

# Ein Framework für echtzeitfähige Ethernet-Netzwerke

Dr. Frank Dopatka

GFU Cyrus AG

Gesellschaft für Unternehmensberatung, 51105 Köln-Deutz

FrankDopatka@web.de

**Zusammenfassung.** Im Rahmen dieser Arbeit wird die Entwicklung eines formalen Frameworks vorgestellt, mit dessen Hilfe zwischen der Kompatibilität zum verbreiteten baumförmigen Standard-Ethernet und der Einhaltung von Echtzeitanforderungen einer automatisierten Anlage variiert werden kann. Damit kann bereits vor der Auswahl einer bestimmten Technologie eine Schedule der Echtzeit-Übertragungen offline kalkuliert und simuliert werden, sobald die Anforderungen der Geräte im industriellen Echtzeit-Netzwerk bekannt sind.

Ausgehend von einer stark vereinfachten Modellbildung werden in mehreren Schritten Verfeinerungen vorgenommen, die zukünftig eine nahtlose Integration in existierende industrielle Lösungen ermöglichen sollen.

## 1 Motivation

Im Umfeld der Automatisierungstechnik ist bereits seit einigen Jahren der Trend zu beobachten, etablierte Feldbusse durch echtzeitfähiges Ethernet zu ersetzen oder zu ergänzen. Das Ziel liegt darin, eine einzige Netzwerkinfrastruktur von der Leitebene im Bürobereich bis hin zu Feldgeräten wie Sensoren, Aktoren und Bedienpulten in industriellen Produktionshallen zu schaffen. Dieser Trend wird als vertikale Integration bezeichnet [1]. Ein Hauptproblem liegt dabei darin, dass die weit verbreiteten Standard-Ethernet Netzwerke nicht den harten Anforderungen der industriellen Echtzeit genügen. So ist bereits in einem kollisionsfreien Vollduplex-Ethernet die Zwischenspeicherung eines Datenpaketes ausreichend, um den Anforderungen der härtesten Echtzeitklasse der Industrial Automation Open Networking Alliance (IAONA) [2] zu widersprechen, bei der die maximale Verzögerung eines Frames im Netzwerk  $1ms$  und die Schwankung dieser Verzögerung – der Jitter – maximal  $1\mu s$  beträgt. Andererseits soll jedoch der Ethernet-Standard auch auf der Feldebene eingehalten werden, da ansonsten wiederum separate Feldbusse entstehen.

Es existiert bereits eine Reihe von industrie-tauglichen Lösungen für echtzeitfähiges Ethernet, die im folgenden Kapitel kurz vorgestellt und klassifiziert werden. Es lässt sich erkennen, dass jeder Ansatz einen individuellen Kompromiß zwischen den Echtzeitanforderungen aus der Automatisierungstechnik und der Kompatibilität zu Standard-Ethernet darstellt. Die eigene Forschung hat sich *nicht* zum Ziel gesetzt, einen weiteren dieser Kompromisse hervorzubringen.

Statt dessen wird die Grundlage für ein technologie-unabhängiges, mathematisch formales Framework gelegt, das im dritten Kapitel vorgestellt wird. Die Vorgehensweise bei der Entwicklung des Frameworks verläuft von einer stark vereinfachten Modellierung und Lösungsfindung schrittweise zu realitätsnäheren Ansätzen. Das Ziel besteht darin, möglichst viele der existierenden Ansätze in dem allgemeinen Framework formal zusammenzufassen. Im dritten Kapitel werden exemplarisch idealisierte Vollduplex-Netze unter Verwendung von Switches betrachtet. Im vierten Kapitel werden dann zwei Verfeinerungen der Modellierung vorgestellt, indem zunächst Hubs und anschließend Verzögerungszeiten in die Modellierung aufgenommen werden. Die Betrachtung endet mit einer Zusammenfassung der erreichten Ergebnisse und mit Perspektiven für die zukünftige Forschung.

## 2 Stand der Forschung

Durch das Fehlen eines deterministischen Medienzugriffs, der Pufferung von Daten-Frames in Switches und ggf. sogar deren Verwerfung bei vollen Puffern sowie durch einen hohen Protokoll-Overhead im Vergleich zu Feldbus-Protokollen können härteste Echtzeitanforderungen der Antriebstechnik nicht mit Standard-Ethernet nach IEEE 802.3 umgesetzt werden. Dies beinhaltet insbesondere die Forderung nach Isochronität. Durch die konstanten Perioden isochroner Übertragungen (IRT) sollen mechanische Wellen durch eine zyklische, hoch präzise getaktete Verbreitung der Ist-Werte einer Master-Achse zu ihren Slaves ersetzt werden. Man wünscht sich in diesem Zusammenhang eine „Software-Welle“ [3].

