Entwicklung von Feldbusknoten für den wissenschaftlichen Gerätebau

Peter Kaever, Helmholtz-Zentrum Dresden-Rossendorf; März 2011

Abstract: *Mit der Entwicklung spezialisierter Feldbusknoten lässt sich eine für den wissenschaftlichen Gerätebau in vielen Fällen sinnvolle Aufteilung zwischen selbst entwickelten und kommerziell verfügbaren Komponenten unter konsequenter Nutzung vorhandener Automatisierungssysteme erreichen. Im Folgenden wird vorgestellt, welche Schritte dabei erforderlich sind und welche Möglichkeiten sich mit diesem Ansatz ergeben.*

Die Verwendung industriell verfügbarer Feldbusknoten ist fester Bestandteil von Anlagen, die im wissenschaftlichen Gerätebau verwendet werden. Speziell für leistungsfähige Systeme werden in unterschiedlichen Forschungszentren seit mehreren Jahren Ethernet-basierte Feldbussysteme eingesetzt. Für spezielle Anforderungen im Bereich der Ein- und Ausgabefunktionen, bei schnellen Prozessen oder leistungsfähigen Verarbeitungsfunktionen bzw. speziellen Schnittstellen ist die Entwicklung spezialisierter Geräte unumgänglich.

Im Helmholtz Zentrum Dresden Rossendorf sind industrielle Komponenten in einer Vielzahl von Anlagen unterschiedlicher Größe anzutreffen. Beispielsweise wurde der Beschleuniger ELBE unter Verwendung eines industriellen SPS-Systems aufgebaut. Die Realisierung der Flüssigmetallanlage LIMCAST oder der Steuerung für das Hochfeldlabor erforderten gleichfalls sichere und nicht sichere Komponenten, bei denen der Zugriff auf industriell erprobte und abgenommene Systeme einen großen Geschwindigkeitsvorteil bot. Im Folgenden soll ein Ansatz vorgestellt werden, bei dem auch spezialisierte Endgeräte direkt in die Automatisierungssysteme eingebunden werden.

Während der Schwerpunkt der Entwicklungstätigkeit in wissenschaftlichen Einrichtungen oftmals auf der Bereitstellung der spezifischen Funktionalität liegt, soll hier zunächst der Produktlebenszyklus beleuchtet werden. Die Integration spezialisierter Geräte in vorhandene Experimente stellt einen nicht zu unterschätzenden Aufwand dar, wenn man die Funktionalität betrachtet, die während der Laufzeit der Experimente nach und nach entsteht.
Bei systemkonformer Realisierung spezifischer Geräte kann mit der Projektierung beginnend über den Betrieb bis zu Systempflege mit Wartung und Diagnose der komplette Lebenszyklus des Gerätes bedient werden. Unter der Voraussetzung, dass die Hersteller der Automatisierungssysteme diese kontinuierlich weiterentwickeln und pflegen ist damit eine langfristige Basis für eine funktionierende Experimentalsystemautomatisierung gelegt. Betrachtet man die Verfügbarkeit industrieller Lösungen in der Automatisierungsbranche, so lässt sich sowohl aus deren Kundenstamm als auch aus dem Agieren am Markt ein konservatives Verhalten ablesen, so dass man auch künftig von einer hohen Verfügbarkeit ausgehen kann. Ethernetbasierte Lösungen bieten einerseits den Vorteil eines modernen Entwicklungsstandes, einer großen Anzahl bereits verfügbarer Komponenten und einer guten Performance.
Beispielhaft soll im Folgenden der Fokus auf der Entwicklung von EtherCAT Slave Devices liegen, wobei abschließend ein Ansatz vorgeschlagen wird, der sich mit geringem Aufwand auf Profinet Automatisierungssysteme erweitern lässt.

Zur besseren Einordnung der vorgeschlagenen Lösung und zur Prüfung der Integrationsfähigkeit in Anlagenkonstellationen, wie sie bei der Experimentautomatisierung verwendet werden, wurde am Helmholtz-Zentrum Dresden-Rossendorf eine Zielarchitektur abgestimmt, deren wesentlichste Merkmale in Bild 1 zu sehen sind. In der Leitebene sind vor allem Elemente zur Steuerung des Gesamtsystems anzutreffen. Hierzu gehören graphische Bedienerschnittstellen, die beispielsweise mit WinCC, National Instruments oder mit selbst entwickelten Oberflächen realisiert sind.
Die Kommunikationsschicht dient dem Datenaustausch zwischen Leitebene und den Automatisierungskomponenten, wobei Komponenten wie OPC-Server zur übergreifenden Kommunikation und flexiblen Anbindung dedizierter Subsysteme eine wichtige Rolle einnehmen.

Automatisierungskomponenten bilden die direkte Schnittstelle zum Prozess ab und sind je nach gewünschter Reaktionsgeschwindigkeit, Sicherheitsanforderungen und Systemdesign als dezentrale Peripherie mit eigener Intelligenz oder als reine Ein-/Ausgabegeräte realisiert.

