Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung AW2

Marco Kirschke

Anwendungsgebiete unterschiedlicher FPGA-basierter MPSoC-Architekturen

Fakultät Technik und Informatik
Department Informatik

Inhaltsverzeichnis

1 Einleitung

Im Anwendungsbereich der eingebetteten Systeme ist die Verwendung von FPGA Technologie durch die Vorteile, die sich aufgrund der parallelen Verarbeitung bieten, ein verbreiteter Ansatz. Die aktuell steigende Anzahl an Logikressourcen auf den FPGA Bausteinen und die zunehmende Verfügbarkeit von Softcore Prozessoren begünstigen den Einsatz von Multiprozessor-Architekturen bei der Erstellung von Echtzeitsystemen.

Für die Hersteller solcher Systeme ergeben sich daraus zwei wesentliche Vorteile: einerseits entstehen flexiblere Entwicklungsprozesse, da die Hardwarekonfiguration zu jedem Zeitpunkt erweitert oder angepasst werden kann. Auf der anderen Seite erlaubt der Einsatz von mehreren Mikroprozessoren die Verarbeitung von parallelen Berechnungen, und durch entsprechende Multiprozessor-Anwendungen können Bestandteile und Konzepte aus dem Bereich der Software-Entwicklung übertragen werden. Weitere Vorteile FPGA-basierter MPSoC, wie der Einsatz in mobilen Systemen mit energieeffizenten und wärmereduzierten Leistungsmerkmalen, wurden bereits in (Kirschke, 2010) behandelt.

In dieser Arbeit sollen die Anwendungsgebiete und Entwicklungsstände von Multiprozessor-Architekturen anhand von aktuellen Forschungsergebnissen und Veröffentlichungen dargestellt werden. Dabei wird zwischen heterogenen und homogenen Architekturen unterschieden. Der Kernpunkt heterogener Architekturen ist das Zusammenspiel verschiedener Prozessorkonfigurationen, mit unterschiedlichen Teilfunktionen zur Lösung einer Gesamtanwendung. Homogene Architekturen verwenden dagegen mehrere identische Prozessortypen und erzeugen die Ergebnisse ihrer Berechnungen durch die Verteilung und parallelen Verarbeitung von Teilaufgaben.

Im zweiten Teil dieser Ausarbeitung werden die Ergebnisse einer Untersuchung vorgestellt, welche die Inhalte von Veröffentlichungen der IEEE Digital Library mit dem Schwerpunkt FPGA-basierter MPSoC der letzten vier Jahre zusammenfasst. Die in Kapitel 3 behandelte Veröffentlichung stellt eine typische Anwendung für eine heterogene MPSoC Architektur in einem industriellen Kontext dar. Dem gegenüber steht im vierten Teil eine Studie, die verschiedene Konfigurationen von homogenen MPSoC zusammenfasst und deren Leistungsmerkmale analysiert. Die im fünften Kapitel angeführte Zusammenfassung vergleicht die in dieser Arbeit behandelten Veröffentlichungen abschließend miteinander.

2 Veröffentlichungen der IEEE Digital Library zum Thema FPGA / MPSoC

Ausgehend von einem Bericht im Rahmen der International Conference on Reconfigurable Computing and FPGAs, in dem unter anderem die steigende Anzahl an Veröffentlichungen zum Thema „FPGA & Multiprocessor“ behandelt wurde [(Dorta u. a., 2009)], sollte im Vorfeld dieser Arbeit untersucht werden, auf welche konkreten Schwerpunkte sich diese beziehen. Dazu wurden 50 Veröffentlichungen der IEEE Xplore Library aus den letzten vier Jahren, anhand der Schlagworte „FPGA“ und „MPSoC“ ermittelt und unter den in Tabelle 1 dargestellten Gesichtspunkten kategorisiert.

Die Auswertung der Inhalte zeigt, dass in einem Großteil der Arbeiten ein Entwurf von Multiprozessorsystemen mit homogener Architektur als Prototypsystem vorgestellt wurde; wobei diese überwiegend innerhalb eines wissenschaftlichen Umfeldes an Universitäten oder Hochschulen als Forschungsprojekte durchgeführt worden sind [vgl. Abbildung 1 und Übersicht in 6.1]. Daraus ist jedoch nicht zwingend abzuleiten, dass heterogene Architekturen in MPSoC weniger verbreitet sind, da diese eine höhere praktische Relevanz besitzen und in der Regel weniger Veröffentlichungen von konkreten Entwicklungsarbeiten mit einem kommerziellen bzw. industriellen Hintergrund verfügbar gemacht werden. Die Verwendung von Xilinx Produkten bei der Erstellung von MPSoC ist anhand der verwen-

Architektur	homogen / heterogen
Inhalt	Vorstellung eines Prototypen / Implementierung einer konkreten Aufgabe / Erstellung eines Frameworks
verwendete (Softcore) Prozessoren	MicroBlaze / Nios / PPC / ARM / etc.
Hersteller	Xilinx / Altera / etc.
NoC	Einsatz von Network-on-Chip Technologie

Tabelle 1: Kategorien für die Auswertung der Inhalte der IEEE Veröffentlichungen

deten Prozessoren und direkt über die Herstellerübersicht zu sehen, und lässt sich unter anderem auf die Tatsache zurückführen, dass Xilinx bereits bevor Softcore Prozessoren in dem jetzigen Umfang verfügbar waren, einen Teil ihrer FPGAs mit fest implementierten PPC Prozessoren ausgerüstet hat, und somit eine gewisse Erfahrung in diesem Bereich vorweisen kann. Festzustellen ist insbesondere,

