Douglas Cunningham, Petra Hofstedt, Klaus Meer, Ingo Schmitt (Hrsg.): INFORMATIK 2015
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2015

Universelle Echtzeit-Ethernet Architektur zur Integration in rekonfigurierbare Automatisierungssysteme

Johannes Ax[1], Aurel Buda[2], Daniel Schneider[3], John Hartfiel[1], Lars Dürkop[4], Thorsten Jungeblut[1], Jürgen Jasperneite[3], Andreas Vedral[2] und Ulrich Rückert[1]

Abstract: Das neue Geschäftsmodell, das mit der Industrie 4.0 verbunden ist, verändert ein etabliertes Dogma der Automatisierungsbranche: Anlagen wurden auf hohen Durchsatz der immer gleichen Produktformate optimiert. Während derartige Systeme mit bekannten Maßnahmen automatisierbar sind, stellt die Variantenfertigung eher ein Automatisierungshindernis dar. Für eine individualisierte Produktion ist die Rekonfigurierbarkeit von Industrieanlagen ein wichtiger Faktor. Die Modularisierung und automatische Konfiguration sind hierbei wesentliche Triebkräfte. Im Rahmen dieser Arbeit wird eine Basisarchitektur für eine intelligente Netzwerkkomponente vorgestellt, die ohne manuelle Konfiguration in unterschiedlichen Echtzeit-Ethernet-Netzwerken eingesetzt werden kann.

Keywords: RMS, RTE, FPGA, partielle dynamische Rekonfiguration

1 Einleitung

Die Automatisierungsbranche wandelt sich zusammenhängend mit den Strömungen im Kontext der „Industrie 4.0“ [AM14, Zue10] immer weiter durch den zunehmenden Einsatz von Informations- und Kommunikationstechnologien (IKT). Es müssen daher Ansätze und Methoden entwickelt werden um die hieraus resultierenden Herausforderungen entlang der Wertschöpfungskette zu lösen:

a.) Komplexitätsanstieg durch Informations- und Kommunikationstechnologien

> Die Komplexität, der Umfang und die Reaktionsfähigkeit von Produktionssystemen verzeichnen einen starken Anstieg [BR15]. Von hoher Bedeutung sind hierbei Themen wie die Optimierung, Flexibilisierung und Beschleunigung von Produktionsprozessen. Diese Ausgangssituation löst eine zunehmende Überforderung der Menschen aus, die mit diesen Systemen, zum Beispiel für die Aufgabenfelder Programmierung, Bedienung, Instandhaltung und Service, arbeiten.

[1] CITEC, Universität Bielefeld, Kognitronik und Sensorik, Inspiration 1, 33615 Bielefeld, {jax, jhartfiel, tj, rueckert}@cit-ec.uni-bielefeld.de
[2] WAGO Kontakttechnik GmbH & Co. KG, Automation Development, Hansastr. 27, 32423 Minden, {aurel.buda, andreas.vedral}@wago.com
[3] Fraunhofer IOSB-INA, embedded systems for automation, Langenbruch 6, 32657 Lemgo, {daniel.schneider, juergen.jasperneite}@iosb-ina.fraunhofer.de
[4] inIT, Hochschule Ostwestfalen-Lippe, Langenbruch 6, 32657 Lemgo, {lars.duerkop, juergen.jasperneite}@hs-owl.de

Dieser Umstand wiederum senkt die Arbeitseffektivität der Benutzer von Automatisierungssystemen im Entwurf, im Aufbau und in der Wartung von Anlagen. Die Folge: Aufwändige Verifikation und Validierung, lange Inbetriebnahmezeiten im Anlagenhochlauf, hohe Fehleranfälligkeit durch verbleibende Hard- und Softwarefehler und damit verbunden die Gefahr von Stillstandzeiten sowie ein häufig nicht ressourcen-optimierter Betrieb der automatisierten Anlagen. Ähnliche Problemstellungen sind auch aus anderen hochtechnisierten Bereichen bekannt; die Gesellschaft für Informatik (GI) sieht hier dringenden Handlungsbedarf zur besseren Beherrschbarkeit derartig komplexer technischer Systeme [GI08].

b.) Rekonfiguration von Maschinen, Anlagen und Sensorik

In Verbindung hiermit zeigt sich ein zunehmender Trend hin zu einer hochflexiblen Variantenfertigung, welche die klassische Automatisierung an die Grenzen der Wirtschaftlichkeit bringt.