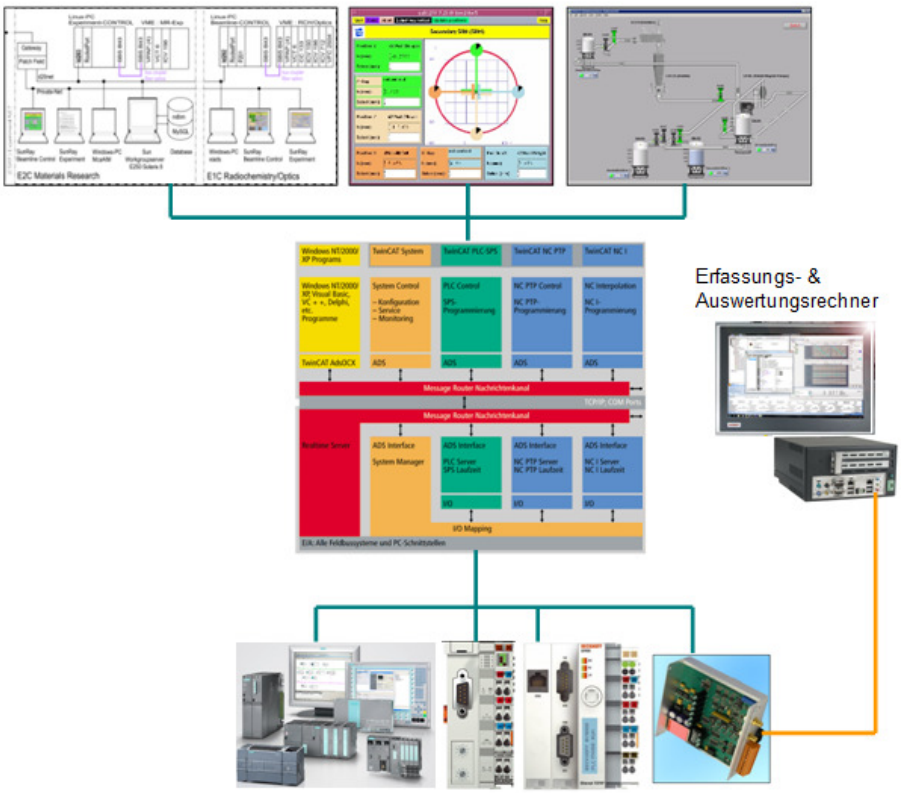

Bild 1 Zielarchitektur Automatisierung

Spezialisierte Endgeräte sind oft mit einem spezialisierten Frontend ausgestattet; in vielen Fällen ist eine zusätzlich eine leistungsstarke Vorverarbeitung der Daten erforderlich. Durch Anbindung an einen Feldbus sind die hier vorgeschlagenen Geräte automatisch vernetzbar, wodurch eine Modularität auf Geräteebene entsteht. Eine finale Messdatenverarbeitung bietet sich gemäß Bild 2 auf einem PC an, der mit dem spezialisierten Feldbusgerät beispielsweise über Ethernet oder PCIe in Verbindung steht. Auf diese Weise ist oft eine kostengünstige und leistungsfähige Verarbeitung möglich.

Kern des hier vorgestellten Realisierungsvorschlages ist die Entwicklung spezialisierter Feldbusknoten. Für eine exemplarische Realisierung wurde zunächst ein Evaluationsboard mit einem EtherCAT Feldbusknoten beschafft. Dieses ist mit einem 16 bit PIC Mikrocontroller ausgestattet, der durch einen leistungsfähigeren ARM9 zu ersetzen war [5]. Zudem waren die eingeschränkten Ein- und Ausgabemöglichkeiten so zu erweitern, dass digitale Ein- und Ausgänge als Bits oder Ports und 16 bit breite analoge Ein- und Ausgänge zur Verfügung gestellt wurden. In Bild 3 ist das erste Funktionsmuster zu sehen, das einen kommerziell verfügbaren EtherCAT Slave Controller sowie ein ARM9 Evaluationsboard nutzt.

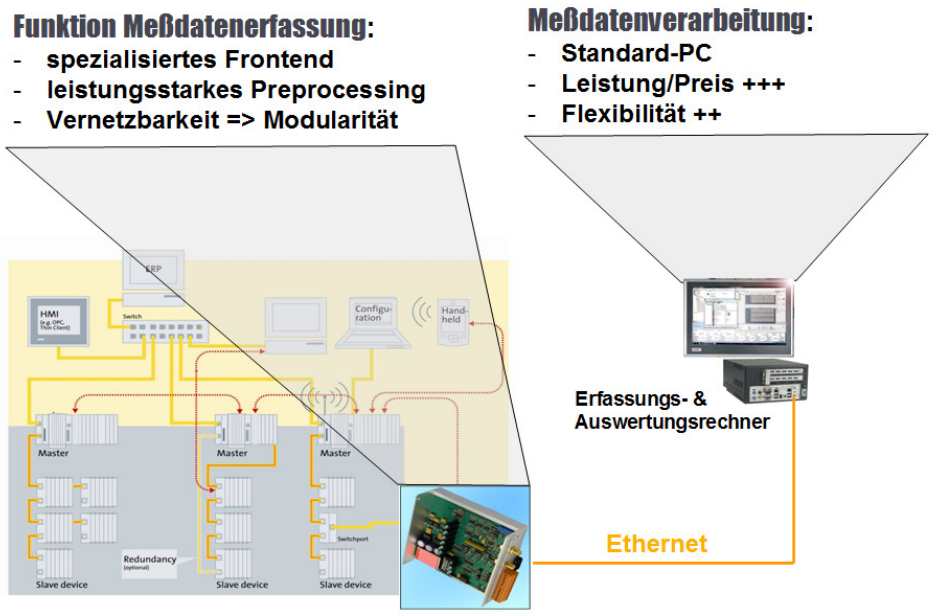

Bild 2 Funktionale Trennung zwischen Messdatenerfassung und -verarbeitung

Wichtigstes Ziel ist die Übertragung von Prozessdaten zwischen EtherCAT Master und dem hier entwickelten EtherCAT Slave Device. Zur Integration wurde ein Netzwerkstrang aufgebaut, in dem ein PC als EtherCAT Master fungiert [3]. Die Kommunikation in EtherCAT basiert auf dem Master-Slave Prinzip; die vom Master abgeschickten Frames mit den Prozessdaten durchlaufen das Slave Device sowohl auf dem Hinweg als auch auf dem Rückweg. Die Frames enthalten das komplette Prozessabbild mit allen PLC-Daten, I/O-Daten und NC-Daten, wobei das Slave Device beim Durchlaufen des EtherCAT Frames die ihm zugehörigen Daten entnimmt und im gleichen Durchlauf die ihm zugeordneten Eingabedaten in das Frame hineinschreibt.