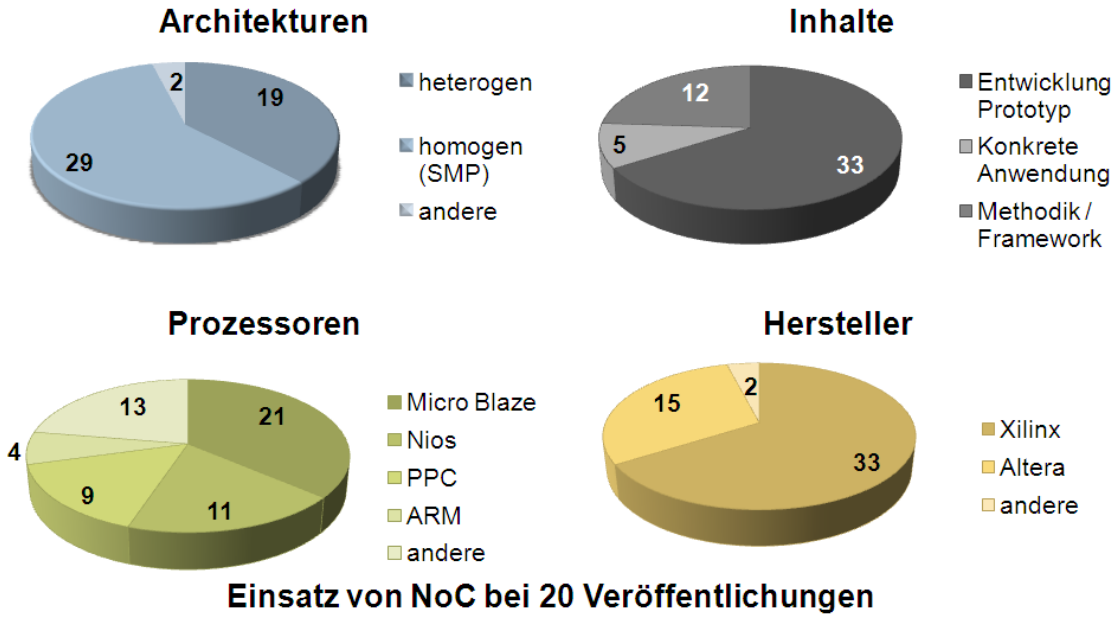

Abbildung 1: Inhalte der untersuchten IEEE Digital Library Veröffentlichungen zu den Schlagworten „FPGA“ und „MPSoC“

dass bei 40% der betrachteten Veröffentlichungen, die Entwicklung und Verwendung von Network-on-Chips (NoC) vorgestellt wurden. Der Einsatz von NoC stellt, gerade für homogen aufgebaute MPSoC, eine Alternative zur Verwendung von herkömmlichen Bussystemen dar, um die Synchronisation und Kommunikation der Prozessoren untereinander zu realisieren.

3 Heterogene Multiprozessor-Architekturen

Die Verwendung von unterschiedlichen, auf Spezialanwendungen optimierten Prozessoren ist ein typisches Mittel in der Entwicklung von eingebetteten Systemen. Dabei werden die Prozessoren je nach Art und Einsatzgebiet selektiert und auf konkrete Aufgaben eingestellt, um ein bestmögliches Verhältnis zwischen Systemressourcen und Systemkosten zu erhalten. Durch die Integration von Beschleunigungskomponenten, wie Coprozessoren und Signal- bzw. Bildbearbeitungs-Pipelines, können zudem parallele Verarbeitungsprozesse von großen Datenmengen in FPGA-basierten Echtzeitanwendungen erzielt werden. Diese beiden Ansätze werden durch die bereits beschriebene Entwicklung der FPGA-

Technologie miteinander verbunden, und lassen sich zudem auf einer gemeinsamen Plattform erstellen.

3.1 Vorteile FPGA-basierter MPSoC im industriellen Umfeld

Das hier vorgestellte Beispiel einer heterogenen Multiprozessor-Architektur wurde in Zusammenarbeit der Universität Rostock und der *basysPrint Ltd.* erstellt und in (Joost und Salomon, 2005) veröffentlicht. Die Firma *basysPrint* ist mittlerweile Teil der Punch Graphix NV, die einen weltweiten Anbieter von Drucksystemen für den Einsatz in der Druckindustrie darstellt.

Die Arbeit beschreibt die Erweiterung einer bestehenden Systemkonfiguration durch ein MPSoC, um Schwachstellen des Ausgangssystems zu beseitigen und die Integration neuer Komponenten zu erleichtern. Bereits 1993 wurde von *basysPrint* eine Technologie (CTcP)[1] zur Herstellung von Druckerplatten im Offset- Verfahren entwickelt, die entgegen zu den bestehenden Lösungen eine Belichtung der Druckerplatten mit ultraviolettem Licht anstatt von Lasern vornimmt. Die dabei verwendeten UV sensitiven Druckerplatten sind umweltverträglicher und wesentlich kostengünstiger als vergleichbare Platten, die mittels Laserbestrahlung belichtet werden müssen; zusätzlich entstehen weniger Energiekosten bei der Erzeugung des UV-Lichts. Diese Vorteile machen das CTcP System, UV-Setter in Abbildung 2 dargestellt, für Unternehmen interessant, die eher ein mittleres Auftragsvolumen und somit häufig wechselnde Druckaufträge verarbeiten (basysPrint, 2010).

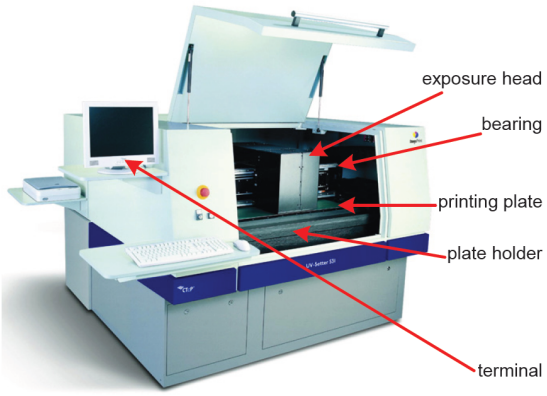

Abbildung 2: Das CTcP System UV-Setter der Firma basysPrint

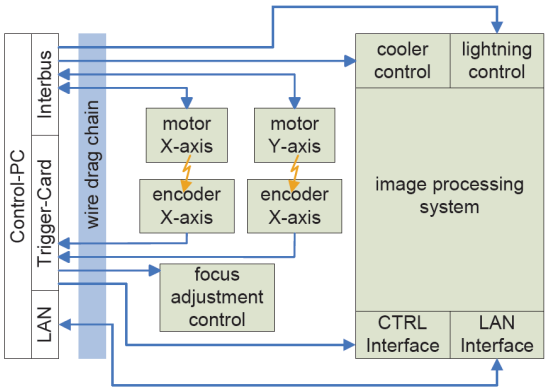

Abbildung 3: Die Ausgangskonfiguration des UV-Setter Systems; das *image processing system* beinhaltet die Komponenten des Belichtungskopfes

Für die Belichtung der Druckerplatten werden die zu druckenden Vorlagen in Streifen unterteilt und über einen *Scrolling Mode* nacheinander auf die Platte übertragen. Dabei ist sowohl die präzise Steuerung der Motoren als auch die unterbrechungsfreie Übertragung der Bilddaten an den Belichtungskopf von größter Bedeutung, um Fehldrucke und somit die Zerstörung der gesamten Platte zu vermeiden.