Die IRT-Übertragungen sind in der Automatisierungstechnik im Vorfeld bekannt. Es lässt sich also im Voraus bestimmen, welcher Sensor, Aktor oder welche Steuerung zu welchem Zeitpunkt senden darf. Ebenso kann die maximale Größe der Echtzeit-Frames im Vorfeld festgelegt werden. Im Gegensatz zu herkömmlichen Ethernet-Übertragungen, bei denen u. a. Multimedia-Daten oder E-Mails mit Anhängen übertragen werden, werden im industriellen Bereich lediglich einige Bytes an Nutzdaten pro Frame versendet, die zumeist Soll- und Ist-Positionen der Antriebe enthalten.

Ein Lösungsansatz besteht darin, eine Modifikation zum Ethernet-Standard oberhalb der Sicherungsschicht vorzunehmen, wie es unter anderem bei dem verbreiteten Ethernet PowerLink (EPL) der Fall ist. Dabei kommen  $100\text{MBit/s}$ -Hubs anstelle von Switches zum Einsatz, da die Verzögerungszeiten der Frames bei Hubs um ein Vielfaches geringer sind. Nachteile bestehen darin, dass neben dem Einsatz von veralteten Hubs lediglich eine Kommunikation zu einem Zeitpunkt im Netzwerk erlaubt ist, da durch die Halbduplex-Übertragung das CSMA/CD-Verfahren mit nicht-deterministischer Backoff-Strategie zum Einsatz kommt. Zusätzlich halbiert der Halbduplex-Betrieb die physikalische Bandbreite des Netzes. Auch zerstören Übertragungen von asynchron sendenden Geräte, die nicht dem EPL-Protokoll folgen, die Echtzeitfähigkeit des Netzes.

Ein zweiter Lösungsansatz liegt in der Ersetzung oder Ergänzung der physikalischen Ethernet-Schicht inklusive der Medienzugriffssteuerung. Die Ersetzung

hat den Zweck, für die härteste IAONA-Echtzeitklasse optimierte Hardware bereit zu stellen. Als Vertreter dieses Ansatzes existieren ProfiNet IRT, SERCOS III sowie EtherCAT. Der Vorteil der optimierten Hardware für die Automatisierungstechnik spiegelt sich gleichzeitig als Nachteil für die Kompatibilität zum Ethernet-Standard wider.

Aus diesen Ansätzen ist zu entnehmen, dass zwar viele gute und auch effiziente Lösungen existieren, die Lösungen jedoch stets einen konkreten Kompromiß zwischen Kompatibilität zum Standard und den Anforderungen der Automatisierungstechnik darstellen. Das zu erstellende Framework soll die Basis für den flexiblen, anwendungsbezogenen Aufbau von industriell echtzeitfähigen Netzen schaffen.

### 3 Grundlagen der Modellierung

Die Idee der Modellierung besteht darin, im ersten Schritt das zu erstellende Netzwerk formal zu modellieren. Die Position der einzelnen Geräte mit verschiedenen hohen Anforderungen an die Echtzeit – wie Antriebe, Sensoren und Aktoren – sowie deren Verkabelung lässt sich zum Großteil aus der mechanischen Konstruktion der konkreten Anlage ableiten. Im Anschluß daran werden die Übertragungen mit hohen Echtzeitanforderungen modelliert, die über das Netzwerk verlaufen.

IRT-Übertragungen, die gleiche Wege im Netzwerk durchlaufen, dürfen unter Umständen nicht gleichzeitig ausgeführt werden, da dadurch eine erhöhte Verzögerungszeit und insbesondere ein erhöhter Jitter resultieren würde. Zwischen diesen Übertragungen entstehen somit „Konflikte“, die zeitlich zu entzerren sind.

Aus den Konflikten wird ein Konfliktgraph gebildet, der durch seine anschließende Färbung die zeitliche Entzerrung vornimmt. Die einmalige Abarbeitung der Schedule ist gleichzusetzen mit einer Umdrehung der „Software-Welle“ in der automatisierten Anlage und stellt einen Produktions-Zyklus dar.