Maschinen oder Anlagen, die nie oder sehr selten umgerüstet werden, sind überwiegend auf hohen Durchsatz der immer gleichen Produktformate optimiert. Derartige Systeme können mit einer weitestgehend statischen Konfiguration betrieben werden. Produktionsprozesse mit einer hohen Variantenzahl und niedriger Losgröße haben dynamischere Anforderungen. Dies impliziert, dass Produktionsprozesse, die z.T. erhebliche Echtzeitanforderungen aufweisen können (z.B. synchrone Antriebsregelungen), mit geringen Stillstandzeiten erweitert oder umgebaut werden müssen. Benötigt werden Mechanismen, die eine Rekonfiguration mit geringem Aufwand erlauben.

Ein potenzielles Konzept ist das sogenannte „Reconfigurable Manufacturing System (RMS)“ [Ko06]. Hierbei können Module und Maschinen in einen Produktionsprozess hinzugefügt oder entfernt werden.

Ein Ansatz zur Umsetzung eines RMS ist die Einführung Service Orientierter Architekturen (SOA) in der industriellen Automatisierung. Dem Anwender wird hierdurch geholfen komplexe Prozesse und Systeme auf eine handhabbare Art zu partitionieren. Der SOA Ansatz erfordert vor allem, dass die verwendeten Feldgeräte eine semantische Beschreibung ihrer Eigenschaften und Dienste bereitstellen, um im gesamten Automatisierungssystem erkannt und nützlich eingesetzt zu werden. In [Lo13] und [Gi14] wurden Architekturen vorgestellt, um SOAs auch für Industrieanlagen mit Echtzeitanforderungen zu ermöglichen. Die Grundidee bei diesen Arbeiten ist die Kapselung der Echtzeitkommunikation in Module. Ein Modul ist eine Teilkomponente die eine spezielle Automatisierungsfunktion anbietet. Das Konzept der Module wurde auch in [Du14] aufgenommen. Die Architektur in dieser Arbeit erlaubt jedoch zusätzlich die Rekonfigurierbarkeit eines Moduls, sodass ein Hinzufügen und Entfernen einzelner Feldgeräte auch innerhalb eines Echtzeitnetzes möglich ist.

Die Echtzeitkommunikation in Automatisierungsnetzen, also auch innerhalb der hier vorgestellten Module, basiert zunehmend auf Echtzeit-Ethernet-Protokollen (RTE[5]-Protokolle). Dabei haben die letzten zehn Jahre zahlreiche RTE-Protokolle hervorgebracht, die nicht kompatibel zueinander sind. Für Hersteller von Automatisierungskomponenten sowie Analgen und Maschinenbauer ist das eine große Herausforderung. Denn Anlagenteile und Komponenten, die eigentlich die gleiche Funktion erfüllen, müssen für unterschiedliche RTE-Protokolle teils neu entworfen werden.

Aus diesem Grund wird in dieser Arbeit eine universelle Architektur vorgestellt, die unterschiedliche RTE-Protokolle unterstützt. Sie bietet neben dem Austausch von Protokollen mit harten Echtzeitanforderungen auch eine automatische Protokolldetektion. Hierdurch werden die Entwicklung von Feldgeräten und deren Integration in ein Modul vereinfacht.

2 Rekonfigurierbares Modul

Die in [Du14] vorgestellte Architektur eines Moduls erlaubt das Hinzufügen und Entfernen einzelner Teilnehmer, auch wenn zwischen diesen eine Echtzeitkommunikation stattfindet. Abbildung 1 zeigt die grundlegende Architektur eines Moduls.

Hinzugefügte Feldgeräte (IO-Devices) melden sich über einen Ad-hoc Kanal (TCP/IP-Stack) mit all ihren Eigenschaften und Diensten bei der Steuerung (PLC/SPS[6]) an. Die neu gewonnene Funktionalität wird in die semantische Beschreibung des Moduls übernommen, sodass die SPS weiterhin alle Eigenschaften und Funktionalitäten des gesamten Moduls kapselt. Über das Modul-Interface wird diese schließlich anderen Modulen zur Verfügung gestellt. Zusätzlich führt die SPS eine Autokonfiguration des RTE-Protokolls durch, wodurch der Aufwand zur Inbetriebnahme des erweiterten Moduls deutlich reduziert wird. Für weitere Details verweisen wir auf [Du14].

Als RTE-Stack (orange Blöcke in Abbildung 1) können in dieser Architektur die verschiedensten RTE-Protokolle zum Einsatz kommen. Die konkrete Auswahl hängt typischerweise von vielen Faktoren ab. Neben den technologischen Eigenschaften können benötigte Komponenten (z.B. spezielle Sensoren), die nur für ein Protokoll verfügbar sind, die Auswahl bereits bestimmen. Sehr häufig wird das Protokoll von den Auftraggebern vorgegeben (z.B. regional oder politisch bedingt). Die Unterstützung möglichst vieler RTE-Protokolle ist daher ein Wettbewerbsvorteil. Das gilt sowohl für Hersteller von Automatisierungskomponenten als auch Anlagenbauer, die diese verwenden. Wünschenswert ist eine universelle RTE-Plattform, die mit ein und derselben Hardware möglichst viele Protokolle unterstützt. Auf dieser Basis können Komponenten, unabhängig vom RTE-Protokoll, entsprechend Ihrer Semantik und Funktion entwickelt und verwendet werden.

