Rekonfigurierbare DSP-Datenpfaderweiterungen für energieeffiziente, eingebettete Prozessorkerne

Steffen Köhler, Jan Schirok, Rainer G. Spallek

Institut für Technische Informatik
Technische Universität Dresden
D-01062 Dresden
{stk,rgs}@ite.inf.tu-dresden.de

Zusammenfassung

Die Steigerung der Verarbeitungsleistung eingebetteter Mikroprozessoren gewinnt insbesondere durch zunehmende Bedeutung audio-visueller Datenverarbeitung in Verbindung mit drahtloser Kommunikation ständig an Bedeutung. Die notwendige Performance ist jedoch durch Anwendung klassischer Techniken des Prozessorentwurfs (Pipelining, Superskalarität) nur teilweise erreichbar. In unserem Beitrag möchten wir aufzeigen, daß die erforderliche Verarbeitungsleistung durch den Einsatz dynamisch rekonfigurierbarer Datenpfade bei gleichzeitig erhöhtem Flexibilitätsgrad erreicht werden kann. Anhand von quantitativen Untersuchungen zu Chipflächen- und Leistungsbedarf einer 0,18μm CMOS-Standardzellenrealisierung der ARRIVE Architektur-Fallstudie wird ersichtlich, daß durch Einsatz eines einfachen RISC Mikroprozessors erweitert um einen rekonfigurierbaren DSP-Datenpfad eine gute Ausnutzung der vorhandenen Applikationsparallelität verbunden mit einem deutlichem Performancegewinn bei gleichzeitig geringem Chipflächen- und Leistungsbedarf erreichbar ist. Als Quelle des ermittelten und dargestellten Leistungsbedarfs dient dabei eine basierend auf repräsentativen DSP Benchmark-Algorithmen durchgeführte Power-Simulation des Chip-Layouts.

1 Motivation

Der aktuelle Entwicklungstrend im Bereich eingebetteteter Prozessorkerne ist - ebenso wie in der Domäne der Desktop-Prozessoren - gekennzeichnet durch ständig steigende Core-Taktfrequenzen, um die notwendige Performance insbesondere bei der Echtzeitverarbeitung von Datenströmen zu erzielen. Eine etablierte Technik zur Leistungssteigerung durch Taktfrequenzerhöhung stellt dabei die Vergrößerung der Pipeline-Tiefe dar. Diese Technik ist im Bereich eingebetteter Prozessorkerne - bedingt durch den zusätzlichen Chipflächen- und Leistungsbedarf der Pipeline-Register sowie Logik für Forwarding und Branch-Prediction - nur in begrenztem Maße einsetzbar. Gleiches gilt für die Realisierung superskalarer Verarbeitungseinheiten, welche durch Diskrepanzen zwischen vorhandener und nutzbarer Applikationsparallelität häufig nicht effektiv arbeiten können.