#### 3.1 Idealierte Vollduplex-Verbindungen

Um sich dem komplexen Problem zu nähern, wird im ersten Schritt von einem stark vereinfachten Netz ausgegangen. Dabei wird die ausschließliche Verwendung von idealisierten, nicht blockierenden Switches angenommen. Dies bedeutet, dass bei  $n$  Ports im Idealfall  $n/2$  Übertragungen gleichzeitig möglich sind, ohne dass Daten zwischengespeichert oder verworfen werden. Ein auf Switches basierendes baumförmiges Netz arbeitet nach IEEE 802.3 im Vollduplex-Modus, in dem auf einem Ethernet-Kabel gleichzeitig gesendet und empfangen werden kann. Verzögerungszeiten und deren Jitter werden zunächst ignoriert. Des Weiteren werden einheitliche Framelängen angenommen. Die zusätzliche Übertragung von asynchronen Frames wird nicht betrachtet und später in die entstandene Schedule der Echtzeitübertragungen integriert. Es wird ebenso angenommen, dass jede IRT-Übertragung in jedem Sendezyklus auftritt und ausschließlich Unicast-Übertragungen vorliegen.

Im ersten Schritt wird die Netzwerk-Infrastruktur als Graph modelliert. Über diesen Graph verlaufen die IRT-Übertragungen, deren Sendezeitpunkte zu planen sind. Das Netzwerk besteht aus einer endlichen Menge von Switches und einer endlichen Menge von Geräten, die zu einer Knotenmenge des Netzes zusammengefasst werden. Angeschlossene Geräte bilden die Blätter im Netzwerk-Baum. Zusätzlich dazu existiert eine endliche Menge von Netzwerkkabeln, wobei ein Netzwerkkabel zwischen zwei Knoten als Kante zwischen den verbundenen Knoten repräsentiert wird.

Ausserdem existiert eine endliche Menge von IRT-Übertragungen, die als Kommunikationslinien bezeichnet werden. Da es sich bei dem Netzwerk um einen Baum handelt, ist der Weg jeder Unicast-Übertragung eindeutig bestimmt. Jeder Kommunikationslinie wird eine einheitliche positive Übertragungsdauer zugeordnet. Jede Kommunikationslinie verläuft von genau einem Quellgerät zu genau einem Zielgerät über einen oder mehrere Knoten des Netzwerkes.

Jeder Port eines Switches besitzt einen unabhängigen Eingangs- und Ausgangsteil. Wird ein Daten-Frame zu einem Zeitpunkt an einem Eingangs- oder Ausgangsteil eines Ports übertragen, an dem bereits ein anderer Frame übertragen wird, wird der zweite Frame verkehrsabhängig gepuffert und im Falle der Überlast sogar verworfen. Dies wird im Folgenden als „Konflikt“ bezeichnet, der aufzulösen ist, da die nicht-deterministische, lastabhängige Verzögerungszeit der Frames in harten Echtzeitumgebungen nicht tolerierbar ist.

Das Auflösen der Konflikte besteht also darin, dafür zu sorgen, dass die betreffenden Kommunikationslinien nicht zur selben Zeit an den konflikt-trächtigen Knoten im Netz ausgeführt werden. Diese zeitliche Entzerrung bedeutet eine Ablaufplanung der Übertragungen, also die Bildung einer Schedule.

Durch die Definition des Konflikt-Begriffes können nun zwei Aussagen getroffen werden, die für die Berechnung der Schedule von Bedeutung sind. Die erste Aussage besteht darin, dass eine lokale Konfliktfreiheit eine globale Konfliktfreiheit impliziert [4, Satz 4.2.4]. Durchlaufen zwei verschiedene Kommunikationslinien einen gemeinsamen Switch und haben dort keinen Konflikt, weil sie nicht in dieselbe Richtung verlaufen, so sind sie global konfliktfrei. Die zweite Aussage besteht darin, dass Konfliktknoten stets einen Weg bilden [4, Satz 4.2.6], also jeweils durch eine eindeutige, zusammenhängende Folge von Knoten miteinander verbunden sind.