[5] Real Time Ethernet
[6] Programmable logic controller / Speicherprogrammierbare Steuerung

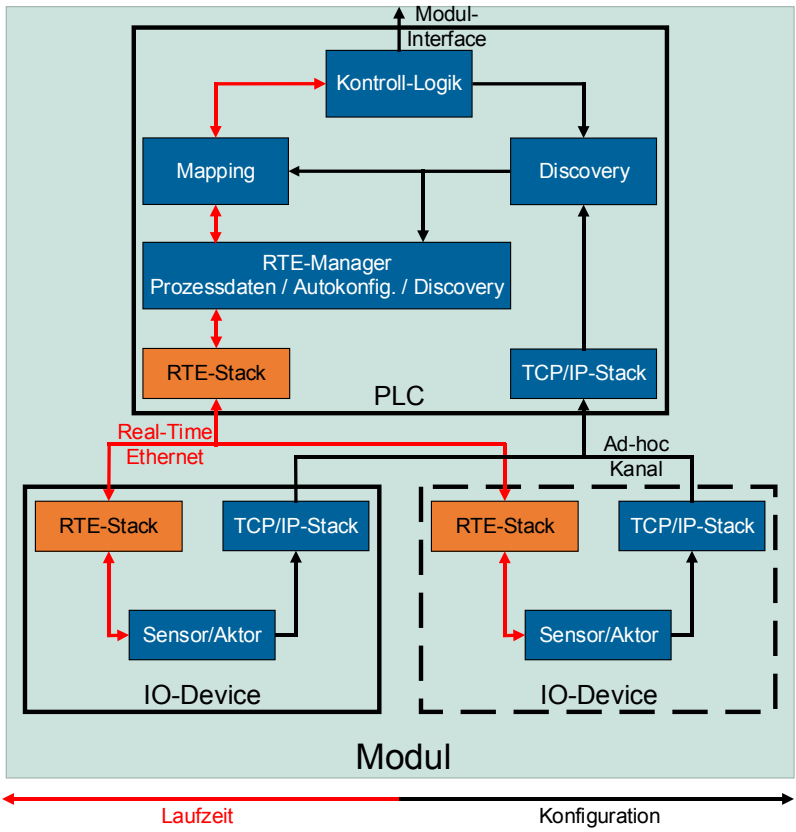

Abbildung 1 Architektur eines rekonfigurierbaren Moduls

3 Universelle RTE-Architektur

In diesem Abschnitt wird eine modulare Basisarchitektur vorgestellt, die eine dynamische Rekonfiguration des gesamten RTE-Stacks ermöglicht. Die Architektur kann zum Aufbau, sowohl von Feldgeräten als auch Steuerungen, genutzt werden. Abschnitt 3.2 geht auf die Gruppierung von RTE-Protokollen ein. In Abschnitt 3.3 werden bestehende Architekturen vorgestellt, die bereits einen Austausch von RTE-Protokollen erlauben. Abschnitt 3.3 stellt anschließend die in dieser Arbeit verwendete Systemarchitektur vor.

3.1 Kategorisierung von RTE-Protokollen

Grundsätzlich lassen sich RTE-Protokolle in drei verschiedene Kategorien (A/B/C) unterteilen [Ja05]. Die mit weicher Echtzeit bezeichneten RTEs der Kategorie A setzen direkt auf dem TCP/IP-Protokoll auf und definieren Echtzeitprotokolle in der Anwendungsschicht. Ein klassischer Vertreter ist beispielsweise Modbus-TCP [Mos06]. RTEs der Kategorie B werden ebenfalls oberhalb von Standard-Ethernet [IEE11] betrieben, sehen allerdings einen Echtzeitkanal, vorbei am TCP/IP Protokollstapel, oberhalb der MAC/LCC Schicht vor (z.B. PROFINET IO RT). Kategorie C stellt den aktuellen Stand und die höchste Leistungsklasse dar. In diese Klasse, mit sogenannten harten Echtzeitanforderungen, fallen unter anderem PROFINET IO IRT [Po10] und EtherCAT [Eth10]. Durch Modifizierungen der MAC-Schicht von Standard-Ethernet werden Zykluszeiten unterhalb von 100 µs unterstützt. Typischerweise werden hierfür ein zeitschlitzbasierter Kanalzugriff (TDMA) mit Bandbreitenreservierung und eine hochpräzise Zeitsynchronisation der Teilnehmer verwendet. Die Implementierung dieser RTEs erforderte bisher spezifische Hardwarekomponenten.