In der Ausgangskonfiguration des Systems [vgl. Abbildung 3] werden die Bilddaten über eine LAN Schnittstelle vom Control-PC zum Belichtungskopf übertragen. Auf dem Control-PC wird die Berechnung der Steuerinformationen für den Belichtungskopf und der Motoren durchgeführt und über separate Verbindungen an die entsprechenden Komponenten verteilt. Das System wurde seit der ersten Version mehrmals angepasst und erweitert, wodurch die vorhandenen Systemressourcen zum Zeitpunkt der Zusammenarbeit mit der Universität Rostock nahezu ausgeschöpft waren. Durch die Modellierung einer neuen Systemkonfiguration sollten folgende Schwachstellen des Ausgangssystems behoben werden:

[1] CTcP - Computer To conventional Plate

- Die Übertragung der Bilddaten und Steuerungstelegramme erfordert eine Vielzahl von physikalischen Leitungen, die zum Belichtungskopf geführt werden müssen und die Bewegungsfreiheit des Kopfes nicht einschränken dürfen.
- Um diese Bewegungsfreiheit zu gewährleisten, besitzen die Leitungen eine gewisse Länge, welche bei der Übertragung der Echtzeitdaten die Signalstabilität und Zuverlässigkeit beeinträchtigt.
- Die Verwendung von verschiedenen Steuerungstelegrammen mit unterschiedlichen Timing Anforderungen erschwert den Umgang mit dem System und schränkt Erweiterungen ein.
- Die Integration einer von *basysPrint* verbesserten Verarbeitungsprozedur, die mechanische Störungen und Vibrationen kompensieren kann, war zum damaligen Zeitpunkt aufgrund fehlender Ressourcen nicht realisierbar.

Die Umsetzung dieser Anforderungen wurde durch ein ALTERA NIOS II-basiertes MPSoC realisiert. Die Anzahl der Leitungen zum Belichtungskopf wurden auf ein Minimum reduziert, so dass nur noch die Bilddaten und grundlegende Steuerungsparameter an diesen übertragen werden. Das neue System realisiert die Verarbeitung der Motorensteuerung, der Trigger Logik und die Koordination des Belichtungsvorganges direkt im MPSoC auf dem Belichtungskopf [vgl. Abbildung 4].

Das MPSoC umfasst drei NIOS II Prozessoren, wobei jeder eine unterschiedliche Anwendung des Gesamtsystems erfüllt. Der *Core 0 master* Prozessor behandelt die Kommunikation und den Empfang von Bilddaten und Statusinformationen vom Control-PC. Des Weiteren steuert er den Einsatz der beiden *Core 1 slave* und *Core 2 slave* Prozessoren. Die Verarbeitung der Echtzeitanwendungen zur Belichtung der Platte, das Auswerten der Trigger Logik und das Ansteuern der Motoren, wird ausschließlich auf *Core 1 slave* vollzogen. Der zweite Slave Prozessor wird dagegen als Peripherie Kontroller eingesetzt, der neben der Fokus Kontrolle und dem Kühler weitere Peripheriekomponenten bearbeitet [vgl. Abbildung 5]. Die Prozessoren kommunizieren untereinander über shared-memory, was den Vorteil besitzt, dass die Bilddaten nur einmal vom Master Prozessor im Speicher abgelegt werden müssen; der verarbeitende Slave Prozessor kann direkt auf diese Daten zugreifen. Die Zugriffskontrolle wird über den von ALTERA verfügbaren HardwareMutex sichergestellt, so dass ein gleichzeitiger Zugriff auf den gemeinsam genutzten Speicher von mehreren Prozessoren ausgeschlossen wird.

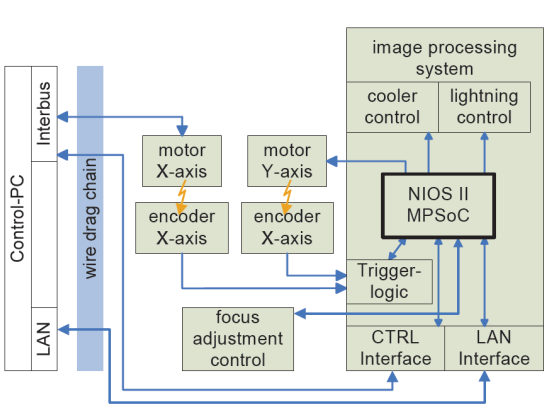

Abbildung 4: Das modifizierte System mit reduzierter Anzahl an Leitungen und dem MPSoC auf dem Belichtungskopf

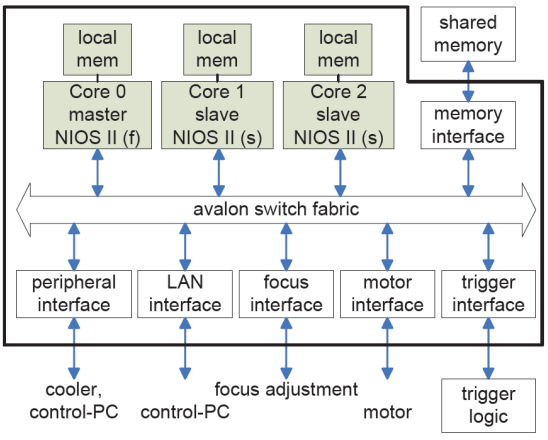

Abbildung 5: Das MPSoC des Belichtungskopfes mit den drei heterogen agierenden NIOS II Prozessoren

Die Realisierung des Systems wurde auf einem Entwicklungsboard umgesetzt, das neben einem ALTERA Cyclone EP1C20 FPGA auch eine Reihe von integrierten Peripherieelementen, wie ein 16MB großer SDRAM, 8MB Flash Speicher, eine Onboard Ethernet Schnittstelle sowie diverse Ein- und Ausgabe Schnittstellen aufweist. Der Ressourcenverbrauch auf dem FPGA für das vorgestellte System

lag mit 9000 Logik Elementen bei 45% der Gesamtkapazität, wodurch die Integration von weiteren Komponenten gegeben ist.

Die oben erwähnten Schwachstellen des Vorgängersystems konnten durch die Umsetzung als MPSoC beseitigt werden:

- Die Anzahl physikalischen Leitung wurden durch die zentrale Verarbeitungslogik auf dem Belichtungskopf reduziert.
- Die Länge dieser Leitungen ist durch die zentrale Verarbeitung und der Verwendung des shared-memory nicht mehr relevant.
- Die zentrale Verarbeitung reduzierte darüber hinaus die Anzahl an Kommunikationstelegrammen vom Control-PC zum Belichtungskopf.
- Für die Integration weiterer Komponenten sind genügend FPGA Ressourcen verfügbar.