Im Anschluss an die Modellierung des Netzes und der Übertragungen wird der Kantenkonfliktgraph betrachtet. Die Idee besteht darin, an dieser Stelle auf die Modellierung der Ports eines Switches zurückzugreifen. Die beiden Teile bilden je einen Knoten im Kantenkonfliktgraph. Jede Unicast-Übertragung verläuft von genau einem Eingangsport zu genau einem Ausgangsport und bildet eine Kante im Graphen. Sind mehrere Kanten mit einem Knoten verbunden, so müssen Sie bei einer gültigen Kantenfärbung verschiedene Farben erhalten. Erhalten zwei Kommunikationslinien dieselbe Farbe, so können sie unabhängig voneinander ausgeführt werden. Jede Farbe repräsentiert einen Zeit-Slot in der resultierenden Schedule und die Anzahl der Farben multipliziert mit der Übertragungszeit der

Frames die Zykluszeit des Netzes. Eine optimale Färbung bedeutet gleichzeitig eine optimale Schedule mit minimaler Ausführungsdauer.

Es wurde bewiesen, dass Kantenkonfliktgraph eines Vollduplex-Netzwerkes bipartit ist [4, Satz 4.4.8], so dass die Färbung erheblich vereinfacht wird. Zunächst ist bereits bei der Entstehung des Konfliktgraphen bekannt, dass er mit  $\chi'(G) = \Delta(G)$  gefärbt werden kann. Auf diese Weise kann vor der Färbung bereits der Switch definitiv ermittelt werden, der für die meisten Zeit-Slots der Schedule verantwortlich sein wird. Ein einfacher optimaler  $O(|E| \cdot \log(|E|))$ -Algorithmus zur Färbung bipartiter Graphen stammt von Alon [5] aus dem Jahre 2003.

Die Schedule für einen Switch aus der Netzwerkinfrastruktur lässt sich leicht erzeugen, sofern der gefärbte Konfliktgraph gegeben ist [4, Satz 6.3.1/2]. Man durchläuft alle gefärbten Kanten und sortiert die Anforderungen mit gleicher Farbe  $F_i$  jeweils in einen Vektor, der dann einem Zeit-Slot entspricht. Da die lokalen Schedules für jeden Switch unabhängig erstellt werden können, ist es möglich, dass derselben Übertragung in zwei lokalen Schedules verschiedene Zeit-Slots zugewiesen werden.

Ein Problem besteht jedoch darin, dass eine Kommunikationslinie, welche über mehrere Switches verläuft, bei unabhängiger Berechnung der lokalen Schedules verschiedenen Zeit-Slots zugeordnet werden kann. Es wurde bewiesen, dass die lokalen Schedules nicht stets zu einer globalen Schedule durch Um-Indizierung der Zuordnungen synchronisiert werden können [4, Satz 6.2.2].

Eine Idee zur Lösung besteht darin, zunächst alle Kantenkonfliktgraphen zu erzeugen und exakt zu färben. Ein Switch mit der größten Anzahl an Farben stellt dann den Wurzel-Switch  $w$  für die weitere Färbung dar. Treten IRT-Übertragungen auch in Switches auf, die direkt mit  $w$  verbunden sind, so erhalten sie jeweils dieselbe Farbe, wie sie in  $w$  besitzen. Im Anschluss daran werden die restlichen Übertragungen des Konfliktgraphen neu gefärbt. Dieser Vorgang wiederholt sich für das gesamte Netzwerk. Auf diese Weise erhält man in einem Schritt synchronisierte Schedules. Der Algorithmus hat Nowak [4, S. 118] formal beschrieben.

## 4 Erweiterung des Frameworks

In diesem Kapitel werden nun zwei Erweiterungen des Frameworks vorgestellt, welche die Modellierung realitätsnäher gestalten. Dazu gehört einerseits die Integration von Hubs in das formalisierte Netzwerk und andererseits die Berücksichtigung von Verzögerungszeiten in Switches sowie deren Auswirkung auf die erstellten Schedules.

### 4.1 Einbindung von Hubs

An einem Hub existiert zu einem Zeitpunkt maximal eine Übertragung, die von allen angeschlossenen Geräten empfangen werden kann. Eine besondere Situation ergibt sich, sobald ein Hub zusammen mit Switches in einem Netzwerk kombiniert wird. Abbildung 1 zeigt fünf Kommunikationslinien, die zwischen zwei

Switches und einem Hub verlaufen. An den Hub ist an Gerät G1 eine Master-Achse sowie 6 Slave-Achsen (G2 bis G7) angeschlossen.

