

```
                in
                  sort(List.filter(fn (v) => v < p)(l))^^(p :: sort(List.
                      filter(fn (v) => v > p)(l)))
                end;
fun compare (liste1) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(mehrfach(liste1))
val listeref1 = ref liste1 val zielendtime = ref []
in
while (!counter < listenlaenge) do
(counter := !counter+1;
zielendtime := !zielendtime^^[valOf(fromString(List.nth(mehrfach(!listeref1
    ),!element)))];
element := !element + 1);!zielendtime end;

fun listtimetest (list1,list2,delay1) = let
val lista = ref list1 : time list ref val listb = ref list2 : time list ref
val liststart = ref 0 val listenda = List.length(list1)
val listendb =  List.length(list2) val delay = ref delay1 : time ref
val ausw = ref []
in
while (!liststart < listenda andalso !liststart < listendb) do
(if List.nth(!lista,!liststart) < List.nth(!listb,!liststart) = true
then if List.nth(!listb,!liststart) = List.nth(!lista,!liststart) + !delay
    = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"]
else if List.nth(!lista,!liststart) = List.nth(!listb,!liststart) + !delay
    = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"];
 liststart := !liststart+1);
!ausw
end;
val end1 = final(listnode1,name1);val end2 = final(listnode2,name2);
val end3 = sort(compare(end1));val end4 = sort(compare(end2));
listtimetest(sort(compare(final(listnode2,name2))),sort(compare(final(
    listnode1,name1))),testzeit);
```

Quelltext 13: *Beispiel zur Verifikation der Zeit, in der nicht auf das Medium zugegriffen werden darf*

```
fun check1 n = Mark.DLL'Wait_Gap 1 n != empty;
fun check2 n = Mark.DLL'CalcDC 1 n != empty;
val name1 = "DLL'Wait_Gap";
val name2 = "DLL'CalcDC";
val testzeit:time =5000;

val listnode1 = PredAllNodes (check1);
val listnode2 = PredAllNodes (check2);
val charname1 = String.explode name1;
val charname2 = String.explode name2;
val explstring = (NodeDescriptor(List.nth(listnode1,0)));
fun testMB (expl,compchar)=
let
val (pref,suff) = Substring.position compchar (Substring.full expl)
val (s,i,n) =  Substring.base suff
val leerbegin = ref i val explode = ref expl
val element = ref 0 val zaehlers = ref 0
val timelist = ref [] val comp = ref compchar
val timelist2 = ref []
in
```

```
21 if i = String.size expl then NONE else SOME i;
22 while (!zaehlers < 50 ) do (
23 if (List.nth(String.explode(!explode),!leerbegin) = #"@") = true
24 then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
   element;
25 if (List.nth(String.explode(!explode),!leerbegin) = #"\n") = true
26 then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
   element;
27 if !element = 1 then (timelist := !timelist^^[List.nth(String.explode(!
   explode),!leerbegin)])
28 else leerbegin := !leerbegin;
29 if (List.nth(String.explode(!explode),!leerbegin) = #"@") = false
30 then leerbegin := !leerbegin+1 else leerbegin := !leerbegin;
31 zaehlers := !zaehlers+1
32 );String.implode(!timelist)
33 end;
34 testMB (explstring,name1);
35
36 fun final (listefinal,comparelist) =
37 let
38 val ergebnis = ref [] val counter = ref 0
39 val element = ref 0 val listenlaenge = List.length(listefinal)
40 val listef = ref listefinal val comp = ref comparelist
41 val zielend = ref []
42 in
43 while (!counter <= listenlaenge-1) do
44                (counter := !counter+1;zielend:= !zielend^^[NodeDescriptor(
                    List.nth(!listef,!element))];
45 ergebnis:= !ergebnis^^[testMB((List.nth(!zielend,!element)),(!comp))];
46                 element := !element + 1);rev(!ergebnis) end;
47
48 fun remove(_, [])    = []
49   | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;
50 fun mehrfach []     = []
51   | mehrfach(x::xs) = x::mehrfach(remove(x,xs));
52 fun sort (nil) = nil
53 |   sort (l) = let
54                   val p:time = List.nth(l, length(l) div 2)
55                in
56                   sort(List.filter(fn (v) => v < p)(l))^^(p :: sort(List.
                       filter(fn (v) => v > p)(l)))
57                end;
58 fun compare (liste1) =
59 let
60 val ergebnis = ref [] val counter = ref 0
61 val element = ref 0 val listenlaenge = List.length(mehrfach(liste1))
62 val listeref1 = ref liste1 val zielendtime = ref []
63 in
64 while (!counter < listenlaenge) do
65 (counter := !counter+1;
66 zielendtime := !zielendtime^^[valOf(fromString(List.nth(mehrfach(!listeref1
   ),!element)))];
67 element := !element + 1);!zielendtime end;
68
69 fun listtimetest (list1,list2,delay1) = let
70 val lista = ref list1 : time list ref val listb = ref list2 : time list ref
71 val liststart = ref 0 val listenda = List.length(list1)
72 val listendb =  List.length(list2) val delay = ref delay1 : time ref
73 val ausw = ref []
74 in
```

```
while (!liststart < listenda andalso !liststart < listendb) do
(if List.nth(!lista,!liststart) < List.nth(!listb,!liststart) = true
then if List.nth(!listb,!liststart) = List.nth(!lista,!liststart) + !delay
   = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"]
else if List.nth(!lista,!liststart) = List.nth(!listb,!liststart) + !delay
   = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"];
 liststart := !liststart+1);
!ausw
end;
val end1 = final(listnode1,name1);val end2 = final(listnode2,name2);
val end3 = sort(compare(end1));val end4 = sort(compare(end2));
listtimetest(sort(compare(final(listnode2,name2))),sort(compare(final(
   listnode1,name1))),testzeit);
```

Quelltext 14: *Überprüfung, ob alle Pakete die Frequenz „1“ besitzen*

```
fun checkfrequenz n =((StripTime (Mark.Top'Medium 1 n) != [[]])
andalso
not(#6(List.nth(ms_to_col (StripTime (Mark.Top'Medium 1 n)),0)) = 1));
if PredAllNodes (checkfrequenz) = [] then "selbe Frequenz " else "
   verschiedene Frequenzen";
PredAllNodes(checkfrequenz);
```

Quelltext 15: *Überprüfung, ob während der Sperrzeit das Medium nicht belegt wird*

```
val Platz3 = Mark.PHY'MediumPHY 1;
val Elementeanzahl = 8;
fun checkpacket3 p = Platz p != [];
fun remove(_, [])    = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;

fun mehrfach []     = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));

val Knotenliste3 = PredAllNodes (checkpacket3);
Mark.PHY'MediumPHY 1 999;

fun alleKnoten (refListe,Knot) =
let
val Liste = ref refListe val length = List.length(refListe)
val zaehl = ref 0 val Gesamt = ref []
val Knoten = ref Knot val nr = ref 0 : int ref
in
while (!zaehl < length ) do (
nr := List.nth(!Liste,!zaehl);
Gesamt := !Gesamt^^[testCS(PACKET_FULL.mkstr(ms_to_col(StripTime(!Knoten (!
   nr)))),9)];
zaehl := !zaehl+1
); !Gesamt
end;

val Knotenohnedoppel3 = mehrfach(alleKnoten (Knotenliste3,Platz3));
fun comparepaket (pakel,Knotenohned) =
let
val Listenelement = ref 0 val Listenelement2 = ref 0
val Paketelement = ref 0 val Paketelement2 = ref 0
val truelist = ref [] val truevergleich = ref []
val vergleichsliste = ref[] val Paketelemente = ref pakel
val Knotenod = ref Knotenohned
```

```
in
while (!Paketelement < !Paketelemente) do (
while (!Listenelement2 < List.length(!Knotenod)) do
(
if List.nth(List.nth(!Knotenod,!Listenelement),!Paketelement) =
List.nth(List.nth(!Knotenod,!Listenelement2),!Paketelement2)
then (truelist := !truelist^^["true"] ; Listenelement2 := !Listenelement2
    +1)
else (truelist := !truelist^^["false"] ; Listenelement2 := !Listenelement2
    +1);
truevergleich := !truevergleich^^["true"]
);

if !truelist = !truevergleich
then (vergleichsliste := !vergleichsliste^^[List.nth(List.nth(!Knotenod,!
    Listenelement),!Paketelement)] ;
Paketelement := !Paketelement+1
; Paketelement2 := !Paketelement)
else (vergleichsliste := !vergleichsliste ; Paketelement := !Paketelement+1
; Paketelement2 := !Paketelement);
Listenelement := 0;
Listenelement2 := 0;
truevergleich := [];
truelist := []
);
!vergleichsliste
end;
val erg3 = comparepaket(Elementeanzahl,Knotenohnedoppel3);
val erg4 = comparepaket(Elementeanzahl,Knotenohnedoppel4);

fun gleichzeitig2 n = StripTime((Mark.DLL'Wait_Gap 1 n)) != empty andalso
    StripTime((Mark.PHY'MediumPHY 1 n)) != 1`[]
andalso (List.nth(comparepaket(Elementeanzahl,Knotenohnedoppel1),3)) = "1";
if PredAllNodes(gleichzeitig2) = [] then "nicht gleichzeitig" else "
    gleichzeitig";

PredAllNodes(gleichzeitig2);
```