3.2 Stand der Technik

Neben dedizierten Protokoll-ICs, wie Systeme von Siemens [Si15] und Phoenix Contact [KW13], die Hardware für jeweils ein festes RTE-Protokoll der Kategorie C bieten, gibt es bereits Multiprotokoll-ICs. Diese verfügen über programmierbare Echtzeiteinheiten, die einen Austausch der unteren Protokollschichten (MAC-Schicht) ermöglichen und somit mehrere RTE-Standards der Kategorie C unterstützen. Gute Beispiele dafür sind die netX ICs der Fa. Hilscher [Hi08] und die AM335x Serie der Fa. Texas Instruments [Te15]. Auch diese Multiprotokoll-ICs stellen einen interessanten Ansatz für eine universelle RTE-Plattform dar. Zum aktuellen Zeitpunkt sind diese SoCs durch ihre geringere Leistungsfähigkeit jedoch nur eingeschränkt in der Lage, auch Zykluszeiten unter 100 µs zu bedienen. Kommende Generationen derartiger ICs können hierzu jedoch im Stande sein und könnten ebenfalls eine geeignete Plattform für eine universelle RTE-Architektur bieten.

Eine leistungsstärkere und flexiblere Plattform bieten FPGAs. Einzelprotokoll-Lösungen werden bereits von Unternehmen wie Softing und Beckhoff zusammen mit den beiden großen FPGA-Herstellern Altera [Al15] und Xilinx [Xi15] angeboten. Die neuesten FPGA-Generationen (Altera-5 Serie, Xilinx-7 Serie) bieten SOC-Systeme an, die neben der FPGA-Logik auch einen leistungsstarken ARM Cortex-A9 Dualcore enthalten, sodass die höheren Protokollschichten und Anwendungen hier ausgeführt werden können. In den Publikationen [GP06] und [GBP08] konnte nachgewiesen werden, dass die rekonfigurierbare FPGA-Logik für die harten Echtzeitanforderungen in den untersten Protokollschichten geeignet ist. Aufgrund ihrer hohen Leistungsfähigkeit der FPGA-Logik mit der kombinierten Flexibilität der SOC-Prozessoren erweisen sich SOC-FPGAs als ideale Zielarchitektur für die Anforderungen an eine universelle RTE-Architektur.

3.3 Systemarchitektur

Die im Folgenden vorgestellte Systemarchitektur bietet eine modulare Basisarchitektur für eine universelle RTE-Plattform, die ohne manuelle Konfiguration in unterschiedlichen RTE-Netzwerken eingesetzt werden kann. Als Hardwarebaustein findet hier ein moderner SoC Verwendung, der sowohl einen dedizierten Prozessor als auch FPGA-Logik integriert hat. Die FPGA-Logik bietet sich für die Umsetzung der MAC-Schicht des RTE-Protokolls an. Dies kann bei Protokollen der Kategorie A und B eine Standard-Ethernet MAC sein und bei Protokollen der Kategorie C die entsprechende echtzeitkritische MAC. Hierbei ist zu beachten, dass die verschiedenen RTE-Protokolle der Kategorie C unterschiedliche Hardwarerealisierungen erfordern. Die Vielfalt der Hardwareanforderungen wird durch die Rekonfiguration der FPGA-Logik gelöst.

Die Systemarchitektur ist in Abbildung 2 zu sehen und setzt sich aus zwei wesentlichen Teilkomponenten zusammen. Dies ist zum einen die spezifische Anwendung bzw. Funktionalität des Geräts (orange) und zum anderen der RTE-Stack (blau). Die Anwendung kann neben einer reinen Softwarelösung, die auf dem Prozessorsystem ausgeführt wird, zur Beschleunigung ebenfalls Teile der FPGA-Logik nutzen. So können für harte Echtzeitanwendungen Teile der Anwendung in Hardware abgebildet und auf dem FPGA ausgelagert werden [Ko13]. Analog zum Anwendungsfall der Rekonfiguration des RTE-Stacks, ist es mit Hilfe von partieller Rekonfiguration des FPGAs auch möglich verschiedene Teile der Anwendung auszutauschen, während die Kommunikation aufrechterhalten wird [Koe11]. Auf diesem Weg ist beispielsweise ein Firmware-Update der Anwendung während des Betriebs denkbar. Auch dieser Ansatz bietet daher schon eine eingeschränkte Möglichkeit ein Modul zu rekonfigurieren. Für eine vollständige Integration in die SOA aus [Du14] müssten semantische Beschreibungen und Dienste hinzugefügt und bereitgestellt werden (z.B. via OPC-UA).