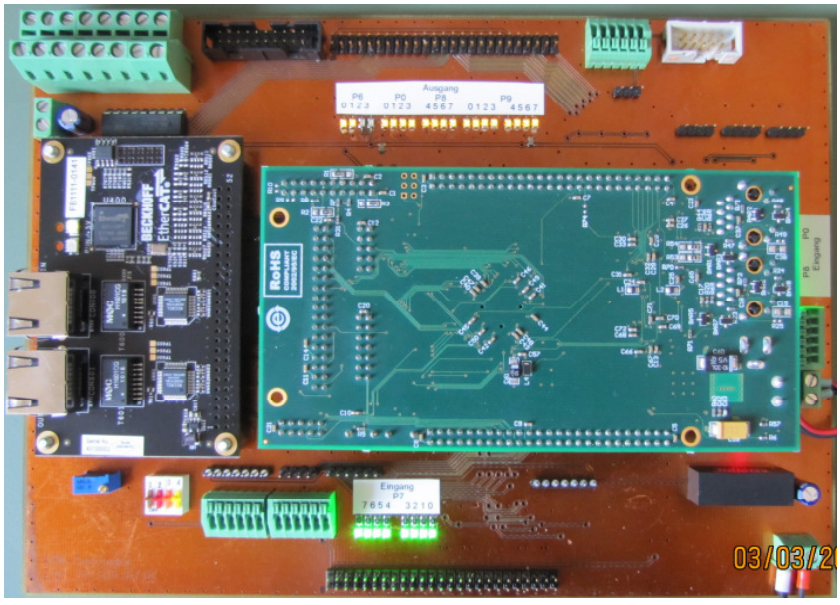

Bild 3 Funktionsmuster eines EtherCAT Slave Device mit ARM9 Mikrocontroller

Bei genauerer Betrachtung vollzieht der in jedem Slave Device vorhandene EtherCAT Slave Controller alle EtherCAT-Bustransfers. EtherCAT Slave Controller [2] stehen als ASIC's oder als IP-Cores für die Realisierung mit FPGA's zur Verfügung. Ihre Aufgabe ist die korrekte Handhabung der Busanschaltung bis zum Transfer der Datenpakete beispielsweise an einen Mikrocontroller. Zur Analyse der Ethercat Frames wurde Wireshark mit einem Plugin erweitert, das die Aufzeichnung

von EtherCAT Frames auf komfortable Weise ermöglicht. Die korrekte Einbindung des EtherCAT Slave Controllers am Bus läßt sich auf diese Weise im Netzwerk überprüfen.

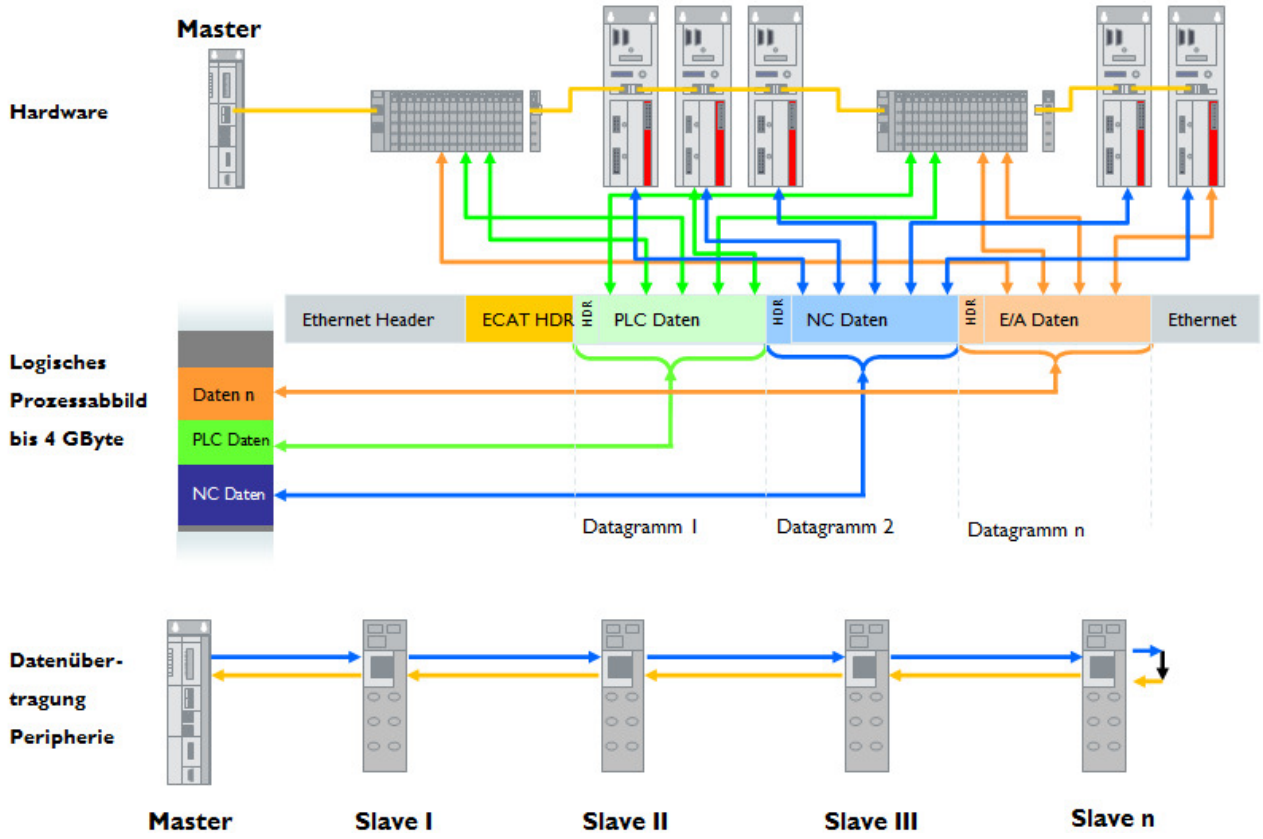

Bild 4: Systemkommunikation bei EtherCAT

Während das Frame die Kette der Slave Devices durchläuft, werden die aktuellen Ausgabewerte entnommen und aktuelle Eingabewerte eingefügt. In jedem Slave Device ist ein EtherCAT Slave Controller implementiert, der für eine Entnahme der an den jeweiligen Slave gerichteten Daten aus dem durchlaufenden Frame sorgt. Das Einfügen der vom Slave Device aus der Umgebung gelesenen Informationen geschieht ebenfalls autark durch den EtherCAT Slave Controller. Dieser stellt auch Signale zur Ansteuerung der Hardware zur Verfügung, die beispielsweise bei Anbindung eines Mikrocontrollers das Auslösen von Interrupts ermöglichen.