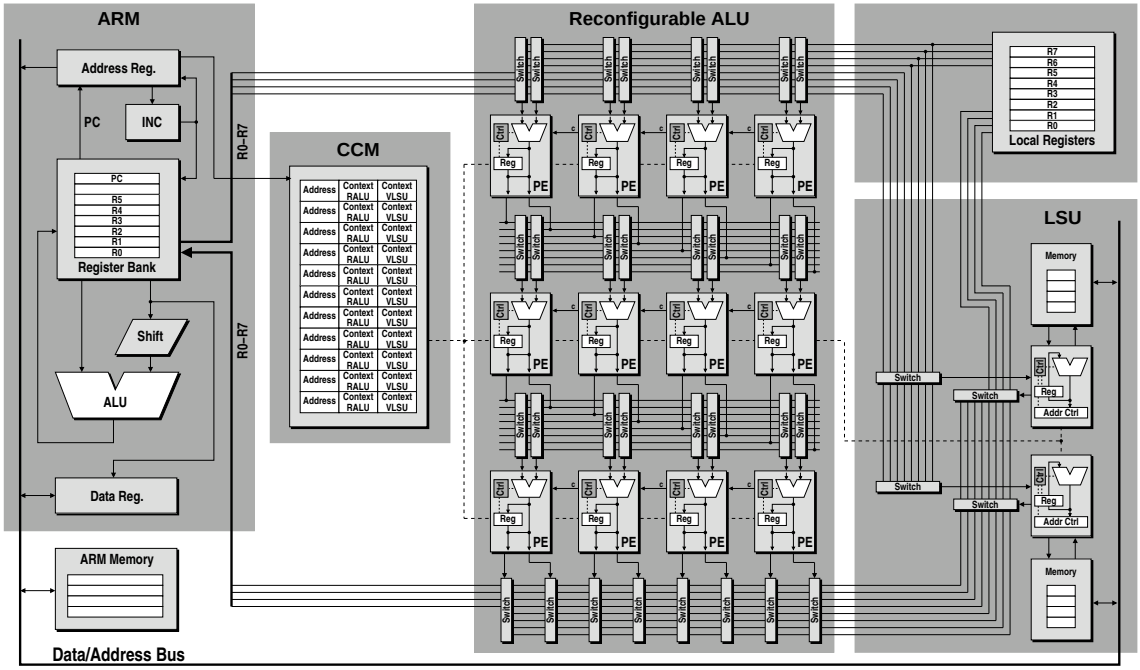

Abbildung 1: Datenpfadarchitektur

Die Hauptzielstellung beim Entwurf bezüglich Performance-, Chipfläche- und Leistungsbedarf effizienter eingebetteter Prozessorkerne besteht daher in einer besseren Ausnutzung der Applikationsparallelität bei gleichzeitig vergleichsweise geringer Pipelinetiefe und Taktfrequenz. Als Nebeneffekt werden die Timing-Anforderungen an die Befehlsspeicherschnittstelle ebenfalls deutlich verringert. Eine vielversprechende Realisierungsvariante stellen dabei Datenpfaderweiterungen dar, welche komplexe arithmetische Operationen - meist aus dem Bereich der digitalen Signalverarbeitung - in einem Prozessorbefehl verarbeiten können (Befehlssatzerweiterung). Diese Erweiterungen werden mittlerweile von nahezu allen Herstellern eingebetteter Prozessoren angeboten [5, 6], jedoch bieten sie aufgrund fest-verdrahteter Operationen und Datenformate meist nur begrenzte Einsatzmöglichen [7]. Des weiteren führt die alleinige Beschleunigung von Verarbeitungsoperationen zu einem Speichertransferengpaß. In unserem Beitrag möchten daher wir ein dynamisch rekonfigurierbares Datenpfadmodell vorstellen, welches in der Lage ist, sowohl Verarbeitungs- als auch Transferoperationen flexibel und effizient auszuführen.

2 Datenpfadmodell

Das Datenpfadmodell umfaßt einen ARMv4-kompatiblen RISC-Prozessorkern [1], der um zwei grobgranular rekonfigurierbare Funktionseinheiten (RFUs) erweitert wurde (Abbildung 1). Die zusätzlichen Datenpfade ermöglichen die optimierte Abarbeitung von DSP-Funktionen. Hauptentwurfsziele waren dabei die Unterstützung von Verarbeitungsparallelität auf arithmetisch-logischer Ebene sowie Transferparallelität. Im einzelnen beinhalten die zusätzlichen Datenpfade ein Kontext-gesteuertes rekonfigurierbares ALU-Feld (RALU) sowie eine Kontext-gesteuerte Load/Store Einheit (LSU). Durch einen zusätzlichen Read/Write-Port ist der DSP-Datenpfad mit den unteren acht Registern des RISC-Kerns gekoppelt, wodurch eine ausreichend enge

Datenpfadkopplung auch in Fällen paralleler Arbeitsweise aller Funktionsblöcke gewährleistet ist (globale Datenregister). Neben dieser Zugriffsmöglichkeit auf die RISC-Prozessorregister verfügen die rekonfigurierbaren Verarbeitungseinheiten über zusätzliche lokale Datenregister. Um der Bildung von Transferflaschenhälsen vorzubeugen, sind alle lokalen Datenregister parallel nutzbar. Das Verarbeitungsprinzip ist dabei eng an das traditioneller VLIW-DSPs angelehnt, stellt jedoch keine klassische Befehlssatzerweiterung dar. Die Spezifikation der Verarbeitungs- und Transferoperationen erfolgt vielmehr in Konfigurationskontexten. Die Abbildung von parallel ausgeführten Operationen auf eine begrenzte Menge von Kontexten ist möglich, da die Anzahl von häufig ausgeführten Operationen in Performance-relevanten Schleifenkonstrukten ebenfalls begrenzt ist. Die Aktivierung der Konfigurationsdaten erfolgt dabei durch einen Konfigurationsmanager (CCM) anhand einer Tabelle, deren Kontexteinträge um ein zusätzliches Adreßfeld erweitert wurden. Durch Vergleich dieser Adressen mit dem Befehlszähler des RISC Prozessors können somit Kontexte für die rekonfigurierbaren Funktionseinheiten (RALU, LSU) separat ausgewählt werden (Content Addressable Memory). Die Realisierung des Datenpfadmodells mittels dreistufiger Pipeline sorgt des weiteren für einen reibungslosen Daten- und Befehlsfluß ohne zusätzlichen Hardwareaufwand durch Forwarding-Notwendigkeit.

