Kommunikationsgetriebene Hardware/Software-Partitionierung eines Netzwerkprotokollstacks auf einer SoC-Plattform

Kim Grüttner, Carsten Beth, Wolfgang Nebel

{kim.gruettner, carsten.beth}@offis.de, nebel@informatik.uni-oldenburg.de

Abstract: Dieses Paper beschreibt eine Untersuchung verschiedener HW/SW-Partitionierungen am Beispiel des LonTalk®-Protokollstacks. Ausgehend von einer maximalen Softwareimplementierung wird einerseits eine Verlagerung der Funktionalität von Soft- in Hardware, andererseits eine Veränderung der HW/SW-Schnittstelle und der Kommunikations-Architektur untersucht. Zur Evaluation wurde ein Altera Excalibur mit einem ARM9-Prozessorkern und integriertem FPGA benutzt.

1 Einleitung

Eingebettete Systeme beinhalten häufig kommunizierende Hard- und Softwarekomponenten. Für Partitionierungsentscheidungen eines Systems in diese Komponenten sind zum Teil konkurrierende Kriterien wie Ausführungsgeschwindigkeit (Performance), Flächenbedarf (Fläche) und Leistungsverbrauch zu bewerten und zu optimieren. Die HW/SW-Schnittstelle ergibt sich im Wesentlichen als Notwendigkeit aus der Partitionierung zwischen Hard- und Software. Der Flächenbedarf und die Performance sind daher in erster Linie nicht direkt von der Schnittstelle abhängig, sondern eher von der Partitionierung. Die an der Schnittstelle zwischen Hard- und Software stattfindende Kommunikation kann allerdings einen entscheidenden Einfluss auf diese Kriterien haben.

In diesem Paper wird eine Untersuchung des Einflusses der Verschiebung der HW/SW-Schnittstelle am Beispiel des LonTalk-Protokolls auf die Performance, den Speicher- und den Flächenbedarf einer Schaltung dargestellt. Ausgangspunkt ist eine Software-Referenz-Implementierung des LonTalk-Protokollstacks [Ade98]. Dabei handelt es sich um das Protokoll des Feldbussystems LonWorks® [Ech99], welches häufig in der Prozess- und Gebäudeautomatisierung eingesetzt wird. Der Protokollstack wurde im Rahmen dieser Fallstudie auf einem Altera Excalibur [Alt] implementiert. Neben einem Mikroprozessorsubsystem mit integriertem ARM9-Prozessorkern beinhaltet dieser Chip ein FPGA, so dass sich Hard- und Software sowie deren Schnittstelle auf demselben Chip implementieren lassen. Neben dem Excalibur-internen Speicher (interner Single- und Dual-Port SRAM), sowie der Möglichkeit der Speicherimplementierung im FPGA, steht auf dem benutzten Entwicklungsboard EPXA1 [Alt] ein externer Speicher zur Verfügung (externer SDRAM). Der für die HW/SW-Kommunikation der unterschiedlichen Partitionierungen benötigte Speicher wird auf diese verschiedenen Speichertypen abgebildet werden.

2 Systemüberblick

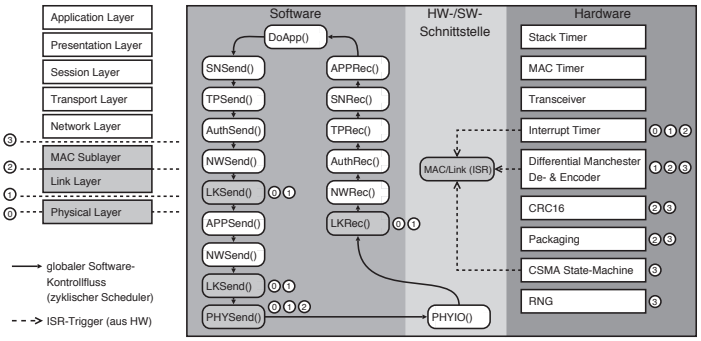


Abbildung 1: Systemüberblick und Partitionierungsvarianten

Die Software-Architektur des LonTalk-Protokollstacks orientiert sich am OSI-Schichtenmodell, welches in Abbildung 1 links dargestellt ist. Bei der HW/SW-Partitionierung wurde mit einer maximalen Software-Implementierung begonnen ⓪, die nur Teile des *Physical Layers* in Hardware implementiert. Die weiteren untersuchten Partitionierungen ① - ③ sind durch die entsprechenden Schnitte im OSI-Schichtenmodell dargestellt, wobei die unterhalb jedes Schnittes befindlichen Schichten in Hardware, die darüberliegenden in Software implementiert wurden.