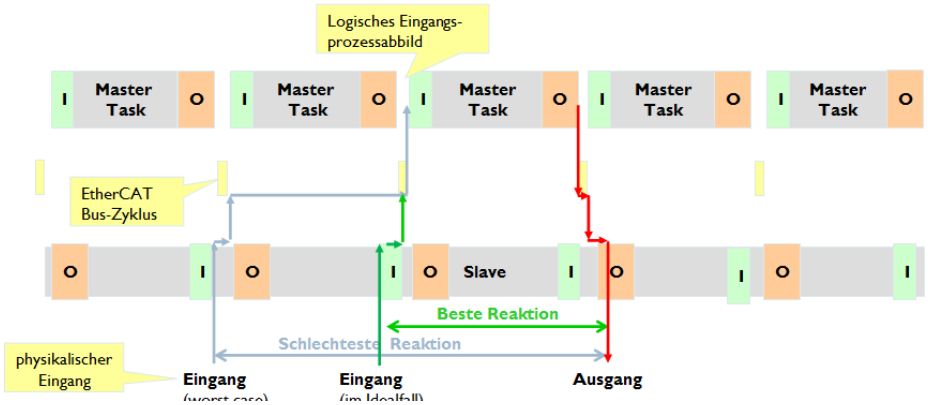

Bild 5: Zeitverhalten bei der Übertragung von Prozeßdatenframes

Beim zyklischen Versenden des EtherCAT-Datenpaketes durch den Master sind zunächst die Abläufe im EtherCAT Slave Controller zu beachten. In Bild 4 wurde ein linearer Strang betrachtet, der keine Verzweigungen oder Hierarchien beinhaltet. Die relevante Zeitdauer für Reaktionen auf einen physischen Eingang durch den Master kann bis zu 2 vollständigen Buszyklen betragen. Je nach Schaltzeitpunkt eines Ereignisses in zeitlicher Relation zum Bustransfer können wie in Bild 5 gezeigt von einem (best case) bis zu zwei Buszyklen (worst case) zwischen Einlesen des Eingangs, dessen Transfer an den Master und der dort stattfindenden Verarbeitung sowie der Übergabe des entsprechenden Ausgabekommandos an den Slave liegen.

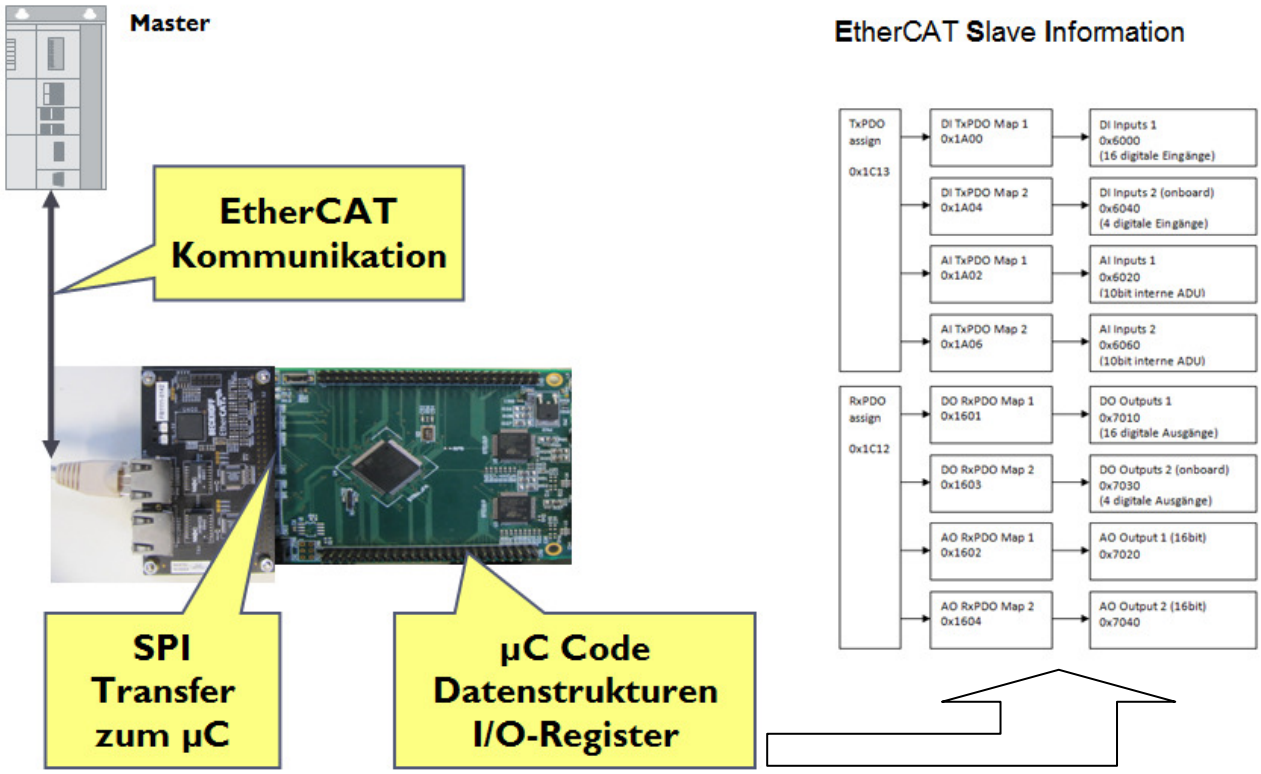

Bild 6: Anbindung des Mikrocontrollers und der Prozessdatenobjekte an EtherCAT

Industriell verfügbare EtherCAT Slave Controller bieten 8 /16 bit breite Schnittstellen bzw. ein 32 bit breites Parallelinterface sowie eine SPI Schnittstelle an, die bis zu 20 MHz betrieben werden kann [2]. Für die hier angestrebte Realisierung wurde eine Schnittstelle gewählt, die auf einer großen Zahl von Mikrocontrollern verfügbar ist und einen geringen Aufwand an Ressourcen benötigt. Die in Bild 6 gezeigte SPI-Schnittstelle hat den Vorteil eines geringen Ressourcenbedarfes auf der Seite des Mikrocontrollers. Nach Festlegung der Anbindung an den Mikrocontroller und Definition der zu realisierenden Schnittstellen ist die Definition der Strukturen zur Beschreibung der Prozessdatenobjekte der nächste logische Schritt. Da sich die Ein-/Ausgabefähigkeiten des Slave Device unmittelbar in diese Datenstrukturen abbilden und den Weg vom Mikrocontrollercode bis zur Verwendung im Master strukturieren, sollen die einzelnen Manifestationen dieser Strukturen im Folgenden erläutert werden.

Bild 7 legt einen Schwerpunkt auf die verschiedenen Ebenen, in denen die Ein- und Ausgabefähigkeiten des Slave Device verwendet werden. Auf Ebene der Mikrocontrollerhardware sind sie in der Regel als Register, im Mikrocontrollercode als Strukturen zum Transfer über die SPI vorhanden. Im EtherCAT Slave Controller werden lokale Abbilder dieser Strukturen für den Transfer in EtherCAT Frames bereitgestellt; im Master dienen Abbilder mit den entsprechend zugeordneten logischen Namen zur Implementierung der Steuerung.