4 Homogene Multiprozessor-Architekturen

FPGA basierte Systeme, die eine homogene Multiprozessor Architektur verwenden, stellen aufgrund der wachsenden Verfügbarkeit von Softcore Prozessoren und der zunehmenden Anzahl an Logikressourcen auf den FPGA Bausteinen, neue Ansätze für die Entwickler dar. Aus dieser Aktualität ergibt sich der Umstand, dass sich der Großteil der verfügbaren Arbeiten zum Thema homogen agierender MPSoC, mit der Erforschung von Grundlagen zur Konfiguration und der Festlegung von Kommunikationsstrategien beschäftigt.

4.1 Leistungsmerkmale symmetrischer MPSoC

Das hier angeführte Beispiel von homogenen MPSoC-Architekturen ist eine Studie der spanischen Rey Juan Carlos University aus Mostoles (Huerta u. a., 2009), in der vier MPSoC erzeugt wurden, die sich in Bezug auf ihre Speicherzugriffsverfahren unterscheiden. Für die Verarbeitung von Anwendungen auf symmetrischen Multiprozessor (SMP) Systemen waren eine Reihe von Vorarbeiten nötig. So werden in den vorgestellten Systemen Virtex 2 bzw. Virtex 4 FPGAs und weitere verfügbare Systembausteine, wie Memory Controller und Bus Systeme, der Firma Xilinx verwendet; das bereitgestellte Betriebssystem Xilkernel bietet jedoch keine direkte Multiprozessor Unterstützung.

Aus diesem Grund wurde in einer Vorarbeit ein Multiprozessor Betriebssystem entwickelt, welches zwar auf dem Xilkernel basiert, und somit grundlegende Funktionalitäten, wie das Scheduling von POSIX Threads und die Verfügbarkeit von Kommunikations- und Synchronisationsdiensten (mailboxes, semaphores und mutexes) unterstützt, gleichzeitig aber auch Funktionen zur parallelen Verarbeitung anbietet. Zusätzlich mussten Hardwarekomponenten entwickelt werden, welche die Identifikation und die Synchronisation der Prozessoren untereinander ermöglichen. Die Identifikation erfolgt über ein einfaches FSL-Register[2], in dem die ProzessorID statisch abgelegt wird; die Synchronisation ist über ein Harwaremutex Modul (*hw_mutex*) realisiert, über welches der exklusive Zugriff auf kritische Speicherbereiche erzeugt wird. Das Betriebssystem setzt weiterhin voraus, dass jeder Prozessor über einen eigenen privaten Speicher verfügt, der als Stack für Kernel- und Systemanweisungen fungiert.

Im Folgenden werden die vier System mit ihren grundlegenden Eigenschaften und den Unterschieden zueinander vorgestellt. Alle Systeme verwenden den OPB[3], um einerseits das Hardware Modul

[2] Fast Simplex Link; der FSL ist ursprünglich eine unidirektionale Verbindung, die eine schnelle Kommunikation zwischen zwei Hardwaremodulen in einem SoC realisiert; diese erfolgt über ein Registerfile, das in zwei Taktzyklen beschrieben und gelesen werden kann.

[3] On-chip Peripheral Bus; ist ein von Xilinx bereitgestelltes Bussystem für den Einsatz in SoC und MPSoC

zur Synchronisation mit den Prozessoren zu verbinden und andererseits Timer für Laufzeitmessungen sowie eine UART Schnittstelle zum Host PC bereitzustellen.

4.1.1 Zwei MicroBlaze mit Speicherzugriff über LMB (System 1)

Das erste, in Abbildung 6 dargestellte System umfasst zwei MicroBlaze Prozessoren der Version 4.0a, die gemeinsam zwei 64 KB große Speicherbereiche des allozierten Block RAMs für Instruktionen und Daten verwenden. Dieses ist über zwei LMB[4] Schnittstellen erreichbar, wodurch beide Prozessoren zeitgleich auf den gleichen Speicherbereich zugreifen können; der Zugriff wird dabei über den *hw_mutex* synchronisiert. Zur Identifikation sind beide Prozessoren mit den separaten FSL Registern verbunden, aus denen sie ihre *cpu ID* beziehen. Weiterhin steht beiden Prozessoren ein 8 KB großer privater Speicher zur Verfügung der als Stack für die Betriebssystemaufrufe verwendet wird. Für die Synthese des Systems wurde ein XC2V6000 Virtex 2 FPGA der Firma Xilinx verwendet.

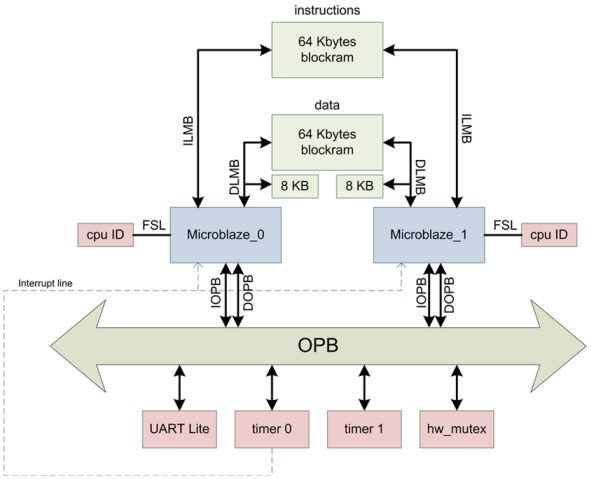

Abbildung 6: Konfiguartion von System 1 mit dem geteilten Speicher für Instruktionen und Daten und deren LMB Anbindungen

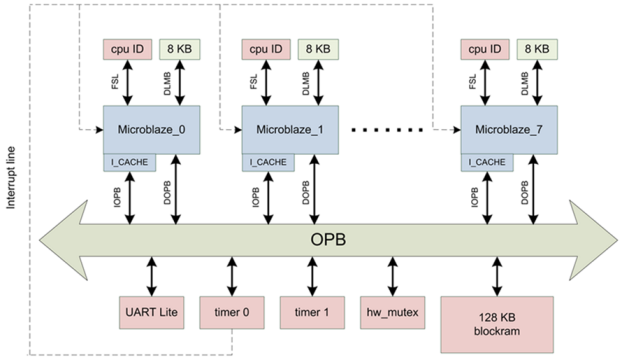

Abbildung 7: Das zweite System mit bis zu 8 Prozessoren und dem OPB für Speicherzugriffe auf das 128KB Block RAM