Quelltext 16: *Beispiel zur Überprüfung eines Zeitraums*

```
fun check1 n = Mark.DLL'Start_Timer 1 n != empty;
fun check2 n = Mark.DLL'One_sec_expired 1 n != empty andalso Mark.DLL'MBDC
    1 n = 1`5000;
fun check3 n = Mark.DLL'Start_Timer 1 n != empty andalso Mark.DLL'MBDC 1 n
    = 1`0;

val name1 = "DLL'Start_Timer";
val name2 = "DLL'One_sec_expired";
val testzeit:time =1000000;

val listnode1 = PredAllNodes (check1);
val listnode2 = PredAllNodes (check2);
val listnode3 = PredAllNodes (check3);
val charname1 = String.explode name1;
val charname2 = String.explode name2;
val explstring = (NodeDescriptor(List.nth(listnode2,0)));
fun testMB (expl,compchar)=
let
val (pref,suff) = Substring.position compchar (Substring.full expl)
val (s,i,n) =  Substring.base suff
val leerbegin = ref i val explode = ref expl
```

```
val element = ref 0 val zaehlers = ref 0
val timelist = ref [] val comp = ref compchar
val timelist2 = ref []
in
if i = String.size expl then NONE else SOME i;
while (!zaehlers < 90 ) do (
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if (List.nth(String.explode(!explode),!leerbegin) = #"\n") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if !element = 1 then (timelist := !timelist^^[List.nth(String.explode(!
    explode),!leerbegin)])
else leerbegin := !leerbegin;
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = false
then leerbegin := !leerbegin+1 else leerbegin := !leerbegin;
zaehlers := !zaehlers+1
);String.implode(!timelist)
end;
testMB (explstring,name2);

fun final (listefinal,comparelist) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(listefinal)
val listef = ref listefinal val comp = ref comparelist
val zielend = ref []
in
while (!counter <= listenlaenge-1) do
                (counter := !counter+1;zielend:= !zielend^^[NodeDescriptor(
                   List.nth(!listef,!element))];
ergebnis:= !ergebnis^^[testMB((List.nth(!zielend,!element)),(!comp))];
                 element := !element + 1);rev(!ergebnis) end;

fun remove(_, [])     = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;
fun mehrfach []      = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));
fun sort (nil) = nil
|   sort (l) = let
                  val p:time = List.nth(l, length(l) div 2)
                in
                  sort(List.filter(fn (v) => v < p)(l))^^(p :: sort(List.
                      filter(fn (v) => v > p)(l)))
                end;
fun compare (liste1) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(mehrfach(liste1))
val listeref1 = ref liste1 val zielendtime = ref []
in
while (!counter < listenlaenge) do
(counter := !counter+1;
zielendtime := !zielendtime^^[valOf(fromString(List.nth(mehrfach(!listeref1
    ),!element)))];
element := !element + 1);!zielendtime end;

fun listtimetest (list1,list2,delay1) = let
val lista = ref list1 : time list ref val listb = ref list2 : time list ref
```

```
val liststart = ref 0 val listenda = List.length(list1)
val listendb =  List.length(list2) val delay = ref delay1 : time ref
val ausw = ref []
in
while (!liststart < listenda andalso !liststart < listendb) do
(if List.nth(!lista,!liststart) < List.nth(!listb,!liststart) = true
then if List.nth(!listb,!liststart) = List.nth(!lista,!liststart) + !delay
   = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"]
else if List.nth(!lista,!liststart) = List.nth(!listb,!liststart) + !delay
   = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"];
 liststart := !liststart+1);
!ausw
end;
val end1 = final(listnode1,name1);val end2 = final(listnode2,name2);
val end3 = sort(compare(end1));val end4 = sort(compare(end2));
listtimetest(sort(compare(final(listnode2,name2))),sort(compare(final(
   listnode1,name1))),testzeit);
```

Quelltext 17: *Überprüfung der Medium Utilization*

```
fun kleinermax n = (Mark.DLL'Max_PN2 1 n) != empty;
fun kleinermax2 n = (Mark.DLL'MBDC 1 n) != empty;
fun checkmedutil () = let
val zaehl = ref 0
val truelist = ref []
val checkutillist = ref []
val vergleich = ref ""
in
while (List.length(!truelist) < List.length(PredAllNodes(kleinermax2)) ) do
    (
while(!zaehl < List.length(PredAllNodes(kleinermax2))) do (
if ms_to_col((Mark.DLL'MBDC 1 (List.nth(PredAllNodes(kleinermax2),!zaehl)))
   ) <= 5000
then checkutillist := !checkutillist^^[true] else checkutillist := !
   checkutillist^^[false];
truelist := !truelist^^[true];
zaehl := !zaehl+1);
if !truelist = !checkutillist = true then vergleich := "Medien Utilization
   immer kleiner 10%"
else vergleich := "Fehlerhafte Medienutilisation");
!vergleich
end;
checkmedutil ();
```

Quelltext 18: *Überprüfung, ob das gleiche Paket vom Medium genommen wird, welches auch darauf gelegt wurde*

```
val Platz = Mark.PHY'PacketoutPHY 1;
val Platz2 = Mark.PHY'PacketinPHY 1;
val Elementeanzahl = 9;
fun checkpacket p = Platz p != [];
fun checkpacket2 q = Platz2 q != [];
fun remove(_, [])    = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;

fun mehrfach []     = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));

fun testCS (expl,anz)=
```

```
let
val leerbegin = ref 0 val explode = ref expl
val element = ref 0 val zaehlers = ref 0
val timelist = ref [] val newexpl = ref ""
val timelist2 = ref [] val test = ref 0
val i = ref 0 val anzahl = ref anz
in
explode := String.implode(remove (#"(" ,remove (#"\"" ,(String.explode(!
    explode)))));
explode := String.implode(remove (#")" ,(String.explode(!explode))));
explode := String.implode(remove (#"_" ,(String.explode(!explode))));
explode := String.implode((String.explode(","^ !explode^",")));
while (!zaehlers < !anzahl(*String.size(!explode)*) ) do (

newexpl := String.extract (!explode, !test, NONE);
leerbegin := 0;
element := 0;
if List.nth(String.explode(!newexpl),!leerbegin) = #","
then (element := !element +1 ; i := !i+1) else element := !element ;
while (!element=1) do (
leerbegin := !leerbegin+1;
if Char.isAlphaNum(List.nth(String.explode(!newexpl),!leerbegin)) = true
then (i := !i+1 ; timelist := !timelist^^[List.nth(String.explode(!newexpl)
    ,!leerbegin)])
else  element := !element+1
);
timelist2 := !timelist2^^[String.implode(!timelist)];
timelist := [];
test := !test+ !i;
zaehlers := !zaehlers+1;
i := 0
); !timelist2 end;

val Knotenliste = PredAllNodes (checkpacket);
val Knotenliste2 = PredAllNodes (checkpacket2);

fun alleKnoten (refListe,Knot) =
let
val Liste = ref refListe val length = List.length(refListe)
val zaehl = ref 0 val Gesamt = ref []
val Knoten = ref Knot val nr = ref 0 : int ref
in
while (!zaehl < length ) do (
nr := List.nth(!Liste,!zaehl);
Gesamt := !Gesamt^^[testCS(PACKET_FULL.mkstr(ms_to_col(StripTime(!Knoten (!
    nr)))),9)];
zaehl := !zaehl+1
); !Gesamt
end;

val Knotenohnedoppel1 = mehrfach(alleKnoten (Knotenliste,Platz));
val Knotenohnedoppel2 = mehrfach(alleKnoten (Knotenliste2,Platz2));
fun comparepaket (pakel,Knotenohned) =
let
val Listenelement = ref 0 val Listenelement2 = ref 0
val Paketelement = ref 0 val Paketelement2 = ref 0
val truelist = ref [] val truevergleich = ref []
val vergleichsliste = ref[] val Paketelemente = ref pakel
val Knotenod = ref Knotenohned
in
```

```
while (!Paketelement < !Paketelemente) do (
while (!Listenelement2 < List.length(!Knotenod)) do
(
if List.nth(List.nth(!Knotenod,!Listenelement),!Paketelement) =
List.nth(List.nth(!Knotenod,!Listenelement2),!Paketelement2)
then (truelist := !truelist^^["true"] ; Listenelement2 := !Listenelement2
    +1)
else (truelist := !truelist^^["false"] ; Listenelement2 := !Listenelement2
    +1);
truevergleich := !truevergleich^^["true"]
);

if !truelist = !truevergleich
then (vergleichsliste := !vergleichsliste^^[List.nth(List.nth(!Knotenod,!
    Listenelement),!Paketelement)] ;
Paketelement := !Paketelement+1
; Paketelement2 := !Paketelement)
else (vergleichsliste := !vergleichsliste ; Paketelement := !Paketelement+1
; Paketelement2 := !Paketelement);
Listenelement := 0;
Listenelement2 := 0;
truevergleich := [];
truelist := []
);
!vergleichsliste
end;
val erg1 = comparepaket(Elementeanzahl,Knotenohnedoppel1);
val erg2 = comparepaket(Elementeanzahl,Knotenohnedoppel2);
rev(List.drop(rev(erg1),1));
if List.length(erg1) = List.length(List.nth(Knotenohnedoppel1,0))-2 = true
andalso List.length(erg2) = List.length(List.nth(Knotenohnedoppel2,0))-2 =
    true
then if rev(List.drop(rev(erg1),1)) = rev(List.drop(rev(erg2),1))
then "Gleiche Pakete" else "Unterschiedliche Pakete" else "Unterschiedliche
     Pakete";
```