Schematisch wurden für diese Strukturen für unser Beispiel in Bild 7 auf der rechten Seite dargestellt und realisieren in unserem Beispiel mehrere digitale und analoge Eingangs- und Ausgangskanäle.

Die busseitige Anbindung des Slave Device erfolgt dabei im Wesentlichen in 4 Schritten, die aufeinander abzustimmen sind [4]:
1. Programmierung des Mikrocontrollers
2. Beschreibung der Fähigkeiten des Slave Device in einer Konfigurationsdatei
3. Projektierung der auf dem Master implementierten Steuerung unter Nutzung des Slave Device
4. Schreiben der Konfiguration in den EtherCAT Slave Controller

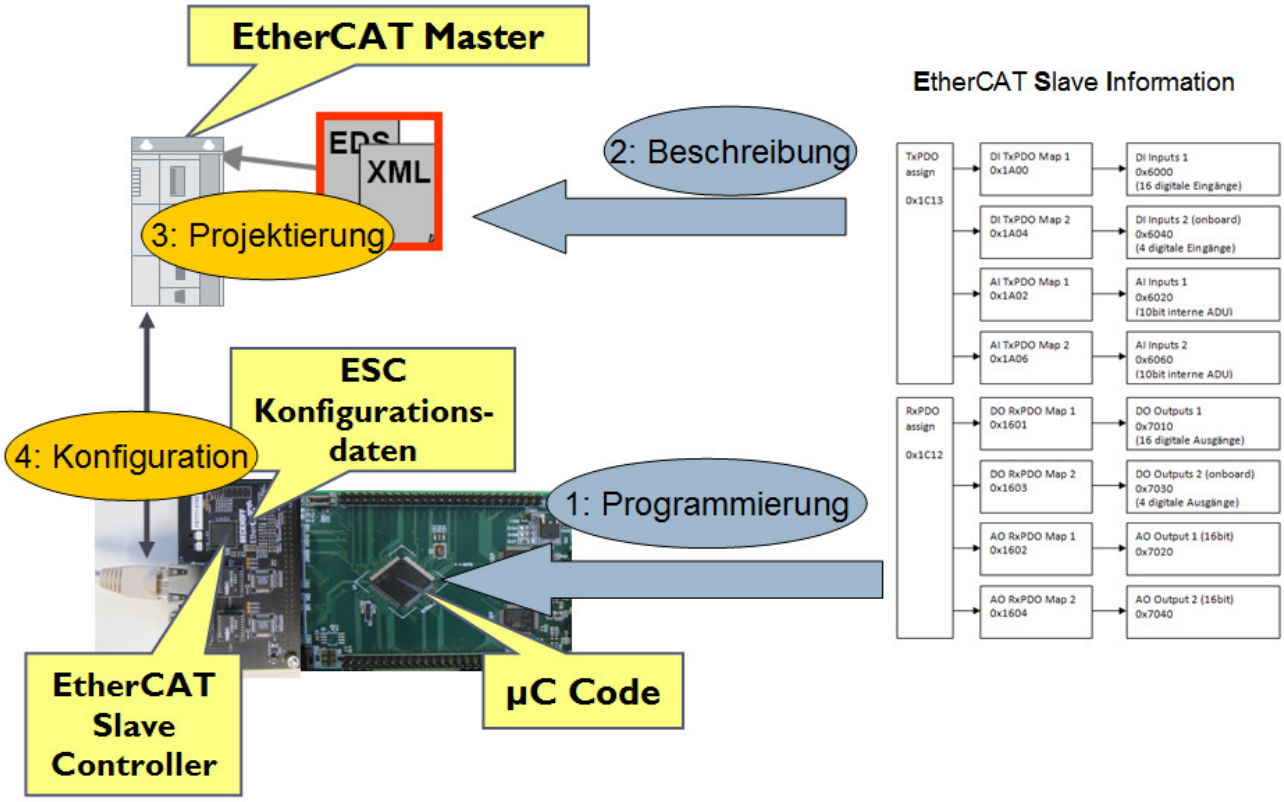

Bild 7: Transfer der Prozessdatenobjekte und Schritte der Einbindung eines Slave Device

Innerhalb des Mikrocontrollercodes werden für den ersten Schritt Listen von Zeigern auf die jeweiligen Prozessdatenobjekte angelegt. Die dazu gehörigen Strukturen werden an eine Funktion übergeben, die das Prozessdatenabbild über die SPI-Schnittstelle aus dem EtherCAT Slave Controller liest. In einem weiteren Schritt wird das Prozessdatenabbild mit den Input- und Outputkanälen des Mikrocontrollers abgeglichen. Der Schritt einer Beschreibung der Fähigkeiten des Slave Device dient der detaillierten Erfassung aller Fähigkeiten des Slave Device sowie dessen eindeutiger Typbezeichnung, deren Daten beispielhaft in Bild 14 dargestellt sind.
Die Projektierung verknüpft unter anderem die logischen Variablen der auf dem Master laufenden Steuerung mit den in der Peripherie vorhandenen Slave Devices auf Ebene der physikalisch vorhandene Kanäle. Eine inhaltliche Konsistenz zwischen Mikrocontrollerprogrammierung und der xml-Beschreibung des Slave Device ist zur korrekten Funktion der Konfiguration am Bus zwingend erforderlich.