Das rekonfigurierbare ALU-Feld besteht aus 4×4 grobgranular Kontext-gesteuert rekonfigurierbaren 16-Bit Verarbeitungselementen (PE). Die Eingangs- und Ausgangssignale der Verarbeitungselemente werden innerhalb des Feldes über konfigurierbare Multiplexer-Schalter horizontal verteilt, während eine vertikale Verbindung nur durch die Verarbeitungselemente selbst möglich ist. Insgesamt kann jedes Verarbeitungselement 48 verschiedene arithmetisch-logische Funktionen ausführen, wobei auch eine Bypass-Operation zur Signalweiterleitung integriert wurde. Eine zusätzliche Carry-Logik dient der Nutzung von verketteten Operationen zu einem Vielfachen der Verarbeitungsbreite eines einzelnen Elements. Zusätzliche Datenregister in den Verarbeitungselementen gestatten die Organisation von Pipelines. Die Quelloperanden (Eingänge der ersten Zeile) des Feldes werden vom lokalen Registersatz über einen Bus zur Verfügung gestellt, während die Zieloperanden (Ausgänge der letzten Zeile) über einen weiteren Bus in diesen zurückgeschrieben werden.

Die Load/Store Einheit (LSU) stellt eine flexible und skalierbare Schnittstelle zu lokalen Datenspeichern zur Verfügung. Sie ermöglicht das parallele Laden und Speichern der lokalen Datenregister über die Quell- und Zielregisterbusse. Spezielle DSP-Adressierungsmodi wie Post-Inkrement/-Dekrement mit Offset und Adreß-Write-Back erleichtern den Transfer von Datenblöcken.

Der Konfigurationsmanager (CCM) vergleicht ständig den Wert des Fetch-Adreßregisters des RISC-Prozessors mit den Adreßeinträgen der Kontexttabelle. Jeder Eintrag spezifiziert dabei einen RALU- sowie einen LSU-Kontext, der parallel zur Ausführung eines RISC-Prozessorbefehls aktiviert wird. Im Falle einer Übereinstimmung werden die Daten zur Aktivierung der Kontexte über spezielle Busse zu den rekonfigurierbaren Einheiten transferiert. Durch die Anwendung dieses Verfahrens werden die beim Start der Applikation in die lokalen Konfigurationsregister geladenen Operationen parallel ausgeführt. Um eine Vereinfachung des Programmiermodells zu erreichen, erfolgt die Aktivierung der Operationen genau wie beim RISC-Prozessor mittels einer dreistufigen Pipeline, womit die Ausführung der rekonfigurierbaren Operationen genau in der Execution-Phase des aktivierenden RISC-Befehles erfolgt.

Die verwendete Realisierungsvariante stellt einen Spezialfall des ARRIVE Architekturmodells dar, welches detailliert in [2] dargestellt ist.

3 ASIC Entwurfsablauf

Der Entwurf wurde mit Hilfe des am Lehrstuhl für Hochparallele VLSI-Systeme und Neuromikroelektronik der TU Dresden vorhandenen Entwurfsflusses und den dort im Rahmen des Europractice Programmes verwendeten ASIC Bibliotheken durchgeführt. Den Ausgangspunkt des ASIC Entwurfes bildete ein synthesefähiges VHDL-Modell, dessen korrekte Arbeitsweise bereits mittels eines FPGA-Prototypen verifiziert wurde [3]. Folgende ASIC Bibliotheken und Entwurfswerkzeuge kamen zum Einsatz:

VHDL-Synthese:	Synopsys Design Compiler
Netzlistensimulation:	Cadence NC-Sim
ASIC-Layout:	Cadence Encounter
Powersimulation:	Cadence Encounter/Synopsys PrimePower
Prozeß:	UMC 6 Metallebenen 0.18μm 1.8V CMOS [8]
Bibliothek:	UMC 0.18μm Standardzellenbibliothek [9]