Prototypisch wurde der Austausch von RTE-Protokollen der Kategorie C für die Standards EtherCAT und PROFINET IO (Conformance Class 3) gezeigt [Wa14]. Die spezifische MAC-Schicht wurde hierzu durch die FPGA-Logik in Hardware abgebildet. Die Konfiguration der gesamten FPGA-Logik kann durch den Prozessor während der Laufzeit durchgeführt werden. Zusätzlich wird der entsprechende SW-Stack des RTE-Protokolls auf dem Prozessorsystem ausgeführt.

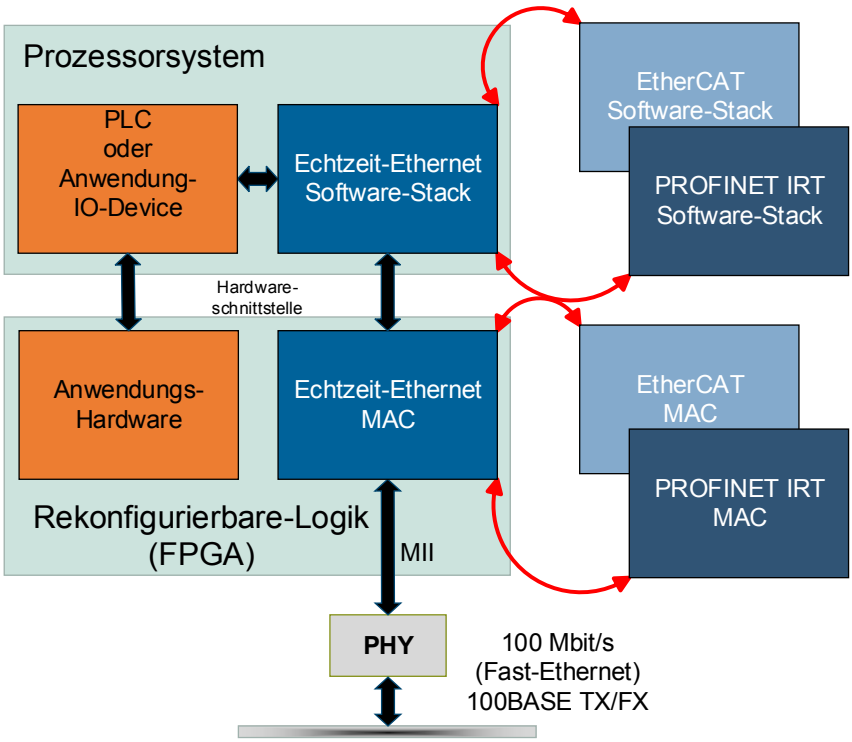

Abbildung 2 Systemarchitektur der universellen RTE-Plattform

4 Protokolldetektion von RTE-Protokollen

Um eine schnelle Integration der universellen RTE-Plattform in ein Modul zu ermöglichen, ist eine automatische Anpassung der Feldgeräte an das entsprechende RTE-Protokoll eines Moduls hilfreich. Dazu muss das verwendete Protokoll selbstständig erkannt werden.

4.1 Prinzip der Protokolldetektion

Für die Identifikation des Protokolls ist eine Auswertung der Ethernet Telegramme nötig, auf die in diesem Kapitel eingegangen wird. Der Aufbau der Telegramme ist in Abbildung 3 am Beispiel von Standard Ethernet, sowie der RTE-Standards PROFINET IO und EtherCAT dargestellt.

Es besteht aus 5 Segmenten, der Zieladresse, der Quelladresse, dem EtherType, den Nutzdaten und einem abschließenden CRC. Optional befindet sich ein VLAN TAG

zwischen Quelladresse und EtherType. Bei RTEs wird dieser Tag zur Priorisierung verwendet. Der EtherType beschreibt eine eindeutige 16 Bit ID. Anhand dieser ID kann zwischen verschiedenen Protokollen unterschieden werden. Neben Protokollen aus der Vermittlungsschicht (z.B. ARP, IPv4, IPV6, etc.) haben auch alle gängigen RTE-Standards eine eigene ID, so dass hier die eindeutige Protokollerkennung angesetzt wird. Eine vollständige Liste der EtherTypes ist unter [IEE15] zu finden.