Bild 8 zeigt einige der exemplarisch angelegten Ein- und Ausgabekanäle des im HZDR entwickelten Slave Device. Dabei ist zunächst erkennbar, dass 2 digitale Eingabekanäle, 2 analoge Eingabekanäle sowie jeweils 2 digitale und analoge Ausgabekanäle realisiert wurden. Diese sind mit ihren logischen Namen erkennbar. Weiter ist es möglich, eine Gliederung innerhalb eines Eingabewortes vorzusehen, in dem einzelne Bits mit ihren Zuordnungen an den Klemmen gleich im

TwinCAT System Manager zu erkennen sind. Dies kann ggf. die Projektierung vereinfachen. Im hier vorliegenden Beispiel wurde ein digitaler Ausgang mit einzelnen Bits ausgestattet, deren Kontroll-LED's als LED0 bis LED3 bezeichnet sind.
WcState und InfoData beinhalten Standarddaten für EtherCAT, welche beispielsweise die Gültigkeit der Daten kennzeichnen oder die Netzwerkkennung des Slave Device sowie seinen Betriebszustand beinhalten

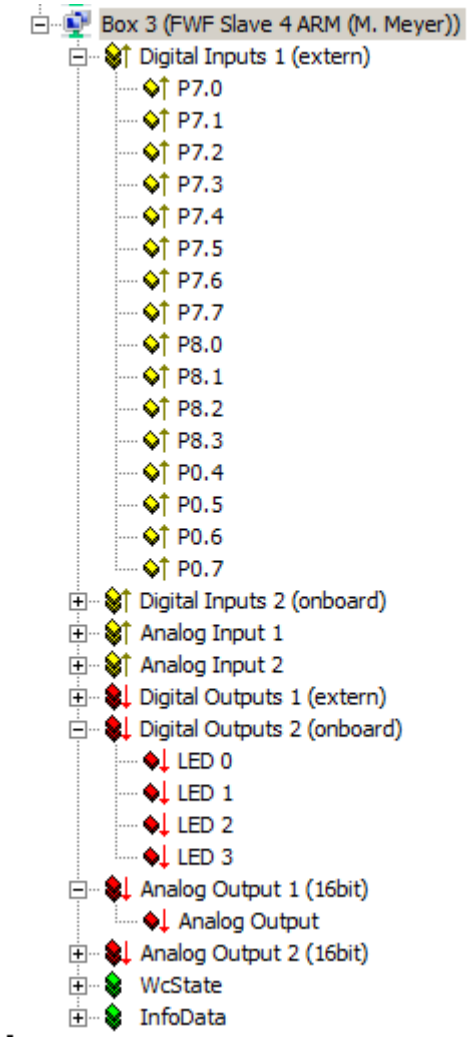

Bild 8: Konfiguration eines am HZDR entwickelten Slave Device im System Manager

In der Realisierung von Beckhoff wird der Transfer zwischen dem Prozessdatenabbild und den Registern des Mikrocontrollers in einem Interrupt durchgeführt, der vom EtherCAT Slave Controller über ein dediziertes Signal ausgelöst wird. Eigene Applikationen zur Verarbeitung von Prozessdaten synchron zum Buszyklus lassen sich an dieser Stelle ebenfalls günstig implementieren, wobei es sinnvoll sein kann, die Verarbeitung zeitlich hinter die Übertragung der Prozessdaten zu legen. Auch nach Abschluss der Interruptserviceroutine ist die Verarbeitung von Prozessdaten möglich, wobei darauf zu achten ist, dass diese asynchron zum Zeitraster des Buszyklus läuft und durch den Interrupt zum Transfer der SPI-Daten unterbrochen wird.

EtherCAT kann mit unterschiedlichen Buszyklen betrieben werden, die als „Sync Manager" und „Distributed Clock" bezeichnet werden. Im Betrieb mit Sync Manager versendet der Master in einem vorgegebenen zeitlichen Raster Frames, welche die Kette der Slaves durchlaufen und durch die Verarbeitungs- und Kommunikationsabläufe im Inneren der Slaves auch das zeitliche Verhalten im Gesamtprozess bestimmen. Zum möglichst zeitgleichen Schalten von Ausgängen auf unterschiedlichen Slave Devices wird dagegen die Betriebsart Distributed Clock verwendet. Im

Folgenden soll die prinzipiell unterschiedliche Funktionsweise beider Betriebsarten erläutert werden.

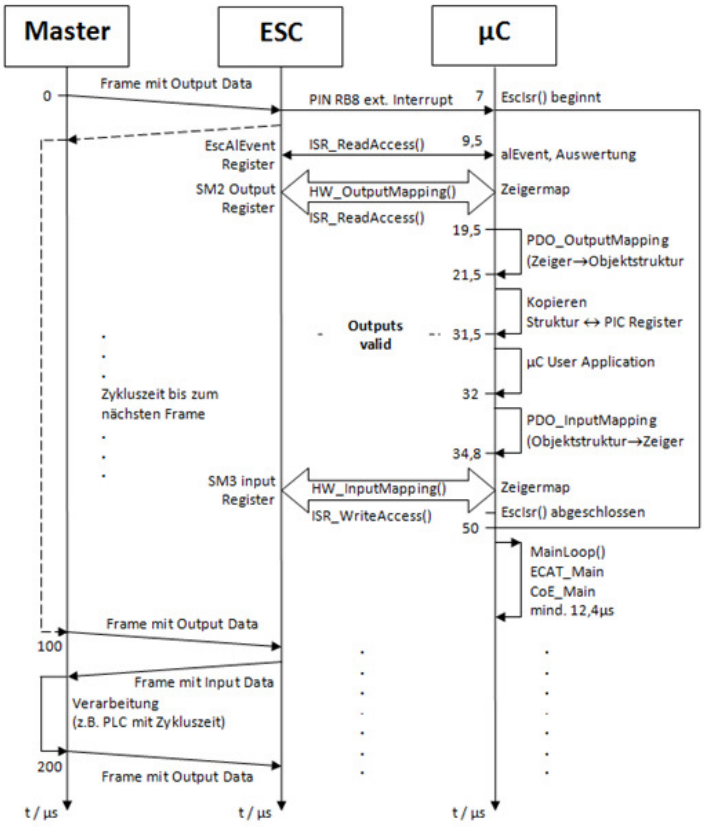

Bild 9: Aktivitäten im Slave Device bei Sync Manager Kommunikation

Die zeitliche Abfolge in der Bearbeitung von Buszyklen vom Transfer der Frames bis zur Reaktion des Mikrocontrollers ist im obigen Bild für den Betrieb mit dem Sync-Manager zu sehen. Nach Entnahme der Eingangsdaten des durch den Master verschickten Frames wird durch den EtherCAT Slave Controller (ESC) ein Hardwareinterrupt ausgelöst, welcher ein Lesen der Prozessdaten über die SPI Schnittstelle zur Folge hat (PDO_OutputMapping). Diese werden darauf an die zugehörigen Register des Mikrocontrollers weitergegeben. Nach Ausführung der lokal auf dem Mikrocontroller realisierten Anwendung werden die als Input der Master-Steuerung verwendeten Datenstrukturen in die Strukturen des Prozessdatenobjektes transferiert (PDO_InputMapping).