Rechts in Abbildung 1 sind die aus den verschiedenen HW/SW-Partitionierungen resultierenden Gesamtsysteme, bestehend aus Software, die auf einem Prozessor (ARM9) ausgeführt wird, dedizierter Hardware (FPGA) und einer HW/SW-Schnittstelle, dargestellt. Die Ziffern neben den Funktionen und den Hardwaremodulen geben an, bei welcher der Partitionierungen diese vorhanden sind, alle übrigen Komponenten sind immer vorhanden. Zur Vereinfachung der Untersuchungen wurde auf die Benutzung eines Betriebssystems verzichtet und ein einfacher zyklischer Scheduler mit freiwilliger Kontrollabgabe benutzt. Dieser ruft ausgehend von einer benutzerdefinierten Anwendung `DoApp()` nacheinander Funktionen der verschiedenen Protokollschichten auf. Zunächst die Funktionen zum Senden, gefolgt von Funktionen zum Empfang von Datagrammen. Die Funktion `PHYIO()` wird zur Erfassung von Sensorwerten aus der Umgebung des Systems benutzt.

Die Kommunikation zwischen Soft- und Hardware erfolgt mit Hilfe von Memory-Mapped-I/O und einer Interrupt Service Routine (ISR). Hardwaremodule welche einen Interrupt auslösen können sind in Abbildung 1 entsprechend gekennzeichnet. Die ISR implementiert alle zeitkritischen Komponenten des Protokollstacks (MAC Sublayer, sowie die Steuerung zum Senden und Empfangen von Daten), weshalb die ISR der Varianten ⓪ bis ② mit Hilfe eines Timer Interrupts periodisch ausgeführt wird. In Variante ③ hingegen sind alle zeitkritischen Komponenten der ISR in HW realisiert (CSMA-State-Machine). Eine zyklische Ausführung der ISR ist nicht mehr erforderlich, und sie wird nur noch nach dem Empfang eines gültigen LonTalk-Paketes durch die CSMA-State-Machine ausgeführt.

3 Kommunikationsgetriebene Hardware/Software-Partitionierung

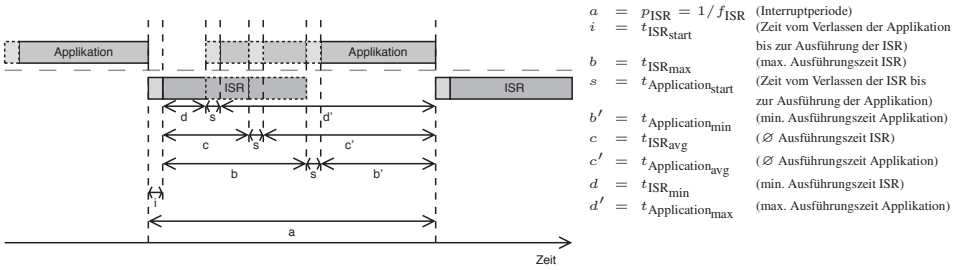


Abbildung 2: Zeit-Diagramm der Unterbrechungen der Applikation durch die ISR [Grü05]

Der zyklische Scheduler kann durch die Auslösung eines Interrupts zu jedem beliebigen Zeitpunkt durch die ISR unterbrochen werden (siehe Abbildung 2). Für jede gültige Implementierung gilt die Zeitbedingung: $t_{\text{ISR}_{\text{max}}} < p_{\text{ISR}} - t_{\text{ISR}_{\text{start}}}$. Ziel der HW/SW-Partitionierung ist es, ausgehend von einer maximalen Software-Implementierung ⓪, durch Auslagerung der Funktionalität von Soft- in Hardware ① - ③, die durchschnittliche Ausführungszeit der Applikation $t_{\text{Application}_{\text{avg}}}$ zu maximieren. Dies kann durch die Minimierung von $t_{\text{ISR}_{\text{avg}}}$ bei konstantem p_{ISR} erfolgen oder durch eine Vergrößerung von p_{ISR}, was wiederum eine Auswirkung auf $t_{\text{ISR}_{\text{avg}}}$ hat und sich damit direkt auf $t_{\text{Application}_{\text{avg}}}$ auswirkt. Als Optimierungskriterium wird daher die Funktion

$$f(t_{\text{Application}_{\text{avg}}}, t_{\text{ISR}_{\text{avg}}}) = \frac{t_{\text{Application}_{\text{avg}}}}{t_{\text{ISR}_{\text{avg}}}} \qquad t_{\text{Application}_{\text{avg}}}, t_{\text{ISR}_{\text{avg}}} \in \mathbb{R}_{>0} \tag{1}$$