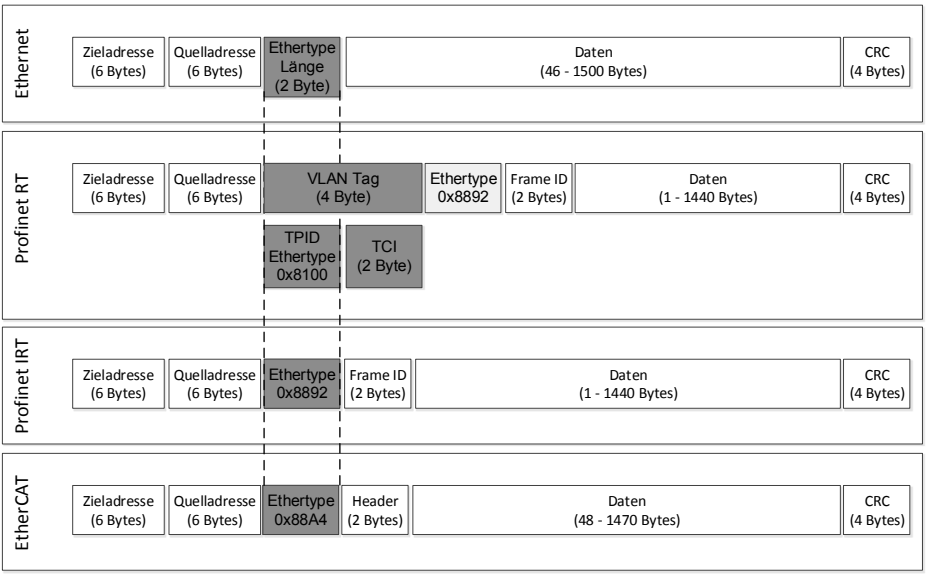

Abbildung 3 Vergleich der Telegramme von Standard Ethernet, PROFINET IO RT/IRT und EtherCAT

Nach Hinzufügen in ein Netzwerk wird der Datenverkehr analysiert und anhand des EtherTypes das Protokoll detektiert. Anschließend wird der entsprechende RTE-Stack geladen. Dies beinhaltet auch die Rekonfiguration des FPGAs mit der spezifischen MAC.

4.2 Prototypische Umsetzung

Die prototypische Umsetzung der Protokolldetektion wurde am Beispiel von EtherCAT und PROFINET IO durchgeführt. Beide Standards wurden auf dem Altera Cyclone V SX C6 SoC [Al15] abgebildet. Dieses System bietet neben einem FPGA-Bereich mit konfigurierbaren Logikelementen ein integriertes Prozessorsystem (HPS), bestehend aus zwei ARM Cortex-A9 Prozessoren.

Die Konfiguration des FPGAs erfolgt über einen Konfigurationsspeicher, auf den sowohl über externe Schnittstellen als auch über die ARM-CPU zugegriffen werden kann. Um Speicherplatz für die Unterstützung vieler verschiedener RTE-Standards einzuspa-

ren, können mit Hilfe einer partiellen Rekonfiguration auch nur Ausschnitte des FPGAs während der Laufzeit um konfiguriert werden [Wa14].

Die Protokollerkennung wird direkt bei der Systeminitialisierung durchgeführt. Dies ermöglicht eine automatische Konfiguration ohne manuelles eingreifen, nachdem die Netzwerkkomponente in das entsprechende Netzwerk hinzugefügt wurde. Abbildung 4 zeigt den Ablaufplan.