Quelltext 19: *Tritt während der Sequenzzeit keine Verzögerung beim Medienzugriff auf?*

```
fun check1 n = Mark.DLL'Txintseq 1 n != empty;
fun check2 n = Mark.PHY'MediumPHY 1 n != empty;
fun check3 n = Mark.DLL'Txintseq 1 n != empty andalso Mark.PHY'MediumPHY 1
    n != empty
andalso StripTime(Mark.PHY'MediumPHY 1 n) != 1`[];
val name1 = "DLL'Txintseq";
val name2 = "PHY'MediumPHY";
val testzeit:time =0;

val listnode1 = PredAllNodes (check1);
val listnode2 = PredAllNodes (check2);
val listnode3 = PredAllNodes(check3);
val charname1 = String.explode name1;
val charname2 = String.explode name2;
val explstring = (NodeDescriptor(List.nth(listnode2,0)));
fun testMB (expl,compchar)=
let
val (pref,suff) = Substring.position compchar (Substring.full expl)
val (s,i,n) =  Substring.base suff
val leerbegin = ref i val explode = ref expl
val element = ref 0 val zaehlers = ref 0
val timelist = ref [] val comp = ref compchar
```

```
val timelist2 = ref []
in
if i = String.size expl then NONE else SOME i;
while (!zaehlers < 90 ) do (
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if (List.nth(String.explode(!explode),!leerbegin) = #"\n") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if !element = 1 then (timelist := !timelist^^[List.nth(String.explode(!
    explode),!leerbegin)])
else leerbegin := !leerbegin;
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = false
then leerbegin := !leerbegin+1 else leerbegin := !leerbegin;
zaehlers := !zaehlers+1
);String.implode(!timelist)
end;
testMB (explstring,name2);

fun final (listefinal,comparelist) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(listefinal)
val listef = ref listefinal val comp = ref comparelist
val zielend = ref []
in
while (!counter <= listenlaenge-1) do
                (counter := !counter+1;zielend:= !zielend^^[NodeDescriptor(
                   List.nth(!listef,!element))];
ergebnis:= !ergebnis^^[testMB((List.nth(!zielend,!element)),(!comp))];
                 element := !element + 1);rev(!ergebnis) end;

fun remove(_, [])     = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;
fun mehrfach []      = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));
fun sort (nil) = nil
|   sort (l) = let
                  val p:time = List.nth(l, length(l) div 2)
                in
                  sort(List.filter(fn (v) => v < p)(l))^^(p :: sort(List.
                      filter(fn (v) => v > p)(l)))
                end;
fun compare (liste1) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(mehrfach(liste1))
val listeref1 = ref liste1 val zielendtime = ref []
in
while (!counter < listenlaenge) do
(counter := !counter+1;
zielendtime := !zielendtime^^[valOf(fromString(List.nth(mehrfach(!listeref1
    ),!element)))];
element := !element + 1);!zielendtime end;

fun listtimetest (list1,list2,delay1) = let
val lista = ref list1 : time list ref val listb = ref list2 : time list ref
val liststart = ref 0 val listenda = List.length(list1)
val listendb =  List.length(list2) val delay = ref delay1 : time ref
```

```
val ausw = ref []
in
while (!liststart < listenda andalso !liststart < listendb) do
(if List.nth(!lista,!liststart) < List.nth(!listb,!liststart) = true
then if List.nth(!listb,!liststart) = List.nth(!lista,!liststart) + !delay
   = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"]
else if List.nth(!lista,!liststart) = List.nth(!listb,!liststart) + !delay
   = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"];
 liststart := !liststart+1);
!ausw
end;
val end1 = final(listnode3,name1);val end2 = final(listnode3,name2);
val end3 = sort(compare(end1));val end4 = sort(compare(end2));
listtimetest(sort(compare(final(listnode3,name2))),sort(compare(final(
   listnode3,name1))),testzeit);
```

Quelltext 20: *Beispiel zur Überprüfung der Abfolge*

```
fun ttx  n = not((Mark.DLL't_seq_expired 1 n) = empty);
fun ttxidle n = not((Mark.DLL'Wait_Gap 1 n) = empty);
val ASK = NOT(POS(FORALL_UNTIL(NF("_", ttx),NF("_", ttxidle))));
if eval_node ASK InitNode = false then "nach tseq folgt tgap" else "Fehler
   ";
```

Quelltext 21: *Überprüfung der Rücksetzung der Medium Utilization*

```
fun checkreset n = Mark.DLL'Start_Timer 1 n != empty andalso Mark.DLL'MBDC
   1 n = 1`0 ;
fun remove(_, [])     = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;
fun mehrfach []      = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));
fun sort (nil) = nil
|   sort (l) = let
                 val p:time= List.nth(l, length(l) div 2)
               in
                 sort(List.filter(fn (v) => v < p)(l))^^(p :: sort(List.
                    filter(fn (v) => v > p)(l)))
               end;
PredAllNodes(checkreset);
fun mbreset () = let
val mbzaehl = ref 0
val mbresetlist = ref []
val mbresetlist2 = ref []

in

while ( !mbzaehl < List.length(PredAllNodes(checkreset))) do (

mbresetlist := !mbresetlist^^Mark.DLL'Start_Timer 1 (List.nth(PredAllNodes(
   checkreset),!mbzaehl));
mbresetlist2 := !mbresetlist2^^Mark.DLL'MBDC 1 (List.nth(PredAllNodes(
   checkreset),!mbzaehl));
mbzaehl := !mbzaehl+1);!mbresetlist end;

fun MBres (expl)=
let

val leerbegin = ref 0 val explode = ref expl
```

```
val element = ref 0 val zaehlers = ref 0
val timelist = ref []
val timelist2 = ref []
val n = ref 0 : time ref
in
while (!zaehlers < List.length(!explode) ) do (
if List.nth(!explode,!leerbegin ) = () @ (!n *1000000) = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
zaehlers := !zaehlers+1;
n:= !n+1
); if !element = List.length(!explode) then true else false
end;
print ("Verstreicht immer eine Sekunde zwischen Nullungen?  ");
MBres (mehrfach(mbreset()));
print ("Zeiten bei denen MBDC genullt wird!  ");
mehrfach(mbreset());
()@100 *(()@100);
```