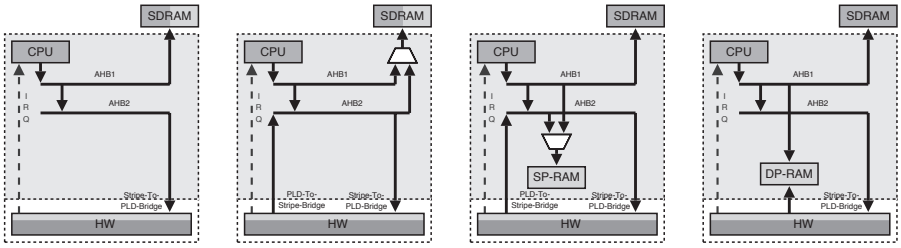
gewählt, welche es durch die Wahl der Partitionierung und unter Einhaltung aller Zeitbeschränkungen zu maximieren gilt.

4 Plattformbasiertere Co-Simulation & Implementierung

Die Implementierung und Untersuchung der unterschiedlichen HW/SW-Partitionierungen (siehe Abbildung 1) des LonTalk-Protokolls wurden unter Benutzung eines heterogenen Co-Designflows (vgl. [Opp04]) durchgeführt. Auf dem System-Level findet zunächst eine HW/SW-Partitionierung des in der initialen Spezifikation angegeben Systems (LonTalk-Referenz-Implementierung) statt, woraus die Anforderungen für die HW/SW-Kommunikation resultieren. Um aus der System-Level-Beschreibung ein ausführbares Modell zu erhalten, mit dem erste Untersuchungen des Systemverhaltens durchgeführt werden können, wird der Softwareteil mit Hilfe einer Programmiersprache und der Hardwareteil mit Hilfe einer Hardwarebeschreibungssprache (HDL) ausgedrückt (heterogenes, ausführbares Modell). Die Ausführung dieses Modells findet mit Hilfe einer HW/SW-Co-Simulation statt. Dazu wurde im Rahmen dieser Fallstudie der Altera Excalibur Stripe Simulator (ESS) [Alt] eingesetzt. Nach der erfolgreichen Co-Simulation mit Hilfe des ausführbaren Modells erfolgt die Implementierung auf dem Altera Excalibur EPXA1 Prototyping-Board.

Die Architektur des Altera Excalibur ermöglicht die Untersuchung unterschiedlicher Kommunikations-Architekturen zwischen Hard- und Software, wobei die Anforderungen bezüglich dieser Kommunikation von der gewählten HW/SW-Partitionierung abhängt. Abbildung 3 zeigt die untersuchten Kommunikations-Architekturen im Überblick, wobei sich die dort angegebenen Ziffern auf die in Abbildung 1 angegebenen Partitionierungen beziehen. Der mit HW beschriftete Teil bezeichnet den Hardwareteil aus Abbildung 1. Der darüberliegende Teil beschreibt das Mikroprozessorsubsystem mit ARM9-Prozessorkern.

Die Partitionierungsvarianten ⓪ und ① führen eine bitweise Übertragung der Daten zwischen Soft- und Hardware durch, weshalb eine einfache Master-Slave Kommunikation über den AHB-Bus [Alt] benutzt wird (Abbildung 3(a)). Bei der Variante ② können die Daten entweder byte- oder paketweise zwischen Soft- und Hardware übertragen werde, wobei im ersten Fall die einfache Kommunikations-Architektur (a) benutzt wird. Im zweiten Fall und bei der Variante ③ empfängt die Hardware ein vollständiges LonTalk-Paket vom Transceiver und löst nach erfolgreichem Empfang einen Interrupt aus. Die auf diese Art empfangenen Paketdaten (max. 256 Byte) müssen in einem Speicher abgelegt werden. Dazu existieren drei Möglichkeiten: 1. im externen SDRAM (Abbildung 3(b)), 2. im internen Single-Port RAM (Abbildung 3(c)) oder 3. im internen Dual-Port RAM (Abbildung 3(d)).