Während die beiden Kommunikationslinien 1 und 2 aufgrund des Halbduplex-Betriebes des Hubs sowie an Port 0 von Switch 1 in Konflikt stehen, sind sie in Switch 2, der ausschließlich im Vollduplexbetrieb arbeitet, konfliktfrei. In diesem Fall impliziert die lokale Konfliktfreiheit also keine globale Konfliktfreiheit. Daher ist ein globaler Knotenkonfliktgraph zu bilden, der das gesamte Netzwerk betrachtet (s. Abbildung 1b). Des Weiteren ist zu beachten, dass die dritte Kommunikationslinie auch vom Port 0 des Switches 1 empfangen wird. Trotz der Übertragung innerhalb des Hubs ist der gesamte Port 0 dieses Switches für die Dauer der Übertragung gesperrt. Da der Hub ausschließlich Broadcasts sendungen zulässt, kann er als einziges Gerät  $D^*$  im Halbduplex-Betrieb angesehen werden. Alle Übertragungen stehen dort in Konflikt zueinander und benötigen einen eigenen Zeit-Slot.

Die Färbung kann mit einem beliebigen Färbungs-Algorithmus zur Knoten-Färbung erfolgen. Aus dem knoten-gefärbten Graphen lassen sich die lokalen Schedules der Switches ableiten (s. Abbildung 1c), indem lediglich die IRT-Übertragungen betrachtet werden, die über den betreffenden Switch verlaufen.



Abb. 1. Netzwerk (a), globaler Konfliktgraph (b) und lokale Schedules (c)

## 4.2 Berücksichtigung von Verzögerungszeiten

Die Modellierung ist bislang vereinfachend davon ausgegangen, dass die Knoten des Netzes die IRT-Übertragungen nicht verzögern. Dies ist in der Praxis nicht der Fall. Die Verzögerung eines Hubs liegt in der Praxis sie bei ca.  $0.4\mu s$  [6], die eines ProfiNet Switches bei ca.  $3\mu s$  [7]. Modelliert man eine Linientopologie (s.



Zeit-Slot  $T_{V, Ges} + T_U = 37\mu s$  lang sein. Die Zykluszeit wird auf diese Weise von  $28\mu s$  ohne Berücksichtigung der Verzögerungen auf  $4 \cdot 37\mu s = 148\mu s$  erhöht.

Die Einordnung der Übertragungen in die globale Schedule nach ihrem frühest möglichen Start-Zeitpunkt stellt eine Optimierung dar. Dies ist mit einem List-Scheduling bzw. einer First-Fit Vorgehensweise zu vergleichen, das aufgrund der im Vorfeld bekannten IRT-Übertragungen offline durchgeführt werden kann. Die Auflösung der Zeit-Slots hat zur Folge, dass eine Färbung des Konfliktgraphen nicht mehr benötigt wird. Die Erstellung des globalen Knotenkonfliktgraphen ist dennoch sinnvoll, um die Konflikte zwischen einzelnen Kommunikationslinien zu erkennen.

Ähnlich wie bei der Graphenfärbung werden dabei zunächst die IRT-Übertragungen in einer Reihenfolge vorsortiert. Entsprechend dieser Reihenfolge werden die IRT-Übertragungen nacheinander direkt in die Schedule eingefügt unter der Prämisse der möglichst frühen Sendung. Es wird also vom Beginn der Schedule bei  $0\mu s$  untersucht, ob die gerade betrachtete IRT-Übertragung eingefügt werden kann oder nicht. Ist dies nicht der Fall, so wird eine erneute Untersuchung unmittelbar nach Beendigung der Übertragung, mit welcher die betrachtete Übertragung in Konflikt steht, durchgeführt. Dies geschieht so lange über alle Switches, die von der betrachteten IRT-Übertragung betroffen sind, bis ein freier Bereich innerhalb der Schedule gefunden wird. Diese Vorgehensweise ist mit dem Greedy-Algorithmus der Graphenfärbung vergleichbar. Beim Greedy-Algorithmus der Graphenfärbung existiert jedoch stets mindestens eine Vorsortierung, welche zu einem optimalen Ergebnis führt [8]. Eine solche optimale Vorsortierung existiert auch bei der Anwendung des List-Schedulings. Die Aufgabe besteht nun darin, eine optimale Vorsortierung der IRT-Übertragungen zu ermitteln.

Nach Abbildung 2b wird zunächst die Anzahl der Konflikte jeder IRT-Übertragung anhand des globalen Knotenkonfliktgraphen ermittelt. Die Anzahl ist jeweils an den Knoten, welche die jeweilige Übertragung repräsentiert, verzeichnet. Im Anschluss daran werden die Übertragungen anhand der Anzahl der Konflikte nacheinander in die Schedule eingefügt, wobei mit der IRT-Übertragung mit den meisten Konflikten begonnen wird.