4.1.2 Bis zu 8 MicroBlaze mit Speicherzugriff über OPB (System 2)

Das zweite vorgestellte System unterstützt bis zu acht MicroBlaze Prozessoren, die über den OPB auf einen gemeinsamen ebenfalls im Block RAM angelegten Speicher mit einer Größe von 128 KB zugreifen können [vgl. Abbildung 7]. Die Prozessoren verfügen über einen 4 KB großen Instruction Cache, um die Busauslastung zu verringern. Zusätzlich verfügen die Prozessoren, wie schon im ersten System, über einen 8 KB großen privaten Speicher, eine Anbindung an ein FSL Register für die *cpu ID* und die oben genannten Schnittstellen an den OPB zur Kommunikation, Synchronisation und Zeitmessung. Die Anzahl der Prozessoren ist auf 8 begrenzt, da der OPB lediglich über 16 Master Schnittstellen verfügt, wobei jeweils zwei für Instruktionen und Daten von jedem Prozessor belegt werden. Die Synthese wurde wieder auf dem XC2V6000 Virtex 2 FPGA des ersten Systems durchgeführt.

4.1.3 Vier MircoBlaze und Verwendung des XCL (System 3 und System 4)

Die letzten beiden Systeme unterscheiden sich zu den vorangegangen primär dadurch, dass bei diesen eine neuere Version des MircoBlaze Prozessors verwendet wurde. Ab der Version 5.0a unterstützt dieser ein Cache Interface, den Xilinx Cache Link (XCL), der im Zusammenspiel mit dem Memory Controller für verbesserte Zugriffszeiten auf den gemeinsam verwendeten Speicher sorgt. Der in diesen

[4] Local Memory Bus

Systemen verwendete Memory Controller besitzt lediglich vier XCL Schnittstellen, wodurch die Anzahl der implementierbaren Prozessoren beschränkt wird. Für die beiden vorgestellten Systeme 3 und 4 werden alle verfügbaren XCL Schnittstellen für Anbindungen an instruction caches verwendet, was eine maximale Anzahl von vier Prozessoren für die Systeme zur Folge hat. Die Konfiguration der beiden Systeme ist weitestgehend identisch, so verfügen beide über einen 4 KB Instructioncache und 2 KB privaten Speicher für jeden Prozessor und der bereits erwähnten Anbindungen an die Timer und den UART über den OPB. Als einziger Unterschied wird in System 3 ein 1 MB großes SRAM als shared memory genutzt; in System 4 ein 64 MB großes DDR RAM. Darüber hinaus wurden beide Systeme auf einem kleineren FPGA implementiert, dem XC4VLX25 Virtex 4 FPGA ebenfalls von Xilinx. In Abbildung 8 wird die Systemkonfiguration von System 4 mit dem 64 MB großen DDR RAM gezeigt.

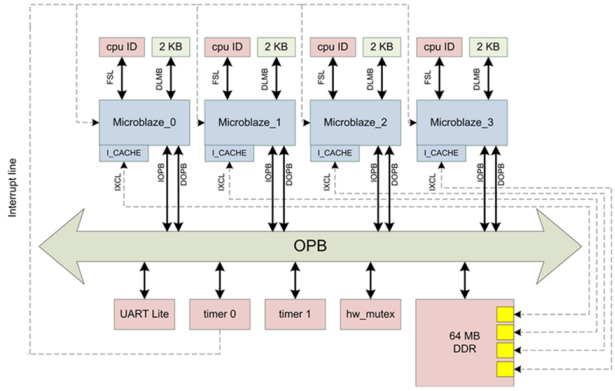

Abbildung 8: System 4, mit den direkten XCL Verbindungen der instruction caches zum gemeinsam verwendeten Speicher

	Matrix	Encryption	Decryption
Speedup	1.9853	1.9662	1.9809
Efficiency	1.4035	1.3899	1.4003

Tabelle 2: Geschwindigkeitsoptimierung und Effizienz des ersten Systems bei der Verwendung von zwei Prozessoren

4.1.4 Auswertung

Zur Bewertung wurden zwei typische Multiprozessoranwendungen implementiert und auf jedem System in mehreren Testläufen mit steigender Anzahl an Prozessoren ausgeführt. Dabei handelt es sich zum einen um eine verteilt ausgeführte Matrixmultiplikation und zum anderen um die Verschlüsselung bzw. Entschlüsselung von Daten mit dem AES Algorithmus, wie er beispielsweise in Wireless LAN Geräten verwendet wird. Für jede Systemkonfiguration wurden die Parameter

- *Speedup*: Berechnungszeit eines Prozessors / Berechnungszeit von N Prozessoren
- *Efficiency*: Speedup von N Prozessoren / Ressourcenverbrauch von N Prozessoren[5]

ermittelt und miteinander verglichen. Die Ergebnisse für System 1 zeigen, dass durch diese Konfiguration sehr gute Ergebnisse bei der Ausführung von Programmen erzielt werden können, die nur kleinere Zugriffe auf den Speicher verwenden [vgl. Tabelle 2]. Da die Prozessoren über separate Verbindungen zum Block RAM verfügen, entstehen kaum Verzögerungen bei den Speicherzugriffen; einzig die Synchronisationsabfragen über den OPB erfordern eine gewisse Bearbeitungszeit.

Aus der Auswertung für das zweite System ist erkennbar, dass ab der Verwendung von vier Prozessoren keine wesentlichen Geschwindigkeitsoptimierungen mehr zu erreichen sind [vgl. Abbildung

[5]Allgemein wird die Effizienz anhand des Speedups mit N Prozessoren / N (Anzahl der Prozessoren) berechnet. Da in den erstellten Systemen der Ressourcenverbrauch aber nicht proportional zur Prozessorenanzahl ist, wird für die Berechnung die tatsächliche Anzahl an benötigten Ressourcen (slices) verwendet; diese sind im Anhang 6.2 zu finden.

9]. Die Gründe sehen die Autoren in der spekulativen Speicherverwaltungsstrategie, die der MicroBlaze 4.0a verwendet; sobald ein Prozessor einen Speicherzugriff initiiert, wird die Instruktion auch an den OPB weitergeleitet, obwohl sich der Aufruf bereits im Cache befinden könnte. Diese Strategie ist für Systeme mit einem Prozessor ausgelegt und versucht so die Zugriffszeiten zu verringern. In einem Multiprozessor System wird dadurch lediglich eine erhöhte Auslastung des Systembusses (OPB) erzeugt, was Zugriffe von anderen Prozessoren auf den Speicher unnötig verzögert.