(a) GP I/O (0,1,2) (b) ext. SDRAM (2,3) (c) int. Single-Port RAM (2,3) (d) int. Dual-Port RAM (2,3)

Abbildung 3: Übersicht der Kommunikations-Architekturen der Implementierungsvarianten [Grü05]

5 Experimentelle Ergebnisse

Zur Partitionierungsvariante ⓪ existiert keine gültige Implementierung, weil die Zeitbedingung $t_{\mathrm{ISR}_{\mathrm{max}}} < p_{\mathrm{ISR}} - t_{\mathrm{ISR}_{\mathrm{start}}}$ verletzt wird. In Abbildung 4 sind die Geschwindigkeit, der dynamische Speicherverbrauch der ISR und die Größe des zur HW/SW-Kommunikation benutzten Speichers der Partitionierungsvarianten ① - ③, mit jeweils unterschiedlichen Kommunikations-Architekturen, gegenübergestellt. Zur Ermittlung der Geschwindigkeit wird Gleichung 1 benutzt, deren Berechnung durch eine Messung von $t_{\mathrm{ISR}_{\mathrm{avg}}}$ und p_{ISR} (unter Vernachlässigung von $t_{\mathrm{ISR}_{\mathrm{start}}}$ und $t_{\mathrm{Application}_{\mathrm{start}}}$) auf dem Prototypingboard ermöglicht wird. Der Flächenverbrauch im FPGA ist in der Größenordnung LC (Logical Cells) dargestellt und unterteilt sich in funktionale, zur HW/SW-Kommunikation und zur Anbindung

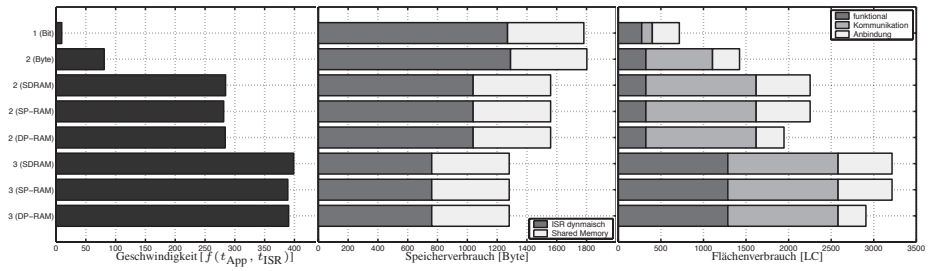


Abbildung 4: Experimentell ermittelte Ergebnisse

an das Mikroprozessor-Subsystem des Excalibur benutzte Hardwarekomponenten. Wie Abbildung 4 zeigt, bringt eine Verlagerung der Funktionalität von Soft- in Hardware eine Erhöhung der Geschwindigkeit. Die Verringerung des dynamischen Speicherbedarfs der ISR ergibt sich aus der Verlagerung zeitkritischer SW-Komponenten in HW. Von entscheidender Bedeutung ist neben der Partitionierung die Wahl der HW/SW-Kommunikation, wie an den Implementierungsvarianten der Partitionierung ② (byte- bzw. paketweise) zu erkennen ist. Die Unterschiede in der Wahl der Kommunikations-Architektur haben sich bei den Varianten ② und ③ in Bezug auf die Geschwindigkeit als nicht signifikant herausgestellt, zumal die Vorteile des DP-RAMs nicht genutzt werden konnten (externer SDRAM wird mit 133 MHz, interner SP- & DP-RAM wird mit 100 MHz getaktet).

6 Zusammenfassung

Es wurde gezeigt, dass die Wahl der HW/SW-Partitionierung einen entscheidenden Einfluss auf die Leistungsdaten eines eingebetteten Systems hat. Neben der Verlagerung der Funktionalität von Soft- in Hardware spielt die Wahl der HW/SW-Kommunikation (und die damit verbundene Kommunikations-Architektur) eine nicht zu unterschätzende Rolle.

Literatur

[Ade98] Adept Systems Incorporated. A C Reference Implementation of the LonTalk Protocol on the MC68360, Juli 1998. Document Revision 1.7, Code Revision 1.7.

[Alt] Altera Corporation. Excalibur Devices. http://www.altera.com/products/devices/arm.

[Ech99] Echelon Corporated. Introduction to the LonWorks System, 1999. Users Guide, ver. 1.0.

[Grü05] K. Grüttner. Einfluss der durch HW-/SW-Partitionierung hervorgerufenen Kommunikation auf Leistungsdaten einer Schaltung. Diplomarbeit, C.v.O. Universität Oldenburg, 2005.

[Opp04] F. Oppenheimer. *OOCOSIM - An Object-Oriented Co-design Method for Embedded HW/SW Systems*. Dissertation, C.v.O. Universität Oldenburg, 2004.