Eine sinnvolle Heuristik besteht darin, die Anzahl der Konflikte der Übertragungen zu zählen, welche noch nicht in die Schedule integriert worden sind und die Knoten der bereits eingefügten Übertragungen mit den inzidenten Kanten aus dem Konfliktgraphen zu entfernen. Dies entspricht einer dynamischen Sortierung und ist mit dem Sättigungsgrad des DSATUR-Algorithmus der Graphenfärbung [9] zu vergleichen, der in der Praxis gute Ergebnisse erzeugt.

Die in Abbildung 3 dargestellte resultierende Schedule liegt bei  $47\mu s$  im Vergleich zu  $148\mu s$  des ersten beschriebenen Ansatz. So benötigt beispielsweise die IRT-Übertragung 1 an Switch S1  $7\mu s$  für den Durchlauf des Daten-Frames zusätzlich  $3\mu s$  Verzögerung, die in dem grauen Block der ersten Zeile der Schedule dargestellt wird. Die Verzögerungszeiten verlängern die idealisierte Schedule, die  $28\mu s$  lang ist, um 68% bei der Anwendung des List-Schedulings. Der erste beschriebene Ansatz hingegen hat die Verzögerungszeiten pauschal auf jeden Zeit-Slot addiert und die Schedule dabei um 529% erhöht.

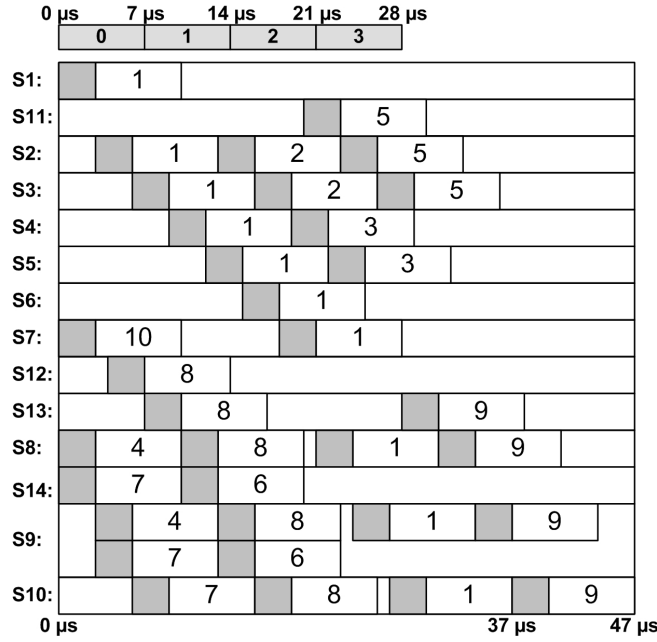


Abb. 3. Optimierte Schedule mit Berücksichtigung der Verzögerungen der Switches

Eine weitere interessante Eigenschaft des List-Schedulings besteht darin, dass es unabhängig von der Framelänge operieren kann. Ebenso können bei der Anwendung des List-Schedulings neben den Unicast-Übertragungen auch Multicast- und Broadcast-Übertragungen berücksichtigt werden. Dabei erhöhen sich lediglich die Anzahl der betroffenen Switches sowie die Anzahl der Konflikte im globalen Knotenkonfliktgraphen. Werden an den Rändern des Netzwerkes Hubs eingesetzt, so können diese jeweils wie ein einzelnes Gerät betrachtet werden.

### 5 Zusammenfassung und Ausblick

Die erstellte formale Modellierung beinhaltet neben dem bereits vorgestellten Szenario des idealisierten Vollduplex-Netzes mit Unicast-Übertragungen zusätzlich idealisierte Halbduplex-Netze [10] sowie die Integration von Multicast- und Broadcast-Übertragungen auf eine gegebene Ethernet-Netzwerkinfrastruktur [11]. Ausserdem werden Wege zur Einbindung von asynchron auftretenden Übertragungen in das formale Modell diskutiert [11]. Siemens ProfiNet und EPL beispielsweise teilen ihre Schedules in jeweils einen isochronen und einen asynchronen Teil ein. Zusätzlich wurden im Rahmen der Forschung Modelle zur Einbindung variabler Frame-Größen ebenso diskutiert wie die Möglichkeit, nicht jeden Frame in jedem Produktionszyklus zu versenden. So müssen unter ande-

rem unkritische Sensor-Daten nicht zwingend in jedem Kommunikationszyklus übertragen werden und können sich statt dessen abwechseln [11].