Eine Übergabe der Prozessdatenobjekte über die SPI-Schnittstelle schließt den Interrupt ab. Dabei ist zu beachten, dass der Abschluß der Interrupt Service Routine asynchron zum Bustransfer abläuft und in der Regel davon auszugehen ist, dass Input-Daten erst im nächsten Buszyklus transferiert werden können. Aufgabe des EtherCAT Slave Controllers ist an dieser Stelle eine Pufferung konsistenter Input-Prozessabbilder des Slave Device. Freie Zeit des Mikrocontrollers außerhalb der Interrupt Service Routine kann von einer weiteren Applikation verwendet werden, deren Aktivität im Oszilloskopausschnitt auf Bild10 zu erkennen ist.

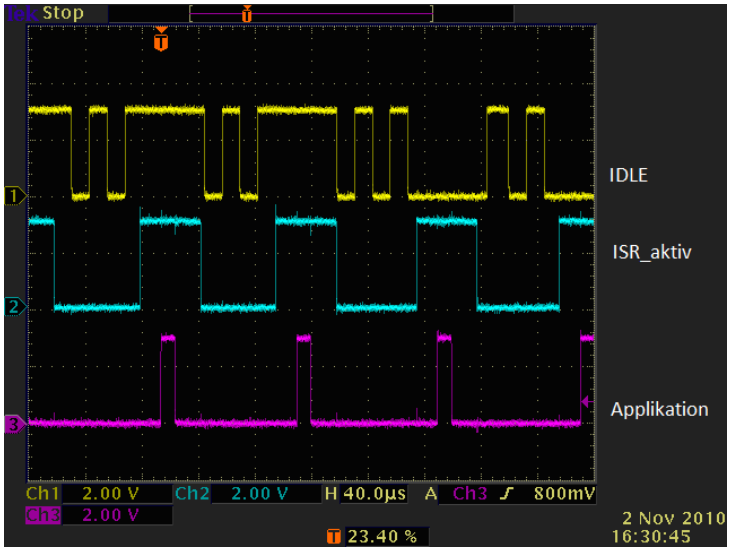

Bild 10: Aktivitäten im Mikrocontroller bei Sync Manager Kommunikation

Der Anspruch eines möglichst zeitgleichen Schaltens von Ausgängen führt zu einem modifizierten Ablauf im Zusammenspiel zwischen EtherCAT Slave Controller und Mikrocontroller.

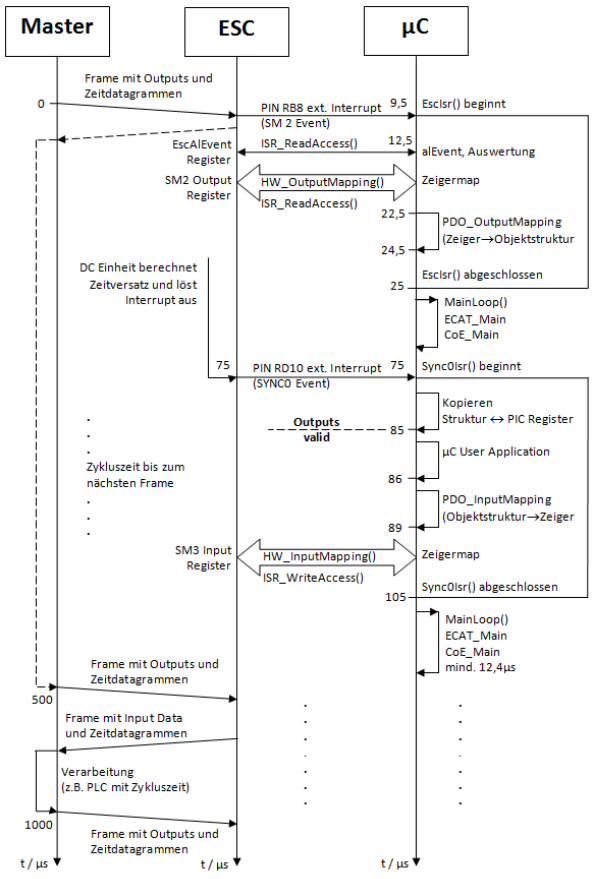

Bild 11: Aktivitäten im Slave Device bei Verwendung von Distributed Clocks

In der Betriebsart Distributed Clocks werden zunächst spezielle Telegramme zur Bestimmung der Verzögerungen und Laufgeschwindigkeit aller lokalen Uhren deren individuelle Parameter ermittelt.

Der Kerngedanke besteht nun darin, unter Nutzung einer vorgegebenen und auf jedes Slave Device lokal umgerechneten Verzögerung dafür zu sorgen, dass die gewünschten Ausgabezeitpunkte für die Ausgänge in der lokalen Zeit mit der Zeitbasis des Masters mit hoher Genauigkeit übereinstimmen. Als Genauigkeitsangabe werden Zeiten besser als 100 ns benannt, wobei Verzögerungen zwischen den gewünschten Transferzeitpunkten und den realen Ausgabezeitpunkten gesondert zu betrachten sind. Diese Funktionalität wird durch den EtherCAT Slave Controller erbracht.
Als Mechanismus zur möglichst schnellen und jitterfreien Ausgabe der Prozessdatenobjekte wird ein zweiter Interrupt genutzt. Die im ersten Interrupt übertragenen Prozessdaten werden bereits in der ersten Interruptserviceroutine in das Mikrocontroller-interne Prozessmodell geschrieben. Im zweiten Interrupt erfolgt lediglich der zeitkritische Transfer vom internen Prozessmodell an die Register des Mikrocontrollers. Latenz und Jitter des Datentransfers über die SPI-Schnittstelle fallen dabei nicht mehr ins Gewicht.