Quelltext 22: *Überprüfung, ob bei Interferenzen wirklich die „Unavailable Time“ abgewartet wird*

```
fun noInter  n = not((Mark.DLL'Wait_tua 1 n) == empty); (* Bedingung tua
    nicht leer *)
PredAllNodes(noInter); (* Gibt es Knoten der die Bedingung erfuellt? *)

val ASK = NOT(POS(EXIST_UNTIL(TT,NF("_", noInter)))); (* Selbes Vorgehen
    mit ASKCTL (Ein Pfad) *)
val ASK2 = NOT(POS(FORALL_UNTIL(TT,NF("_", noInter)))); (* Selbes Vorgehen
    mit ASKCTL (Alle Pfade) *)
eval_node ASK InitNode;
eval_node ASK2 InitNode;*)
```

Quelltext 23: *Überprüfung der CCA-Zeit*

```
fun check1 n = Mark.DLL'CCA_begin 1 n != empty;
fun check2 n = Mark.DLL'CCA_finished 1 n != empty;
val name1 = "DLL'CCA_begin";
val name2 = "DLL'CCA_finished";
val testzeit:time =20;

val listnode1 = PredAllNodes (check1);
val listnode2 = PredAllNodes (check2);
val charname1 = String.explode name1;
val charname2 = String.explode name2;
val explstring = (NodeDescriptor(List.nth(listnode1,0)));
fun testMB (expl,compchar)=
let
val (pref,suff) = Substring.position compchar (Substring.full expl)
val (s,i,n) =  Substring.base suff
val leerbegin = ref i val explode = ref expl
val element = ref 0 val zaehlers = ref 0
val timelist = ref [] val comp = ref compchar
val timelist2 = ref []
in
if i = String.size expl then NONE else SOME i;
while (!zaehlers < 90 ) do (
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if (List.nth(String.explode(!explode),!leerbegin) = #"\n") = true
```

```
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if !element = 1 then (timelist := !timelist^^[List.nth(String.explode(!
    explode),!leerbegin)])
else leerbegin := !leerbegin;
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = false
then leerbegin := !leerbegin+1 else leerbegin := !leerbegin;
zaehlers := !zaehlers+1
);String.implode(!timelist)
end;
testMB (explstring,name1);

fun final (listefinal,comparelist) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(listefinal)
val listef = ref listefinal val comp = ref comparelist
val zielend = ref []
in
while (!counter <= listenlaenge-1) do
                (counter := !counter+1;zielend:= !zielend^^[NodeDescriptor(
                   List.nth(!listef,!element))];
ergebnis:= !ergebnis^^[testMB((List.nth(!zielend,!element)),(!comp))];
                 element := !element + 1);rev(!ergebnis) end;

fun remove(_, [])    = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;
fun mehrfach []     = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));
fun sort (nil) = nil
|   sort (l) = let
                  val p:time = List.nth(l, length(l) div 2)
                in
                  sort(List.filter(fn (v) => v < p)(l))^^(p :: sort(List.
                      filter(fn (v) => v > p)(l)))
                end;
fun compare (liste1) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(mehrfach(liste1))
val listeref1 = ref liste1 val zielendtime = ref []
in
while (!counter < listenlaenge) do
(counter := !counter+1;
zielendtime := !zielendtime^^[valOf(fromString(List.nth(mehrfach(!listeref1
    ),!element)))];
element := !element + 1);!zielendtime end;

fun listtimetest (list1,list2,delay1) = let
val lista = ref list1 : time list ref val listb = ref list2 : time list ref
val liststart = ref 0 val listenda = List.length(list1)
val listendb =  List.length(list2) val delay = ref delay1 : time ref
val ausw = ref []
in
while (!liststart < listenda andalso !liststart < listendb) do
(if List.nth(!lista,!liststart) < List.nth(!listb,!liststart) = true
then if List.nth(!listb,!liststart) = List.nth(!lista,!liststart) + !delay
    = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"]
```

```
else if List.nth(!lista,!liststart) = List.nth(!listb,!liststart) + !delay
    = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"];
 liststart := !liststart+1);
!ausw
end;
val end1 = final(listnode1,name1);val end2 = final(listnode2,name2);
val end3 = sort(compare(end1));val end4 = sort(compare(end2));
listtimetest(sort(compare(final(listnode2,name2))),sort(compare(final(
    listnode1,name1))),testzeit);
```

Quelltext 24: *Überprüfung, ob bei einem negativen CCA Check der nachfolgende FFP ebenfalls nicht belegt wird*

```
fun skipnext  a = (TI.DLL'Cancel_next_Frame 1 = ArcToTI a) ;
fun denied a = Mark.DLL'Access_allowed 1 a = empty;
fun medbeleg a = Mark.PHY'MediumPHY 1 a != empty;

val askallow = POS(AND(EXIST_NEXT(NF("_",medbeleg)),
AND(MODAL(AF("_",skipnext)),EXIST_NEXT(NF("_",denied)))));
eval_node askallow InitNode
```

Quelltext 25: *Überprüfung, ob bei einem negativen CCA Check der nachfolgende FFP wirklich gesperrt ist*

```
fun skipnext  a = (TI.DLL'Cancel_next_Frame 1 = ArcToTI a) ;
fun denied a = Mark.DLL'Access_allowed 1 a = empty;
fun medbeleg a = Mark.PHY'MediumPHY 1 a != empty;

val askallow = POS(FORALL_UNTIL(TT,(AND(MODAL(AF("_",skipnext)),EXIST_NEXT(
    NF("_",denied))))));
eval_node askallow InitNode
```

Quelltext 26: *Überprüfung der zufälligen ECCA-Zeit*

```
fun check1 n = Mark.DLL'ECCASTART 1 n != empty;
fun check2 n = Mark.DLL'ECCAEND 1 n != empty;

val name1 = "DLL'ECCASTART";
val name2 = "DLL'ECCAEND";
val testzeit:time =4*20;

val listnode1 = PredAllNodes (check1);
val listnode2 = PredAllNodes (check2);

val charname1 = String.explode name1;
val charname2 = String.explode name2;
val explstring = (NodeDescriptor(List.nth(listnode1,0)));
fun testMB (expl,compchar)=
let
val (pref,suff) = Substring.position compchar (Substring.full expl)
val (s,i,n) =  Substring.base suff
val leerbegin = ref i val explode = ref expl
val element = ref 0 val zaehlers = ref 0
val timelist = ref [] val comp = ref compchar
val timelist2 = ref []
in
if i = String.size expl then NONE else SOME i;
while (!zaehlers < 90 ) do (
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = true
```

```
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if (List.nth(String.explode(!explode),!leerbegin) = #"\n") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if !element = 1 then (timelist := !timelist^^[List.nth(String.explode(!
    explode),!leerbegin)])
else leerbegin := !leerbegin;
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = false
then leerbegin := !leerbegin+1 else leerbegin := !leerbegin;
zaehlers := !zaehlers+1
);String.implode(!timelist)
end;
testMB (explstring,name1);
fun final (listefinal,comparelist) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(listefinal)
val listef = ref listefinal val comp = ref comparelist
val zielend = ref []
in
while (!counter <= listenlaenge-1) do
                (counter := !counter+1;zielend:= !zielend^^[NodeDescriptor(
                   List.nth(!listef,!element))];
ergebnis:= !ergebnis^^[testMB((List.nth(!zielend,!element)),(!comp))];
                 element := !element + 1);rev(!ergebnis) end;

fun remove(_, [])     = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;
fun mehrfach []      = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));
fun sort (nil) = nil
|   sort (l) = let
                  val p:time = List.nth(l, length(l) div 2)
                in
                  sort(List.filter(fn (v) => v < p)(l))^^(p :: sort(List.
                      filter(fn (v) => v > p)(l)))
                end;
fun compare (liste1) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(mehrfach(liste1))
val listeref1 = ref liste1 val zielendtime = ref []
in
while (!counter < listenlaenge) do
(counter := !counter+1;
zielendtime := !zielendtime^^[valOf(fromString(List.nth(mehrfach(!listeref1
    ),!element)))];
element := !element + 1);!zielendtime end;
fun listtimetest (list1,list2,delay1) = let
val lista = ref list1 : time list ref val listb = ref list2 : time list ref
val liststart = ref 0 val listenda = List.length(list1)
val listendb =  List.length(list2) val delay = ref delay1 : time ref
val ausw = ref []
in
while (!liststart < listenda andalso !liststart < listendb) do
(if List.nth(!lista,!liststart) < List.nth(!listb,!liststart) = true
then if List.nth(!listb,!liststart) >= List.nth(!lista,!liststart) + !delay
    = true andalso
List.nth(!listb,!liststart) <= List.nth(!lista,!liststart) +4* !delay
```

```
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"]
else if List.nth(!lista,!liststart) >= List.nth(!listb,!liststart) + !delay
    = true andalso
List.nth(!lista,!liststart) <= List.nth(!listb,!liststart) +4* !delay
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"];
 liststart := !liststart+1);
!ausw
end;
val end1 = final(listnode1,name1);val end2 = final(listnode2,name2);
val end3 = sort(compare(end1));val end4 = sort(compare(end2));
fun merge [ ] M = M
| merge L [ ]: time list = L
| merge (L as x::xs) (M as y::ys) =
if x < y
then x :: (merge xs M)
else y :: (merge L ys);

fun split L =
let
val t = (length L) div 2
in
( List.take (L,t) , List.drop (L,t) )
end;

fun sorting [ ] = [ ]
| sorting [x] = [x]
| sorting xs =
let
val (ys,zs) = split xs
in
merge (sorting ys) (sorting zs)
end;

fun vermehren (Liste,mehr) =
let
val Listen = ref Liste :  time list ref
val Listenelement = ref 0
val Listenelement2 = ref 1
val truelist = ref []
val reftime =ref  100 : time ref
val mehrlist = ref mehr : time list ref
in
while (!Listenelement2 < List.length(!Listen)-1) do
(
if List.nth(!Listen,!Listenelement2) - List.nth(!Listen,!Listenelement) < !
    reftime
then mehrlist := !mehrlist^^[List.nth(sorting(!mehrlist),!Listenelement)]
    else mehrlist := !mehrlist;
Listenelement := !Listenelement+1;
Listenelement2 := !Listenelement2+1

);
!mehrlist
end;
val mehrliste = vermehren(end4,end3);
end3;
sorting(mehrliste);
end4;
listtimetest(end4,sorting(mehrliste),20);
```