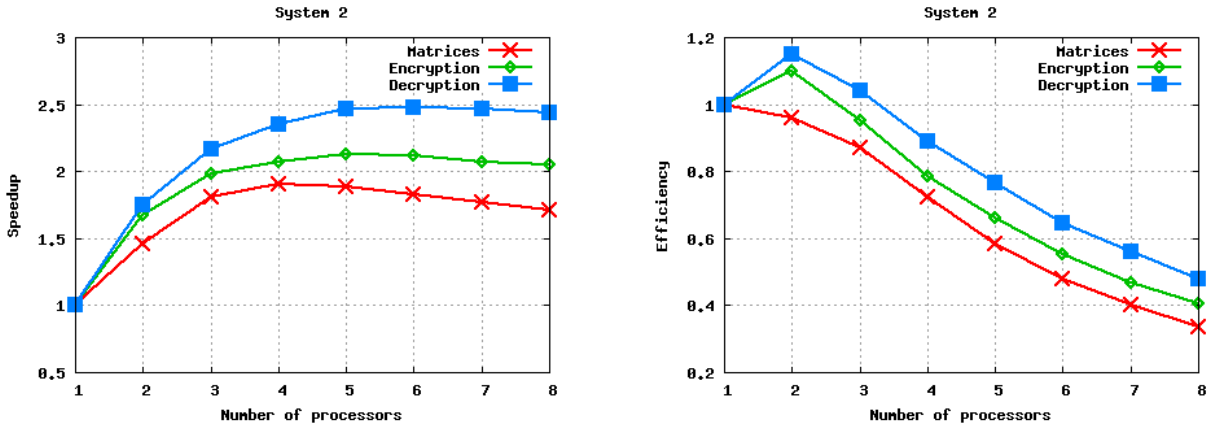

Abbildung 9: Die Auswertung für das zweite System 2 zeigt keine weitere Verbesserungen des Speedups mit mehr als 4 Prozessoren

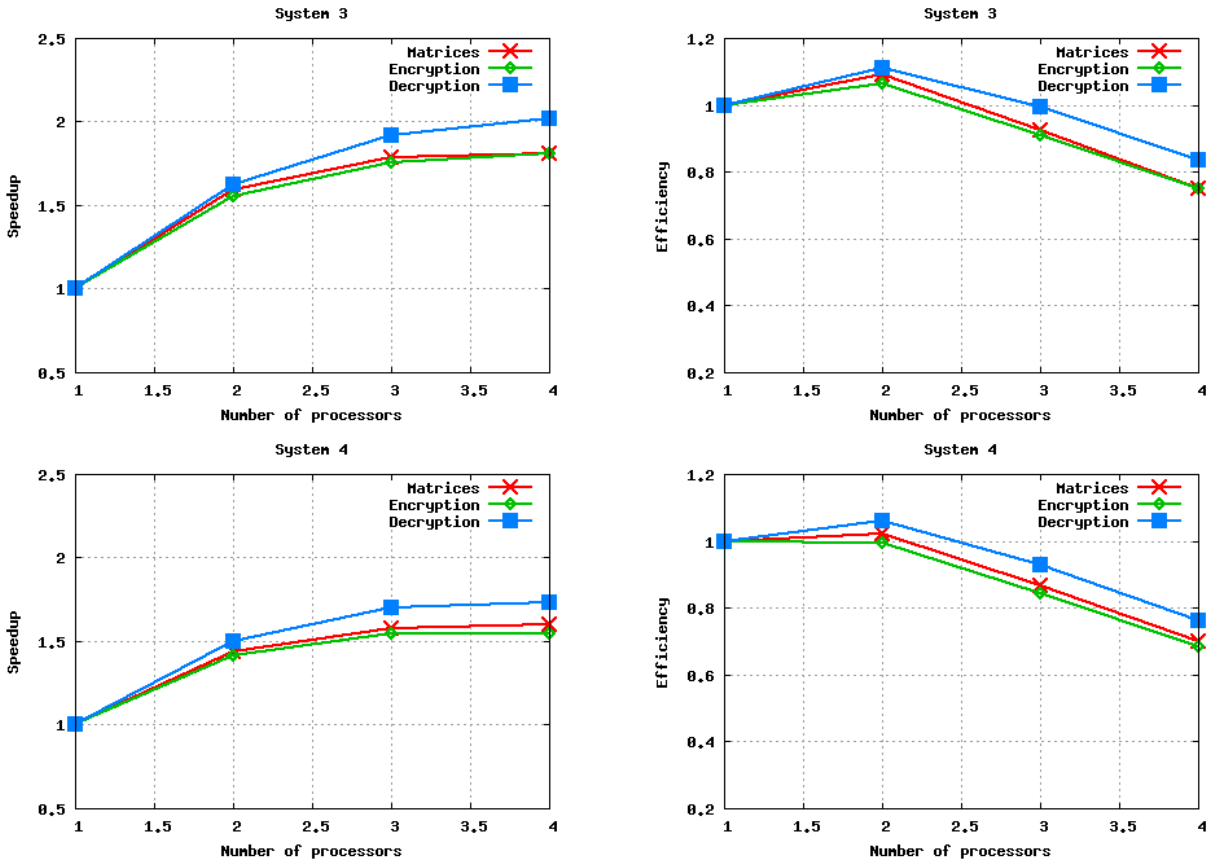

Abbildung 10: Die unterschiedlichen Ergebnisse für den Speedup bei den Systemen 3 und 4 lässt sich auf die tatsächlichen Zugriffszeiten des physischen Speichers zurückführen

Der Einsatz der neueren MicroBlaze Version in den Systemen 3 und 4 und die dadurch verfügbare Verwendung der XCL Schnittstellen zum Memory Controller stellt einen geeigneten Ansatz zur Entlastung des OPB dar, wie Abbildung 10 zeigt. Da in den vorgestellten Systemen aufgrund der begrenzten

Anzahl an XCL Schnittstellen nur die Instruktionen über Caches behandelt werden konnten, bietet sich für weitere Entwicklungen die Integration von Memory Controllern an, die eine größere Anzahl an XCL Schnittstellen verarbeiten können.