Mit diesem Framework wurde also ein grundlegendes formales Modell und eine technologie-unabhängige Vorgehensweise zur Berechnung von Zykluszeiten automatisierter Anlagen geschaffen. Die ersten realitätsnahen Erweiterungen binden Technologien wie Siemens ProfiNet und Ethernet PowerLink in die Modellierung ein.

Als nächste Erweiterungen könnten Rundsende-Techniken von Frames, wie sie von Sercos III und EtherCAT verfolgt werden, in die Modellierung integriert werden. Ebenfalls sinnvoll ist es, das Modell auf reale automatisierte Anlagen anzuwenden, um Vergleiche zwischen einer theoretisch ermittelten und realen Zykluszeit anzustellen und um den praktischen Einsatz des Modells zu erproben. Dazu wurde bereits ein erster objektorientierter Prototyp geschaffen, der eine Simulation von Netzen, Protokollen und Schedule-Berechnungen mit verschiedenen Algorithmen zulässt [11]. Die Vollendung dieses Prototyps kann ein weiterer Bestandteil zukünftiger Forschung darstellen.

## Literaturverzeichnis

1. Soucek S., Loy D.: Vertical Integration in Building Automation Systems. in: Proc. of the 5th IEEE Int. Conf. on Industrial Informatics 2007, Vol. 1, S. 81–86, 2007.
2. Schnell G.: Bussysteme in der Automatisierungs- und Prozesstechnik. Vieweg-Verlag, Braunschweig, Wiesbaden, 5. Auflage, 2003.
3. Moning A., Lanz R.: Datenkommunikation und Rechnernetze, 2006, FH Bern, Skript zur Vorlesung, <http://www.sws.bfh.ch/~lanz/SKRIPTE/DataKomm.pdf> (01.07.07).
4. Nowak, U.: Graphentheoretische Modellierung eines automatisierungstechnischen Echtzeitnetzwerks und Algorithmenentwurf zum Kommunikations-Scheduling, Diplomarbeit der Universität Siegen, Fachbereich Mathematik, <http://www.uwenowak.de/arbeiten/diplomarbeit.pdf> (11.11.07).
5. Alon N.: A Simple Algorithm for Edge-Coloring Bipartite Multigraphs, in: Information Processing Letters 85, Nr. 6, 2003, S. 301-302, ISSN 0020-0190, [http://dx.doi.org/10.1016/S0020-0190\(02\)00446-5](http://dx.doi.org/10.1016/S0020-0190(02)00446-5).
6. Scheitlin H., Zuber R.: Kommunizieren Sie pünktlich?, 2003, [http://www.edison.ch/uploads/media/Kommunizieren\\_Sie\\_puenktlich.pdf](http://www.edison.ch/uploads/media/Kommunizieren_Sie_puenktlich.pdf) (03.07.07).
7. Popp M.: Das Profinet IO-Buch - Grundlagen und Tipps für Anwender, Hüthig-Verlag, Heidelberg, 2005, ISBN 3-7785-2966-8.
8. Turau V.: Algorithmische Graphentheorie, 2. Auflage, Addison-Wesley Verlag, München, 2004, ISBN 3-4862-0038-0.
9. Brélez D.: New Methods to Color the Vertices of a Graph, in: Communications of the ACM 22, Nr. 4, 1979, S. 251-256, ISSN 0001-0782, <http://dx.doi.org/10.1145/359094.359101> (15.11.07).
10. Dopatka F., Wismüller R.: A Top-Down Approach for Realtime Industrial-Ethernet Networks using Edge-Coloring of Conflict-Multigraphs, in: IEEE International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEE-DAM), Taormina, Italien, 23.-26.05.2006, ISBN 1-4244-0193-3, S. 883-890.
11. Dopatka F.: Ein Framework für echtzeitfähige Ethernet-Netzwerke in der Automatisierungstechnik mit variabler Kompatibilität zu Standard-Ethernet, Dissertation der Universität Siegen, Fachbereich Elektrotechnik und Informatik, 2008, <http://dokumentix.ub.uni-siegen.de/opus/volltexte/2008/370/> (27.06.09).