Quelltext 27: *Beispiel zur Überprüfung, ob bei einer erkannten Belegung nicht auf das Medium zugegriffen wird*

```
val Platz3 = Mark.PHY'MediumPHY 1;
val Elementeanzahl = 9;
fun checkpacket3 p = List.nth(StripTime(Platz3 p),0) != [] andalso List.
    length(StripTime(Platz3 p)) >= 1
andalso StripTime((Mark.DLL'CCA_finished 1 p)) != empty ;
fun remove(_, [])     = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;

fun mehrfach []      = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));

val Knotenliste3 = PredAllNodes (checkpacket3);
Mark.PHY'MediumPHY 1 1;
fun testCS (expl,anz)=
let
val leerbegin = ref 0 val explode = ref expl
val element = ref 0 val zaehlers = ref 0
val timelist = ref [] val newexpl = ref ""
val timelist2 = ref [] val test = ref 0
val i = ref 0 val anzahl = ref anz
in
explode := String.implode(remove (#"(" ,remove (#"\"" ,(String.explode(!
    explode)))));
explode := String.implode(remove (#")" ,(String.explode(!explode))));
explode := String.implode(remove (#"_" ,(String.explode(!explode))));
explode := String.implode((String.explode(","^ !explode^",")));
while (!zaehlers < !anzahl(*String.size(!explode)*) ) do (

newexpl := String.extract (!explode, !test, NONE);
leerbegin := 0;
element := 0;
if List.nth(String.explode(!newexpl),!leerbegin) = #","
then (element := !element +1 ; i := !i+1) else element := !element ;
while (!element=1) do (
leerbegin := !leerbegin+1;
if Char.isAlphaNum(List.nth(String.explode(!newexpl),!leerbegin)) = true
then (i := !i+1 ; timelist := !timelist^^[List.nth(String.explode(!newexpl)
    ,!leerbegin)])
else  element := !element+1
);
timelist2 := !timelist2^^[String.implode(!timelist)];
timelist := [];
test := !test+ !i;
zaehlers := !zaehlers+1;
i := 0
); !timelist2 end;

fun alleKnoten (refListe,Knot) =
let
val Liste = ref refListe val length = List.length(refListe)
val zaehl = ref 0 val Gesamt = ref []
val Knoten = ref Knot val nr = ref 0 : int ref
in
while (!zaehl < length ) do (
nr := List.nth(!Liste,!zaehl);
Gesamt := !Gesamt^^[testCS(PACKET_FULL.mkstr(ms_to_col(List.nth(StripTime(!
    Knoten (!nr)),0))),9)];
zaehl := !zaehl+1
```

```
); !Gesamt
end;
alleKnoten (Knotenliste3,Platz3);
val Knotenohnedoppel3 = mehrfach(alleKnoten (Knotenliste3,Platz3));

fun comparepaket (pakel,Knotenohned) =
let
val Listenelement = ref 0 val Listenelement2 = ref 0
val Paketelement = ref 0 val Paketelement2 = ref 0
val truelist = ref [] val truevergleich = ref []
val vergleichsliste = ref[] val Paketelemente = ref pakel
val Knotenod = ref Knotenohned
in
while (!Paketelement < !Paketelemente) do (
while (!Listenelement2 < List.length(!Knotenod)) do
(
if List.nth(List.nth(!Knotenod,!Listenelement),!Paketelement) =
List.nth(List.nth(!Knotenod,!Listenelement2),!Paketelement2)
then (truelist := !truelist^^["true"] ; Listenelement2 := !Listenelement2
    +1)
else (truelist := !truelist^^["false"] ; Listenelement2 := !Listenelement2
    +1);
truevergleich := !truevergleich^^["true"]
);

if !truelist = !truevergleich
then (vergleichsliste := !vergleichsliste^^[List.nth(List.nth(!Knotenod,!
    Listenelement),!Paketelement)] ;
Paketelement := !Paketelement+1
; Paketelement2 := !Paketelement)
else (vergleichsliste := !vergleichsliste ; Paketelement := !Paketelement+1
; Paketelement2 := !Paketelement);
Listenelement := 0;
Listenelement2 := 0;
truevergleich := [];
truelist := []
);
!vergleichsliste
end;
val erg3 = comparepaket(Elementeanzahl,Knotenohnedoppel3);
if ((List.nth(erg3,5) = "1" andalso List.nth(erg3,4) = "NFHLBTLB") orelse
(List.nth(erg3,3) = "1" andalso List.nth(erg3,4) = "NFHLBTLB")) then "Error
    " else "Leer"
```