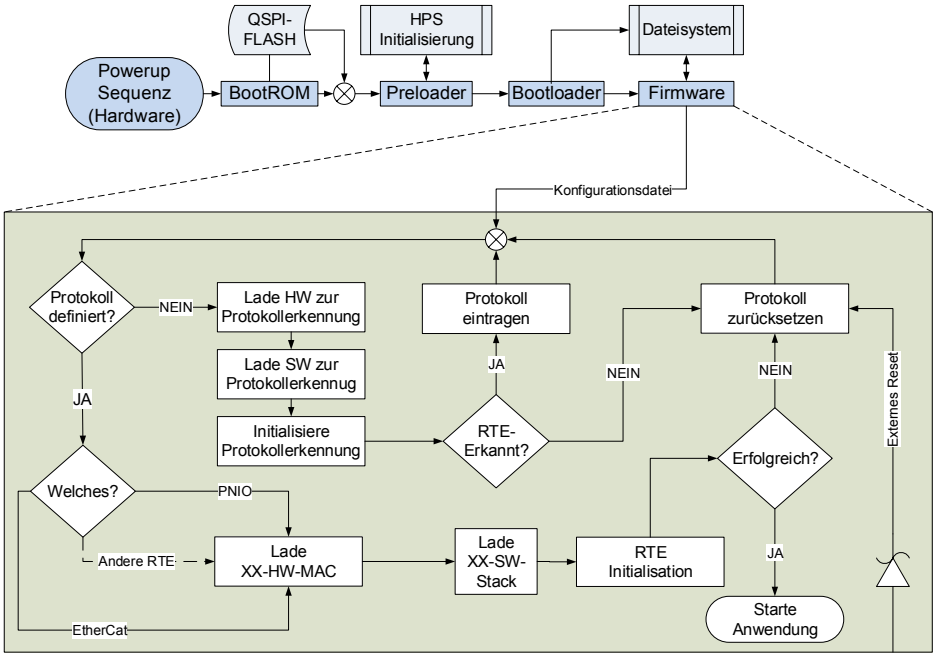

Abbildung 4: Ablauf der Systeminitialisierung mit Protokolldetektion

Nach dem Hochfahren des Systems wird aus dem BootROM heraus der Preloader gestartet, welcher sich im externen Speicher (QSPI-Flash) befindet. Im QSPI-Flash befinden sich neben dem Preloader auch der Bootloader, die zu startende Firmware (FW) sowie ein Dateisystem. Im Dateisystem liegen HW-Images der PROFINET IO und Ether-CAT Cores, sowie persistent speicherbare Konfigurationsdaten. Nachdem der Preloader das Prozessorsystem (HPS) initialisiert hat, wird der Bootloader gestartet, welcher wiederum die HPS FW startet. Die Initialisierung des FPGAs und das Laden der entsprechenden SW-Stacks werden von der FW durchgeführt.

Nach einem Start liest die FW die gespeicherten Konfigurationsdaten und wertet aus, ob bereits ein RTE-Protokoll konfiguriert ist. Ist dies der Fall, wird dessen HW-Image in den FPGA geladen und der passende SW-Stack samt Anwendung gestartet. Ist noch kein Protokoll konfiguriert wird ein spezielles HW-Image in den FPGA geladen, das zur Detektion des RTE-Protokolls dient. Hierbei handelt es sich um einen normalen Ether-

net-Switch, der zusätzlich Informationen über die EtherTypes von Telegrammen, die den Switch durchlaufen, an die FW weiterleitet. Detektiert die FW danach EtherTypes eines RTE-Protokolls, in diesem Fall 0x8892 für PROFINET IO oder 0x88A4 für EtherCAT, werden das HW-Image, sowie SW-Stack samt Anwendung für das detektierte Protokoll geladen. Eine weitere Protokolldetektion im laufenden Betrieb ist in der aktuellen Ausbaustufe nicht geplant.

Wie sich das Gerät im Falle eines Neustarts verhalten soll ist konfigurierbar. Es kann eingestellt werden, dass die Protokolldetektion nach jedem Neustart gestartet werden soll, oder dass das erste detektierte Protokoll auch nach Neustarts sofort angewendet wird. Ferner ist es möglich ganz ohne Protokolldetektion das RTE-Protokoll vorzugeben. Diese Einstellungen können jederzeit zurückgesetzt oder modifiziert werden.

5 Zusammenfassung und Ausblick

Im Rahmen dieser Arbeit wurde die Architektur einer universellen RTE-Plattform vorgestellt. Hiermit können Hersteller von Komponenten unabhängig vom RTE-Protokoll Feldgeräte oder auch Steuerungen entwickeln. Anlagen- und Maschinenbauer können mit diesen Komponenten Module nach [Du14] für unterschiedliche RTEs aufbauen und schneller neue Märkte schneller erschließen.

Insbesondere die gezeigte Protokolldetektion und automatische Netzwerkkonfiguration eignen sich für eine Integration in serviceorientierte und selbstkonfigurierende Automatisierungssysteme. Bestehende „Plug and Produce“ Ansätze können hierdurch sinnvoll erweitert werden.

Die vorgestellten Konzepte sind für beliebige RTE-Protokolle anwendbar. Validiert wurden Sie am Beispiel von EtherCAT und PROFINET IO.

Für eine vollständige Integration in Service Orientierte Architekturen werden zukünftig semantische Beschreibungen und die Abbildung von Diensten oberhalb der Eigenschaften der RTE-Protokolle betrachtet. Erste Ansätze sind in [Du14] beschrieben. Die Standardisierung semantischer Beschreibungen ist aktuell Gegenstand von Industrie 4.0 Arbeitskreisen.

6 Danksagungen

Dieses Forschungs- und Entwicklungsprojekt wird mit Mitteln des Bundesministeriums für Bildung und Forschung (BMBF) im Rahmen des Spitzenclusters “Intelligente Technische Systeme OstWestfalenLippe“ (it’s OWL) gefördert und vom Projektträger Karlsruhe (PTKA) betreut. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

Literaturverzeichnis

[Al15] Altera: Industrial Automation. 2015. Online: https://www.altera.com/solutions/industry/industrial/applications/automation/industrial-networking.html. [Zugriff am 18 03 2015].

[Al15] Altera: SoC FPGA ARM Cortex-A9 MPCore Processor Advance Information Brief. 2012. Online: https://www.altera.co.jp/ja_JP/pdfs/literature/hb/soc-fpga/aib-01020-soc-fpga-cortex-a9-processor.pdf. [Zugriff am 18 03 2015].