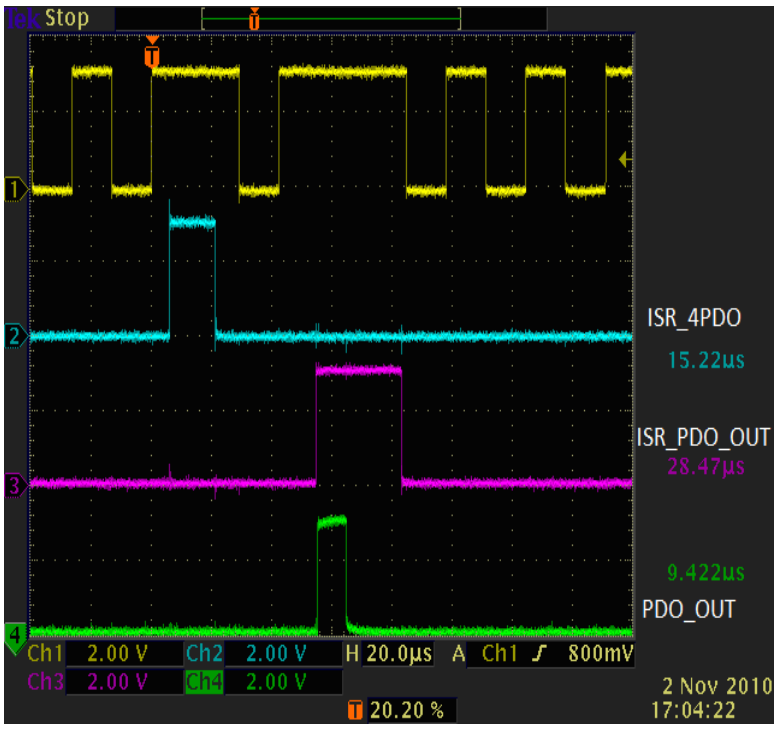

Bild 12: Aktivitäten im Mikrocontroller bei Verwendung von Distributed Clocks

In dieser Betriebsart ist die Zyklusdauer je nach Ausdehnung des Busstranges so einzustellen, dass die erforderlichen Zeiten für den Durchlauf des Frames und die Übernahme der Prozessdaten gesichert sind.
Naturgemäß lassen sich zwar Latenz und Jitter für die eigene Realisierung minimieren; eine Synchronisierung mit anderen Geräten, die ggf. auch andersartige Anbindungen zum EtherCAT Slave Controller realisieren, ist jedoch nur unter Einschränkungen und bei unbekannten Slave Devices nur durch empirische Einstellungen möglich.

Während die Übertragung von Prozessdaten die wichtigste Möglichkeit zur Nutzung eigener Slave Devices darstellt, bieten FTP-Protokolle und Mailboxen weitere Funktionen. Im Rahmen der hier vorgenommenen exemplarischen Realisierung wird darauf nicht eingegangen, da diese geringere Voraussetzungen an das Antwortverhalten des Slave Device stellen und ihre Komplexität hinter den Prozessdatenobjekten zurückbleibt.
Bei der Nutzung eines derartigen Slave Device lassen sich alle Funktionen des TwinCAT System Managers wie beispielsweise automatische Erkennung der am Bus angeschlossenen Devices, Fehlerzähler, Konfigurationsinformationen, etc. nutzen um das Device am Bus in Betrieb zu nehmen. Die Verbindung zu einem OPC Server ist nach logischer Verknüpfung mit Variablen eines SPS-Programmes ohne weiteres herstellbar.

Abschließend sei bemerkt, dass in der aktuellen Konfiguration die Ankunft von EtherCAT-Frames zur Übertragung der Prozessdaten die einzigen Ereignisse im System darstellen. Bei komplexeren Verarbeitungsfunktonen und ggf. darüber hinausgehender Funktionalität zur Übertragung von Massendaten wird die Beibehaltung einer klaren interne Strukturierung beispielsweise durch ein Betriebssystem ermöglicht.

Da am Helmholtz-Zentrum Dresden-Rossendorf auch Profinet Geräte entwickelt werden sollen, fiel die Wahl bei der Auswahl des Mikrocontrollers auf einen ARM9 in Kombination mit dem freien Betriebssystem ECOS. Von Siemens werden ERTEC-Chips zur Realisierung von Profinet Devices angeboten, die intern mit einem ARM946ES ausgestattet sind. ERTEC Chips sind für die Realisierung von Profinet Feldbusknoten bis zu Profinet IRT geeignet. Eine bestehende Portierung von ECOS für diesen ARM9 motiviert neben den zahlreichen Vorzügen von ECOS dessen Auswahl als Betriebssystem. Die Anbindung eines Ethernet-Anschlusses ist dabei in der Regel bereits vorgesehen und erlaubt perspektivisch die Übertragung von Massendaten auf einem separaten Kanal.

Ziel ist die Nutzung möglichst vieler gemeinsamer Systemkomponenten zur Realisierung von Feldbusendgeräten unter EtherCAT und Profinet für den wissenschaftlichen Gerätebau. Hierbei stellen ein ARM9 Core mit ECOS als Betriebssystem eine günstige Möglichkeit zur Nutzung gemeinsamer Elemente dar. Unterschiedlich sind hierbei die Hardware zur Feldbusanbindung und die feldbusspezifischen Strukturen zur Übertragung der Prozessdatenobjekte.
Die Realisierung von Feldbusknoten in EtherCAT und Profinet ist auf diese Weise unter Nutzung einer Fülle gemeinsamer Softwaremodule für Betriebssystem und Applikationen möglich, was Aufwand und Pflege positiv beeinflusst.

Mein Dank geht an dieser Stelle an die Kollegen in der Zentralabteilung Forschungstechnik für viele fruchtbare Diskussionen und an Herrn Markus Meyer, der im Rahmen eines studentischen Praktikums [1] maßgeblich an der Inbetriebnahme des Evaluationsboards bis zur umfangreichen Portierung des Codes auf den ARM9 mitgewirkt hat.

Quellen:

[1] **Meyer, M.** *Aufbau eines EtherCAT-Slave mit Beckhoff ASIC und Integration in TwinCAT;* Beleg zum praktischen Studiensemester Fachbereich Elektrotechnik der Hochschule für Technik und Wirtschaft Dresden (FH); 4.3.2011

[2] **BECKHOFF AUTOMATION GMBH**: *Hardware Data Sheet ET1100 - EtherCAT Slave Controller*. Mai 2010. Download am 04.01.2011 von www.beckhoff.com

[3] **BECKHOFF AUTOMATION GMBH**: *Beckhoff Information System 01/2011*. Januar 2011. Download am 27.01.2011 von infosys.beckhoff.com

[4] **EtherCAT TECHNOLOGY GROUP**: *EtherCAT Slave Information Specification*. Mai 2009. Download am 27.09.2010 von www.ethercat.org

[5] **STMicroelectronics**: *RM0006 Reference manual - STR91xFA ARM9 based microcontroller familiy*. Juli 2009. Download am 11.11.2010 von www.st.com