Quelltext 28: *Beispiel zur Überprüfung, ob innerhalb der Kanalbelegungszeit nur CCA ausgeführt wird*

```
fun check1 n = Mark.DLL'TTx_on 1 n != empty andalso StripTime(Mark.DLL'
    TTx_on 1 n) = 1`[()];
fun check2 n = Mark.DLL'CCA_required 1 n != empty;
fun check3 n = Mark.DLL'CCA_finished 1 n != empty;
fun check4 n = Mark.DLL'CCA_finished 1 n != empty;
val name1 = "DLL'TTx_on";
val name2 = "DLL'CCA_required";
val name3 = "DLL'CCA_finished";
val name4 = "DLL'CCA_finished";
val testzeit:time =0;

val listnode1 = PredAllNodes (check1);
val listnode2 = PredAllNodes (check2);
val listnode3 = PredAllNodes (check3);
```

```
val listnode4 = PredAllNodes (check4);
val charname1 = String.explode name1;
val charname2 = String.explode name2;
val explstring = (NodeDescriptor(List.nth(listnode1,0)));
fun testMB (expl,compchar)=
let
val (pref,suff) = Substring.position compchar (Substring.full expl)
val (s,i,n) =  Substring.base suff
val leerbegin = ref i val explode = ref expl
val element = ref 0 val zaehlers = ref 0
val timelist = ref [] val comp = ref compchar
val timelist2 = ref []
in
if i = String.size expl then NONE else SOME i;
while (!zaehlers < 90 ) do (
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if (List.nth(String.explode(!explode),!leerbegin) = #"\n") = true
then (element := !element+1 ; leerbegin := !leerbegin +1) else element := !
    element;
if !element = 1 then (timelist := !timelist^^[List.nth(String.explode(!
    explode),!leerbegin)])
else leerbegin := !leerbegin;
if (List.nth(String.explode(!explode),!leerbegin) = #"@") = false
then leerbegin := !leerbegin+1 else leerbegin := !leerbegin;
zaehlers := !zaehlers+1
);String.implode(!timelist)
end;
testMB (explstring,name1);

fun final (listefinal,comparelist) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(listefinal)
val listef = ref listefinal val comp = ref comparelist
val zielend = ref []
in
while (!counter <= listenlaenge-1) do
                (counter := !counter+1;zielend:= !zielend^^[NodeDescriptor(
                   List.nth(!listef,!element))];
ergebnis:= !ergebnis^^[testMB((List.nth(!zielend,!element)),(!comp))];
                 element := !element + 1);rev(!ergebnis) end;

fun remove(_, [])    = []
  | remove(x, y::ys) = if x=y then remove(x, ys) else y::remove(x,ys) ;
fun mehrfach []     = []
  | mehrfach(x::xs) = x::mehrfach(remove(x,xs));
fun sort (nil) = nil
|   sort (l) = let
                 val p:time = List.nth(l, length(l) div 2)
               in
                 sort(List.filter(fn (v) => v < p)(l))^^(p :: sort(List.
                     filter(fn (v) => v > p)(l)))
               end;
fun compare (liste1) =
let
val ergebnis = ref [] val counter = ref 0
val element = ref 0 val listenlaenge = List.length(mehrfach(liste1))
val listeref1 = ref liste1 val zielendtime = ref []
```

```
in
while (!counter < listenlaenge) do
(counter := !counter+1;
zielendtime := !zielendtime^^[valOf(fromString(List.nth(mehrfach(!listeref1
    ),!element)))];
element := !element + 1);!zielendtime end;

fun listtimetest (list1,list2,delay1) = let
val lista = ref list1 : time list ref val listb = ref list2 : time list ref
val liststart = ref 0 val listenda = List.length(list1)
val listendb =  List.length(list2) val delay = ref delay1 : time ref
val ausw = ref []
in
while (!liststart < listenda andalso !liststart < listendb) do
(if List.nth(!lista,!liststart) < List.nth(!listb,!liststart) = true
then if List.nth(!listb,!liststart) = List.nth(!lista,!liststart) + !delay
    = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"]
else if List.nth(!lista,!liststart) = List.nth(!listb,!liststart) + !delay
    = true
then ausw := !ausw^^["korrekt"] else ausw := !ausw^^["falsch"];
 liststart := !liststart+1);
!ausw
end;
val end1 = final(listnode1,name1);
val crowdlist = final(listnode2,name2)^^final(listnode3,name3)^^final(
    listnode4,name4);
val end3 = sort(compare(end1));val end4 = List.drop(sort(compare(crowdlist)
    ),1);
listtimetest(end4,sort(compare(final(listnode1,name1))),testzeit);
```

Quellen- und Literaturverzeichnis

[1] ISA 100.11a, *Wireless Systems for Industrial Automation: Process Control and Related Applications*, 2011.

[2] ISO/IEC 15909-1, *Systems and software engineering – High-level Petri nets – Part 1: Concepts, definitions and graphical notation*, 2004.

[3] IEEE Std. 1900.2, *IEEE Recommended Practice for the Analysis of In-Band and Adjacent Band Interference and Coexistence Between Radio Systems*, 2008.

[4] VDI/VDE Richtlinie 2185, *Funkgestützte Kommunikation in der Automatisierungstechnik*, 2. Auflage, Juni 2006.

[5] VDI/VDE Richtlinie 4001-2, *Terminologie der Zuverlässigkeit*, Juli 2006.

[6] IEC 62591, *Industrial communication networks – Wireless communication network and communication profiles – WirelessHART*, 2010.

[7] IEC 62601, *Industrial communication networks – Fieldbus specifications – WIA-PA communication network and communication profile*, 2011.

[8] IEC/TS 62657-2/Ed1, *Industrial Communication Networks – Wireless Communication Network Part 2: Coexistence Management*, 2011.

[9] IEEE Std. 802.11, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 2007.

[10] IEEE Std. 802.15.1, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)*, 2005.

[11] IEEE Std. 802.15.2, *IEEE Recommended Practice for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 15.2: Coexistence of Wireless Personal Area Networks With other Wireless Devices Operating in Unlicensed Frequency Bands*, 2003.

[12] IEEE Std. 802.15.4, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, 2006.

[13] IEEE Std. 802.15.4, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, 2003.

[14] IEEE Std. 802.15.4a, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), Amendment 1: Add Alternate PHYs*, 2007.

[15] IEEE Std. 802.16.2, *IEEE Recommended Practice for Local and Metropolitan Area Networks Coexistence of Fixed Broadband Wireless Access Systems*, 2001.

[16] IEEE Std. 802.16.2, *IEEE Recommended Practice for Local and Metropolitan Area Networks Coexistence of Fixed Broadband Wireless Access Systems*, 2004.

[17] K. Ahmed u. a., „A Cognitive Radio Approach to Realize Coexistence Optimized Wireless Automation Systems“, in: *IEEE International Conference on Emerging Technologies and Factory Automation*, Mallorca, Spain, Sep. 2009.

[18] M. Ajmone Marsan u. a., „On Petri Net-based Modeling Paradigms for the Performance Analysis of Wireless Internet Accesses“, in: *9th International Workshop on Petri Nets and Performance Models*, RWTH Aachen, Germany, Sep. 2001.

[19] L. Angrisani u. a., „Experimental study of coexistence issues between IEEE 802.11b and IEEE 802.15.4 wireless networks“, in: *IEEE Trans. on Instrumentation and Measurement*, Bd. 57, Aug. 2008, S. 1514–1523.

[20] L.L. Bello und E. Toscano, „Coexistence issues of multiple co-located IEEE 802.15.4/ ZigBee networks running on adjacent radio channels in industrial environments“, in: *IEEE Transactions on Industrial Informatics, Special Section on Communication in Automation*, Bd. 5, 2, Mai 2009.

[21] L.L. Bello und E. Toscano, „Cross-Channel Interference in IEEE 802.15.4 networks“, in: *IEEE International Workshop on Factory Communication Systems (WFCS)*, Dresden, Germany, 2008, S. 139–148.

[22] T. Benkner, *Grundlagen des Mobilfunks*, J. Schlembach Fachverlag, 2007.

[23] M. Bertocco, G. Gamba und A. Sona, „Is CSMA/CA really efficient against interference in a wireless control system? An experimental answer“, in: *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2008.

[24] N. Bisnik und A.A. Abouzeid, „Queuing network models for delay analysis of multihop wireless ad hoc networks“, in: *Elsevier Ad Hoc Networks J.* 7.1 (Jan. 2009), S. 79–97.

[25] D. Block und U. Meier, „Wireless Deterministic Medium Access: A Novel Concept Using Cognitive Radio“, in: *Conference on Advances in Cognitive Radio (COCORA)*, Venice, Italy, Apr. 2013.

[26] AWE Communications, `http://www.awe-communications.com`.

[27] L. Drozd, „Computational electromagnetics applied to analyzing the efficient utilization of the RF transmission hyperspace“, in: *Proc. IEEE/ACES Int.* Honolulu, Hawaii, USA, Apr. 2005.

[28] J.-P. Ebert und A. Willig, *A Gilbert-Elliot Bit Error Model and the Efficient Use in Packet Level Simulation*, TU Berlin, März 1999.

[29] EnOcean, `http://www.enocean.com`.

[30] S. Feldmann, T. Hartmann und K. Kyamakya, *Modeling and Evaluation of Scatternets Performance by using Petri Nets*, Institute of Communications Engineering, Hanover, Germany.

[31] A. Gnad, L. Rauchhaupt und L. Gollub, „Multi-Functional Interface for Test of Industrial Wireless Solutions“, in: *Embedded World*, 2008.

[32] N. Golmie, *Coexistence in Wireless Networks – Challenges and System-Level Solutions in the Unlicensed Bands*, Cambridge University Press, 2006.

[33] N. Golmie, R.E. Van Dyck und A. Soltanian, „Interference of Bluetooth and IEEE 802.11: Simulation Modeling and Performance Evaluation“, in: *Proceeding of the 4th ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Rome, Italy, Juli 2001, S. 11–18.

[34] S. Gordon und J. Billington, „Analysing the wap class 2 wireless transaction protocol using colored petri nets“, in: *M. Nielsen and D. Simpson, editiors, ICATPN*, Bd. 1825, Lecture Notes in Computer Science, Springer-Verlag, 2000, S. 207–226.

[35] Wireless Communication Technologies Group, `http://www.antd.nist.gov/wctg/bluetooth/btint.html`.

[36] F. Gustrau, *Hochfrequenztechnik - Grundlagen der mobilen Kommunikation*, Carl Hanser Verlag, 2011.