[AM14] AMA Fachverband für Sensorik e.V.: „Sensor-Trends 2014", http://www.ama-sensorik.de/verband/publikationen/, 2014

[BR15] B&R: „Auf der Überholspur mit reACTION Technology“, http://www.brautomation.com/de/produkte/innovations2015/ultraschnellereaktionen/, 13.01.2015

[Du14] Durkop, L., Trsek, H., Otto, J., & Jasperneite, J.: A field level architecture for reconfigurable real-time automation systems. In Factory Communication Systems (WFCS), 2014 10th IEEE Workshop on (pp. 1-10). IEEE.

[Eth10] EtherCAT Technology Group: EtheCAT Communication. 2010.

[GBP08] Griese, B.; Brinkmann, A.; Porrmann, M.: SelfS – A Real-Time Protocol for Virtual Ring Topologies.16th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS'08), Miami, Florida, USA, 2008.

[GI08] Gesellschaft für Informatik (GI): 38. Jahrestagung „INFORMATIK 2008 – Beherrschbare Systeme – dank Informatik (http://www.informatik2008.de/371.html)

[Gi14] Girbea, A.; Suciu, C.; Nechifor, S.; Sisak, F.: Design and Implementation of a Service-Oriented Architecture for the Optimization of Industrial Applications. IEEE Transactions on Industrial Informatics, vol. 10, pp. 185–196, 2014.

[GP06] Griese, B.; Porrmann, M.: A Reconfigurable Ethernet Switch for Self-Optimizing Communication Systems. IFIP Conference on Biologically Inspired Cooperative Computing (BICC 2006), pp. 115-124, Santiago de Chile, Chile, 2006.

[Hi08] Hilscher: Technical Data Reference Guide netX 500/100 Edition 1.3. 2008.

[IEE11] IEEE: IEEE Standard for Information Technology – Telecommunications and Informationexchange between Systems – Local and Metropolitan Area Networks – Specific Requirements: Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method, 2011.

[IEE15] IEEE Online: http://standards.ieee.org/develop/regauth/ethertype/eth.txt. [Zugriff am 16.3.2015].

[Ja05] Jasperneite, J.: Echtzeit-Ethernet im Überblick. *Automatisierungstechnische Praxis,* pp. 29-34, März 2005.

[Ko06] Koren, Y.: “General RMS Characteristics. Comparison with Dedicated and Flexible Systems”, in Reconfigurable Manufacturing Systems and Transformable Factories. pp. 27–45. Springer, 2006.

[Ko13] Korf, S; Sievers, G.; Ax, J.; Cozzi D.; Jungeblut, J.; Hagemeyer, J.;Porrmann, M. Rückert, U.: Dynamisch rekonfigurierbare Hardware als Basistechnologie für intelligente technische Systeme. Wissenschaftsforum 2013 Intelligente Technische Systeme, 2013.

[Koe11] Koester, M.; Luk, W.; Hagemeyer, J.; Porrmann M.; Rückert, U.: Design Optimizations for Tiled Partially Reconfigurable Systems. Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, pp. 1048-1061, 2011.

[KW13] KW-Software GmbH: TPS-1 Single Chip Interface Solution for PROFINET Devices Version 1.2.9. 2013.

[Lo13] Loskyll, M.: Entwicklung einer Methodik zur dynamischen kontextbasierten Orchestrierung semantischer Feldgerätefunktionalitäten, Technische Universität Kaiserslautern, 2013, Dissertation.

[Mos06] Modbus-IDA: MODBUS Messaging on TCP/IP Implementation Guide. 2006.

[Po10] Popp, M.: Das PROFINET IO-Buch. ISBN 978-3-8007-3274-6, VDE Verlag, 2010.

[Si10] Siemens AG: ERTEC 400 - Enhanced Real-Time Ethernet Controller Version 1.2.2. 2010.

[Te15] Texas Instruments: PROFINET on TI's Sitara processors. 2015

[Wa14] Walter, M.; Ax, J.; Buda, A. Nußbaum, K.; Hartfiel, J.; Jungeblut T.; Mario, P.: Dynamische Rekonfiguration von Echtzeit-Ethernet-Standards mit harten Echtzeitanforderungen. KommA 2014, Lemgo, Deutschland, 2014.

[Xi15] Xilinx: Industrial Automation. 2015. Online: http://www.xilinx.com/applications/industrial/industrial-networking.html. [Zugriff am 18 03 2015].

[Zue10] Zuehlke, D.: SmartFactory – Towards a factory of things. In: Annular Reviews in Control 34 (2010). S. 129-138, Elsevier Ltd., 28.03.2010