Fasst man die Ergebnisse der vorgestellten Studie zusammen, so wird deutlich, dass bei der Entwicklung von Systemen mit homogenen Multiprozessor-Architekturen, sowohl die Größe der Speicherzugriffe, die aus den Vorgaben der zu erstellenden Anwendung entstehen, als auch die Auswahl einer geeigneten Busstrategie, genauer betrachtet werden müssen. So können bei der Bearbeitung von Daten mit geringem Speicherbedarf bereits gute Ergebnisse mit direkten Speicherzugriffen [vgl.4.1.1] erreicht werden. Die Ergebnisse der Systeme 2 bis 4 zeigen dagegen, dass bei einer erhöhten Anzahl an Prozessoren die Art und Verwendung von geeigneten Bussystemen eine wesentliche Rolle für die Leistungsfähigkeit von MPSoC spielt.

5 Zusammenfassung

Das Einsatzgebiet von Hardware / Software Codesign gestützten Echtzeitsystemen hat sich durch die Integration von Multiprozessor-Architekturen auf einen größeren Anwendungsbereich ausgedehnt. Dabei können bei der Erstellung solcher Systeme die Architekturen an die jeweiligen konkreten Aufgaben angepasst werden, was neben einer großen Flexibilität in der Entwicklungsphase, auch schnelle Reaktionszeiten auf Veränderungen von öffentlichen Standards und Regelwerken mit sich bringt. Gerade in der Entwicklung von FPGA basierten Systemen stellt das neue Möglichkeiten dar, die bei traditionellen Entwicklungsprozessen nicht vorhanden waren.

Der Ansatz heterogener Multiprozessor-Architekturen wird bereits seit Jahren in vielen Bereichen der Industrie eingesetzt; durch die Implementierung solcher System auf Basis von FPGA-basierten MPSoC können die Entwickler nun auch während des Entwicklungsprozesses Änderungen an den Prozessorkonfigurationen vornehmen und Erweiterungen schneller in die Systeme integrieren. Gleichzeitig bleibt Ihnen die Eigenschaft erhalten, verfügbare Systemressourcen optimal auszunutzen, um somit kostengünstige Produkte zu entwickeln.

Die Erzeugung von homogenen Multiprozessor-Architekturen ist ein weiteres Gebiet, welches durch die wachsende Anzahl an Logikelementen auf den FPGAs und deren Verfügbarkeit bei sinkender Preisentwicklung begünstigt wird. Die Bereitstellung von Multiprozessoranwendungen, mit denen ein Großteil der parallelen Abläufe durch Softwareentwicklung gelöst werden kann, stellt einen weiteres interessantes Einsatzgebiet für FPGA-basierte eingebettete Systeme dar. Die Integration einer solchen softwaregestützten Entwicklung muss allerdings durch die genaue Planung der zugrundeliegenden Hardwarekonfiguration begleitet werden; darunter fallen das Design von Kommunikations- und Synchronisationsmechanismen für die Prozessoren, die Verwendungen geeigneter Bussysteme und Memory Controller sowie der Einsatz von SMP - Betriebssystemen zur verteilten Ausführung der Anwendungen. Zusätzlich muss in diesem Zusammenhang der Einsatz der Network-on-Chip Technologie untersucht werden, da diese unterschiedliche Strategien sowohl zur Anordnung der Prozessoren als auch zu deren Kommunikation untereinander vorsieht und die Erzeugung von skalierbaren MPSoC ermöglicht.

Literatur

[basysPrint 2010] BASYSPRINT: *Web presence of basysPrint - vendor of CTcP technology*, 2010. – URL `http://www.basysprint.com/en/technology`. – from companies webpage [Jul 2010]

[Dorta u. a. 2009] DORTA, Taho ; JIMÉNEZ, Jaime ; MARTÍN, José L. ; BIDARTE, Unai ; ASTARLOA, Armando: *Overview of FPGA-Based Multiprocessor Systems*. In: *2009 International Conference on Reconfigurable Computing and FPGAs*, URL `http://ieeexplore.ieee.org`, 2009. – IEEE Digital Library [May 2010]. – ISBN 978-0-7695-3917-1

[Huerta u. a. 2009] HUERTA, Pablo ; CASTILLO, Javier ; PEDRAZA, Cesar ; CANO, Javier ; MARTINEZ, Jose I.: *Symmetric multiprocessor systems on FPGA*. In: *2009 International Conference on Reconfigurable Computing and FPGAs*, URL `http://ieeexplore.ieee.org`, 2009. – IEEE Digital Library [Apr 2010]. – ISBN 978-1-4244-5293-4

[Joost und Salomon 2005] JOOST, Ralf ; SALOMON, Ralf: *Advantages of FPGA-Based Multiprocessor Systems in Industrial Applications*. In: *Industrial Electronics Society, 2005. IECON 2005. 31st Annual Conference of IEEE*, URL `http://ieeexplore.ieee.org`, 2005. – IEEE Digital Library [Apr 2010]. – ISBN 0-7803-9252-3

[Kirschke 2010] KIRSCHKE, Marco: *Implementierungsansätze für ein FPGA basiertes Multiprozessorsystem*. January 2010. – Hochschule für Angewandte Wissenschaften - Hamburg