[37] G. Haßlinger und O. Hohlfeld, *The Gilbert-Elliott Model for Packet Loss in Real Time Services on the Internet*, University of Technology Darmstadt.

[38] A. Heindl und R. German, *Performance Modeling of IEEE 802.11 Wireless LANs with Stochastic Petri Nets*, TU Berlin, Germany, 2000.

[39] H. Heinemann, „Umsetzung eines Kanalmodells für Petri-Netz modellierte Funksysteme“, Bachelorarbeit, Otto-von-Guericke-Universität Magdeburg, 2013.

[40] A. Henley und S. Bond, *Radiocommunications Agency Radio Technology & Compatibility Group - Building Shielding Loss at 5GHz. Amended Report, RTCG Project no. 424*, 1997.

[41] I. Howitt und J.A. Gutierrez, „IEEE 802.15.4 Low Rate-Wireless Personal Area Network Coexistence Issues“, in: *IEEE Wireless Communications and Networking (WCNC)*, Bd. 3, März 2003, S. 1481–1486.

[42] ITU-R, *Report SM.2028-1: Monte Carlo simulation methodology for the use in sharing and compatibility studies between different radio services or systems*, 2002.

[43] Texas Instrument Inc., *CC2420 Coexistence (AN041)*, Application Note, Juni 2006.

[44] European Telecommunications Standards Institute, `http://www.etsi.org`.

[45] R. Jayaparvathy u. a., „Performance Analysis of IEEE 802.11 DCF with Stochastic Reward Nets“, in: *International Journal of Communication Systems*, Bd. 20, 3, 2007.

[46] K. Jensen und L.M. Kristensen, *Coloured Petri Nets: Modelling and Validation of Concurrent Systems*, Springer, 2009.

[47] R. Khalili und K. Salamatian, „Evaluation of Packet Error Rate in Wireless Networks“, in: *MSWIM*, Venice, Italy, 2004.

[48] J. Kiefer und E. Schnieder, „Petri net modelling of fieldbus systems“, in: *16th International Conference on Application and Theory of Petri Nets – Petri Nets applied to Protocols*, Torino, Italy, 1995, S. 97–109.

[49] W. König, H. Rommelfanger und D. Ohse, *Taschenbuch der Wirtschaftsinformatik und Wirtschaftsmathematik*, Harri Deutsch Verlag, 2003.

[50] S. Koker, „Modellierung des Koexistenzverhaltens von WSAN-FA mit Petri-Netzen“, Bachelorarbeit, Otto-von-Guericke-Universität Magdeburg, 2013.

[51] P. Kyösti u. a., *WINNER II Channel Models*, WINNER, 2007.

[52] K. Lemmer und E. Schnieder, „Diagnosis of automation systems on the base of time-related Petri nets“, in: *12th IFAC World Congress*, Bd. 9, Sydney, Australia, 1993, S. 217–220.

[53] Jennic Ltd., *Co-existence of IEEE 802.15.4 at 2.4 GHz (JN-AN-1079)*, Application Note, Feb. 2008.

[54] J. Lunze, *Ereignisdiskrete Systeme: Modellierung und Analyse dynamischer Systeme mit Automaten, Markovketten und Petrinetzen*, Oldenbourg, 2009.

[55] Y. Ma, J. Han und K. Trivedi, „Channel allocation with recovery strategy in wireless networks“, in: *European Transactions on Telecommunications (ETT)*, Bd. 11, 4, 2000, S. 395–406.

[56] Y. Ma, J. Han und K. Trivedi, „Composite performance & availability analysis of wireless communication networks“, in: *IEEE Transactions on Vehicular Technology*, Bd. 50, 5, Sep. 2001, S. 1216–1223.

[57] R. Matheson, „The electrospace model as a frequency management tool“, in: *Int. Symposium On Advanced Radio Technologies*, Boulder, Colorado, USA, März 2003.

[58] A. Mathew u. a., „IEEE 802.11 & Bluetooth interference: simulation and coexistence“, in: *7th Annual Conference on Communication Networks and Services Research*, 2009, S. 217–223.

[59] M. Meyer, *Kommunikationstechnik - Konzepte der modernen Nachrichtenübertragung*, Vieweg+Teubner Verlag, 2012.

[60] D. Miorandi, A.A. Kherani und E. Altman, „A queueing model for HTTP traffic over IEEE 802.11 WLANs“, in: *Elsevier Computer Networks J.* 50.1 (Jan. 2006), S. 63–79.

[61] A. Müller, *CLDPS - Verbindungslose Datenübertragung über DECT*, White Paper, Höft & Wessel.

[62] R. Morrow, *Wireless network coexistence*, McGraw-Hill, 2004.

[63] T. Murata, „Petri Nets: Properties, Analysis and Applications“, in: *Proceedings of the IEEE*, Bd. 77, 4, 1989, S. 541–580.

[64] M. Ozdemir und A.B. McDonald, „An M/MMGI/1/K queuing model for IEEE 802.11 ad hoc networks“, in: *Proceedings of the 1st ACM International Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, ACM Press, 2004, S. 107–111.

[65] M. Petrova u. a., „Performance Study of IEEE 802.15.4 Using Measurements and Simulations“, in: *IEEE Wireless Communications and Networking (WCNC)*, Las Vegas, USA, Apr. 2006.

[66] M. Radetzki u. a., „Robustheit nanoelektronischer Schaltungen und Systeme“, in: *4. GMM/GI/ITG-Fachtagung*, Wildbad Kreuth, Germany, Sep. 2010.

[67] L. Rauchhaupt, E. Hintze und A. Gnad, „Über die Bewertung der Zuverlässigkeit industrieller Funklösungen - Die praktische Umsetzung“, in: *atp* 4 (2007), S. 50–57.

[68] L. Rauchhaupt, E. Hintze und A. Gnad, „Über die Bewertung der Zuverlässigkeit industrieller Funklösungen - Die theoretischen Grundlagen“, in: *atp* 3 (2007), S. 38–47.

[69] S. Ray, D. Starobinski und J.B. Carruthers, „Performance of wireless networks with hidden nodes: A queuing-theoretic analysis“, in: *Computer Communications* 28.10 (2005), S. 1179–1192.

[70] SEAMCAT - Spectrum Engineering Advanced Monte Carlo Analysis Tool, `http://www.ero.dk/seamcat`.

[71] A. Schimschar und M. Krätzig, „Assessment of LBT Systems using Petri Net modelling“, in: *1st IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control (CESCIT*, Würzburg, Germany, Apr. 2012.

[72] A. Schimschar, M. Krätzig und M. Wollschlaeger, „Koexistenzsimulation von drahtlosen Systemen als Bestandteil des Engineerings“, in: *at - Automatisierungstechnik* (2013).

[73] A. Schimschar und L. Rauchhaupt, „Impact of different Bit Rates on Performance Characteristics of Industrial WLAN Solutions“, in: *7th International Conference on Informatics in Control, Automation and Robotics (IFAC)*, Funchal, Madeira, Juni 2010.

[74] A. Schimschar u. a., „Modellierung von Funkkomponenten im Kontext des Lebenszyklus der industriellen Automation“, in: *Automation 2013*, Baden-Baden, Germany, Juni 2013.

[75] S.Y. Shin, J.S. Kang und H.S. Park, „Packet Error Rate Analysis of ZigBee under Interferences of Multiple Bluetooth Piconets“, in: *Vehicular Technologie Conference*, Barcelona, Spain, Apr. 2009.

[76] S.Y. Shin u. a., „Packet Error Rate Analysis of IEEE 802.15.4 under IEEE 802.11b Interference“, in: *Vehicular Technologie Conference*, Bd. 3, Melbourne, Australia, Mai 2006, S. 1186–1190.

[77] S.Y. Shin u. a., „Packet Error Rate Analysis of ZigBee Under WLAN and Bluetooth Interferences“, in: *IEEE Transactions on Wireless Communications*, Bd. 6, Aug. 2007, S. 2825–2830.

[78] A. Soltanian und R.E. Van Dyck, „Physical layer performance for coexistence of Bluetooth and IEEE 802.11b“, in: *Virginia Tech. Symposium on Wireless Personal Communications*, Juni 2001.

[79] W. Stallings, *Data and Computer Communications*, New York: Macmillan, 1988.

[80] CPN Tools, `http://www.cpntools.org`.

[81] International Telecommunication Union, *ITU Radio Regulations*, 2004.

[82] ETSI TR 102 861 V1.1.1, *Intelligent Transport Systems (ITS); STDMA recommended parameters and settings for cooperative ITS; Access Layer Part*, Jan. 2012.

[83] ETSI TR 102 862 V1.1.1, *Intelligent Transport Systems (ITS); Performance Evaluation of Self-Organizing TDMA as Medium Access Control Method Applied to ITS; Access Layer Part*, Dez. 2011.

[84] ETSI TR 102 889-2 V1.1.1, *Electromagnetic compatibility and Radio spectrum Matters (ERM); System Reference Document; Short Range Devices (SRD); Part 2: Technical characteristics for SRD equipment for wireless applications using technologies different from Ultra-Wide Band (UWB)*, Aug. 2011.

[85] ETSI Draft EN 300 328 V1.8.1, *Electromagnetic compatibility and Radio spectrum Matters (ERM); Wideband transmission systems; Data transmission equipment operating in the 2,4 GHz ISM band and using wide band modulation techniques; Harmonized EN covering essential requirements under article 3.2 of the R&TTE Directive*, Apr. 2012.

[86] ETSI EN 300 175-1 V2.4.1, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 1: Overview*, Apr. 2012.

[87] ETSI EN 300 175-2 V2.4.1, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 2: Physical Layer (PHL)*, Apr. 2012.

[88] ETSI EN 300 175-3 V2.4.1, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 3: Medium Access Control (MAC) layer*, Apr. 2012.

[89] A. Vießmann u. a., „Petri net based controller concept for cognitive radios in wireless access networks“, in: *Proceedings of the ICAT*, London, Great Britain, Juli 2006.

[90] B. Wagner, *Entwurf diskreter Steuerungen (Skript zur Vorlesung)*, `http://www.rts.uni-hannover.de/labor/pneu/skript/Skript.html`, Universität Hannover, Institut für Steuerungstechnik und Fachdidaktik der Elektrotechnik, Okt. 1999.

[91] WiSE - A wireless System Engineering Tool, `http://www.bell-labs.com/org/wireless/wisext.html`.

[92] C. Won, „Adaptive Radio Channel Allocation for Supporting Coexistence of 802.15.4 and 802.11b“, in: *Vehicular Technologie Conference*, Bd. 4, Sep. 2005, S. 2522–2526.

[93] C. Xiong, T. Murata und J. Tsai, „Modelling and simulation of routing protocol for mobile ad hoc networks using coloured Petri nets“, in: *Proc. of Conference on Application and Theory of Petri nets: formal methods in Software Engineering and Defence Systems*, 2002.

[94] O. Younes und N. Thomas, „An SRN Model of the IEEE 802.11 DCF MAC Protocol in Multi-Hop Ad Hoc Networks with Hidden Nodes“, in: *Comput. J.* Bd. 54, 6, 2011, S. 875–893.

[95] W. Yuan, X. Wang und J.-P. M.G. Linnartz, „A Coexistence Model of IEEE 802.15.4 and IEEE 802.11b/g“, in: *IEEE Symposium on Communications and Vehicular Technology*, Nov. 2007.

[96] ZVEI - Zentralverband Elektrotechnik- und Elektronikindustrie e.V. - Fachverband Automation, *Koexistenz von Funksystemen in der Automatisierungstechnik - Erläuterungen zum zuverlässigen Parallelbetrieb von Funklösungen*, 2008.

[97] G. Zeng, H. Zhu und I. Chlamtac, „A novel queueing model for 802.11 wireless LANs“, in: *Proceedings of WNCG Wireless Networking Symposium*, 2003.

[98] Zensys, *WLAN Interference to IEEE 802.15.4*, White paper, März 2007.

[99] C. Zhang und M.C. Zhou, „A stochastic Petri net approach to modelling and analysing of ad hoc networks“, in: *International Journal of Intell. Control Systems*, Bd. 8, 1, Jan. 2003, S. 8–19.

[100] PROFIBUS Nutzerorganisation e.V., *WSAN Air Interface Specification V1.0*, Juni 2012.

[101] PROFIBUS Nutzerorganisation e.V., *WSAN System Specification V1.0*, Juni 2012.

[102] ifak, *Projektbericht „Durchführung von Tests zur Bewertung der Koexistenz verschiedener Funksysteme im industriellen Umfeld (KoTest)“*, Zentralverband Elektrotechnik- und Elektronikindustrie (ZVEI).

[103] ifak, *Projektbericht „Funk-Transfer-Tester für industrielle Funklösungen“*, INNO-WATT, Projekt-Reg. Nr.: IW050103.

[104] ifak, *Projektbericht „Industrial Wireless Diagnosis System (inWiDia)“*, BMBF, Programm KMU-Innovationsoffensive Informations- und Kommunikationstechnologie (IKT), Projekt-Reg. Nr.: 01BN1013.

[105] ifak, *Projektbericht „Zuverlässige Funkkommunikation für funktional sichere Automatisierungssysteme (SafetyRadio)“*, BMWi, Programm InnoNet, Projekt-Reg. Nr.: 16IN0429.

[106] www.software kompetenz.de, *Begriffsdefinition: Echtzeitfähigkeit, Rechtzeitigkeit, Gleichzeitigkeit, Jitter, Determinismus*, `http://www.software-kompetenz.de/?28612`.

[107] openautomation.de, *Ifak erhält Auftrag der ZVEI-Task Force "Wireless Regulierung für die industrielle Automation"*, `http://www.openautomation.de/3229-0-ifak-erhaelt-auftrag-der-zvei-task-force-wireless-regulierung-fuer-die-industrielle-automation.html`, Apr. 2013.

Ehrenerklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die Hilfe eines kommerziellen Promotionsberaters habe ich nicht in Anspruch genommen. Dritte haben von mir weder unmittelbar noch mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen. Verwendete fremde und eigene Quellen sind als solche kenntlich gemacht.

Ich habe insbesondere nicht wissentlich:

- Ergebnisse erfunden oder widersprüchliche Ergebnisse verschwiegen,
- statistische Verfahren absichtlich missbraucht, um Daten in ungerechtfertigter Weise zu interpretieren,
- fremde Ergebnisse oder Veröffentlichungen plagiiert,
- fremde Forschungsergebnisse verzerrt wiedergegeben.

Mir ist bekannt, dass Verstöße gegen das Urheberrecht Unterlassungs- und Schadensersatzansprüche des Urhebers sowie eine strafrechtliche Ahndung durch die Strafverfolgungsbehörden begründen kann.

Ich erkläre mich damit einverstanden, dass die Dissertation ggf. mit Mitteln der elektronischen Datenverarbeitung auf Plagiate überprüft werden kann.

Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht.

Magdeburg, den 2. September 2013 André Schimschar

Lebenslauf

Persönliche Daten

Name	André Schimschar
Anschrift	Hohe Straße 8 39124 Magdeburg
Telefon	01573 7652083
E-Mail	andre.schimschar@ifak.eu
Geboren	21.01.1983 in Schönebeck
Familienstand	ledig

Schulbildung

1989–1993	Karl-Liebknecht-Grundschule Schönebeck
1993–1995	Gymnasium „Am Malzmühlenfeld“ Schönebeck
1995–2002	Europaschule Gymnasium Gommern, Abitur

Wehrdienst

2002–2003	Grundwehrdienst in der Clausewitz-Kaserne Burg

Hochschulstudium

10/2003–06/2009	Otto-von-Guericke-Universität Magdeburg
Fach	Elektrotechnik
Abschluss	Diplom-Ingenieur
Schwerpunkte	Automatisierungstechnik Regelungstechnik Drahtlose Kommunikation

Praktika

1999	Vereins- und Westbank Magdeburg
2005	NOWUS Automatisierungstechnik GmbH
2008	Dr. Weigel Anlagenbau GmbH

Berufliche Tätigkeiten

Seit 07/2009	Wissenschaftlicher Mitarbeiter Institut für Automation und Kommunikation e. V. Magdeburg
06/2011–09/2012	Projektleiter des Forschungsvorhabens „Miniaturisierte energieautarke Komponenten mit verlässlicher drahtloser Kommunikation für die Automatisierungstechnik (MIKOA)“
10/2010–12/2012	Projektleiter des Forschungsvorhabens „Funk im Lebenszyklus industrieller Automation (FiLiA)“

Zusatzqualifikationen

04/2010–03/2013 Stipendium der Phoenix-Contact-Stiftung

Magdeburg, den 2. September 2013