6 Anhang

6.1 Übersicht der IEEE Digital Library Veröffentlichungen

#	Jahr	Name	Architektur			Inhalte			Prozessoren					Hersteller			NOC
			heterogen	homogen (SMP)	andere	Entwicklung Prototyp	Konkrete Anwendung	Methodik / Framework	Micro Blaze	Nios	PPC	ARM	andere	Xilinx	Altera	andere	
5	2006	A Complete Multi-Processor System-on-Chip FPGA-Based Emulation Framework	1			1			1		1			1			
	2006	A fast HW-SW FPGA-based thermal emulation framework for multi-processor system-on-chip	1					1	1		1			1			
	2006	Design of a Hierarchy-Bus Based MPSoC on FPGA		1		1						1			1		
	2006	Energy-Aware MPEG-4 Single Profile in HW-SW Multi-Platform Implementation	1			1				1			1		1		
	2006	Evaluating SoC Network Performance in MPEG-4 Encoder	1				1			1				1			
9	2007	A Multiprocessor System-on-Chip Implementation of a Laser-based Transparency Meter on an FPGA	1				1		1				1	1			1
	2007	An FPGA Design Flow for Reconfigurable Network-Based Multi-Processor Systems on Chip		1		1							1	1			
	2007	An FPGA implementation of a snoop cache with synchronization for a multiprocessor system-on-chip		1		1							1	1			
	2007	Automated Integration and Communication Synthesis of Reconfigurable MPSoC Platform		1				1					1	1			
	2007	Design of Heterogeneous MPSoC on FPGA	1			1				1		1			1		
	2007	Efficient External Memory Interface for Multi-Processor Platforms Realized on FPGA Chips		1		1			1					1			
	2007	Feature - NoC emulation- a tool and design flow for MPSoC		1				1			1			1			1
	2007	HETEROGENEOUS MPSOC ARCHITECTURES FOR EMBEDDED COMPUTER VISION	1			1					1			1			
	2007	Multi-Processor System-Level Synthesis for Multiple Applications on Platform FPGA	1					1	1					1			
16	2008	A Multi-MicroBlaze Based SOC System- From SystemC Modeling to FPGA Prototyping		1		1			1					1			
	2008	A Pthreads-based MPI-1 Implementation for MMU-less Machines	1			1			1		1			1			
	2008	An Embedded Dynamically Self-Reconfigurable Master-Slaves MPSoC Architecture	1			1					1			1			
	2008	An ILP formulation for architectural synthesis and application mapping on FPGA-based hybrid multi-processor SOC		1		1			1					1			
	2008	An MPSoC architecture for the Multiple Target Tracking application in driver assistant system	1				1			1					1		
	2008	External DDR2-constrained NOC-based 24-processors MPSOC design and implementation on single FPGA		1		1			1					1			1
	2008	Generic crossbar network on chip for FPGA MPSoCs	1			1					1		1	1			1
	2008	Implementation of a reconfigurable data protection module for NoC-based MPSoCs		1		1			1					1			1
	2008	Micronmesh for fault-tolerant GALS Multiprocessors on FPGA		1		1			1				1		1		1
	2008	MPSoC design of RT control applications based on FPGA SoftCore processors	1				1		1					1			
	2008	Multi-FPGA Emulation of a 48-Cores Multiprocessor		1		1			1					1			1
	2008	Multistage Interconnection Network for MPSoC - Performances study and prototyping on FPGA		1		1							1		1		1
	2008	Performance Evaluation ofFPGA based Crossbar NoC Architecture		1		1						1			1		1
	2008	Run-Time Management of a MPSoC Containing FPGA Fabric Tiles	1					1				1		1			
	2008	System-Level Performance Estimation for Application-Specific MPSoC Interconnect Synthesis		1				1	1					1			
	2008	xENoC - An eXperimental Network-On-Chip Environment for Parallel Distributed Computing on NoC-based MPSoC Architectures		1				1		1					1		1
20	2009	A MPSoC prototyping platform for flexible radio applications	1			1			1		1			1			
	2009	Automatic design methodologies for MPSOC and prototyping on multi-FPGA Platforms		1		1			1					1			1
	2009	Automatic parallelization experiments on 16PE NOC based MPSOC		1		1			1					1			1
	2009	Design of AXI Bus Based MPSoC on FPGA	1			1				1					1		
	2009	Driver assistance system design and its optimization for FPGA based MPSoC		1			1			1					1		
	2009	Emulation-based transient thermal modeling of 2D-3D Systems-On-Chip with active cooling		1				1					1	1			
	2009	Exploring multistage interconnection network implementations for FPGA-based MPSOCs		1		1							1	1			1
	2009	Generic Array-Based MPSoC Architecture	1			1			1		1			1			1
	2009	Inducing Thermal-Awareness in Multicore Systems Using Networks-on-Chip		1				1	1					1			1
	2009	Multiprocessor System-on-Chip Profiling Architecture- Design and Implementation		1		1							1	1			
	2009	Multiprocessor Task Migration Implementation in a Reconfigurable Platform		1		1			1					1			
	2009	Overview of FPGA-Based Multiprocessor Systems			1			1					1			1	
	2009	Performance Evaluation of Cluster-Based Homogeneous Multiprocessor System-on-Chip Using FPGA Device		1		1				1					1		1
	2009	Performance evaluation of concurrently executing parallel applications on multi-processor systems	1			1			1					1			
	2009	Performance measurements of synchronization mechanisms on 16PE NOC based multi-core with dedicated synchronization and data NOC		1		1			1					1			1
	2009	Prototype design of cluster-based homogeneous Multiprocessor System-on-Chip		1		1				1					1		1
	2009	Radar based collision avoidance system implementation in a reconfigurable MPSoC	1			1				1					1		
	2009	Rapid Design of Specific MPSoC prototype within FPGA			1			1		1					1		
	2009	Self-optimization of MPSoCs Targeting Resource Efficiency and Fault Tolerance		1				1					1			1	1
	2009	The design methodology and the implementation of MPSOC based on Delta MINs on FPGA		1		1							1	1			

Abbildung 11: Auflistung der untersuchten Veröffentlichungen mit ihren kategorisierten Inhalten

6.2 Synthese Ergebnisse der Systeme aus dem vierten Kapitel

Grundlage der Auswertung, der in Kapitel 4 vorgestellten Systeme, war neben der Geschwindigkeitserhöhung auch die Verwendung von Systemressourcen. Die Anzahl der verwendeten Slices wurde direkt in die Effizienzbetrachtung einbezogen.

	Slices			LUTs		
	Number	%	Ratio	Number	%	Ratio
1 CPU	1637	4.9	1	2366	3.5	1
2 CPUs	2344	6.9	1.4146	3561	5	1.5051

Tabelle 3: Synthese Ergebnisse von System 1

	Slices			LUTs		
	Number	%	Ratio	Number	%	Ratio
1 CPU	1660	4.9	1	2329	3.4	1
2 CPUs	2341	6.9	1.4102	3539	5.2	1.5195
4 CPUs	3867	11.4	2.3295	6137	9.0	2.6350
8 CPUs	7134	21.1	4.2976	11806	17.4	5.0691

Tabelle 4: Synthese Ergebnisse von System 2

	Slices			LUTs		
	Number	%	Ratio	Number	%	Ratio
1 CPU	3132	30.1	1	4415	20.5	1
2 CPUs	4581	42.6	1.4626	6740	31.3	1.5266
3 CPUs	6041	56.1	1.9288	9112	42.3	2.0639
4 CPUs	7569	70.4	2.4167	11555	53.7	2.6172

Tabelle 5: Synthese Ergebnisse von System 3

	Slices			LUTs		
	Number	%	Ratio	Number	%	Ratio
1 CPU	3493	32.4	1	5071	23.5	1
2 CPUs	4948	46.0	1.4165	7392	34.3	1.4577
3 CPUs	6400	59.5	1.8322	9760	45.3	1.9247
4 CPUs	7921	73.6	2.2677	12199	56.7	2.4056

Tabelle 6: Synthese Ergebnisse von System 4