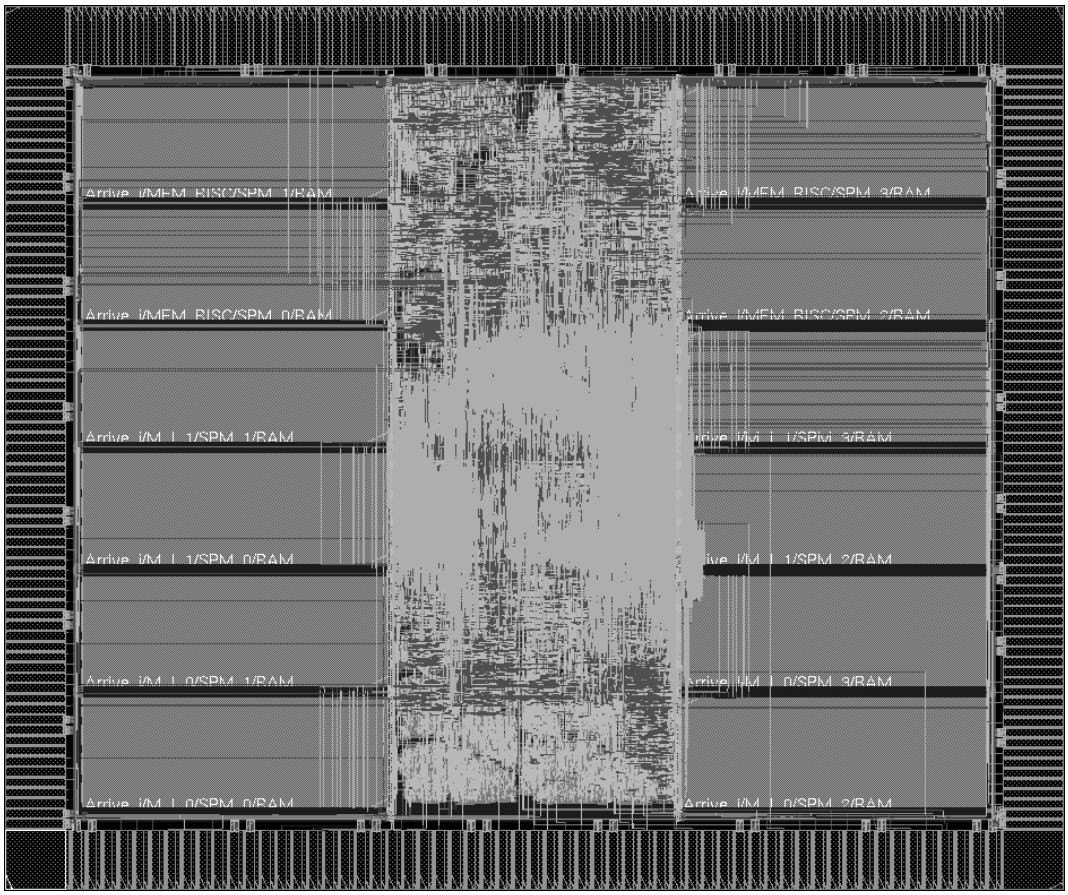

Abbildung 2: ASIC Layout

Das resultierende Layout benötigt eine Chipfläche von 4.9×4.09mm=20mm^2 und ist ein Abbildung 2 dargestellt. Den größten Teil (12mm^2) benötigen die 32-Bit SRAM Speicherblöche für den RISC Controller sowie die Load/Store-Einheiten (je 64 KByte). Die eigentliche Kern-Chipfläche (8mm^2) wird zu 50% vom RALU-Array belegt, während der RISC Controller nur etwa 1mm^2 Chipfläche beansprucht.

4 Leistungsbewertung

Tabelle 1 faßt die Ergebnisse der funktionalen Simulation sowie der Powersimulation zusammen. Die Leistungsverbrauchsdaten des zu Vergleichszwecken herangezogenen, ebenfalls in einem 0,18μm CMOS Prozeß gefertigten ARM922T Prozessorkerns sind aus [4] entnommen. Die dort angeführten Größen stellen Durchschnittswerte dar, während die ermittelten ARRIVE Daten den realen Leistungsverbrauch widerspiegeln. Die maximale möglich Taktfrequenz wird hauptsächlich durch die verwendete Pipelinetiefe sowie durch die Anordnung der Nutzoperationen im RALU-Feld bestimmt. In letzterem Fall läßt sich die ungünstigste Anordnung (längster kritischer Pfad) durch zeilenweise Anordnung von Operation mit Übertragsnutzung innerhalb der Verarbeitungselemente konstruieren. Eine solche Anordnung ist für die im Rahmen dieses Beitrags untersuchten Benchmark-Algorithmen jedoch nicht relevant, sodaß die in den Verarbeitungselementen vorhandenen Pipeline-Register nicht zur Anwendung kommen.

Benchmark	Taktzyklen		Leistungsverbrauch (mW)	
	ARM922T	ARRIVE	ARM922T	ARRIVE
FIR Filter (Tap)	9.5	0.25	≈ 200	591
IIR Filter (Biquad)	59.7	2	≈ 200	656
FFT Radix-2 (512pt.)	99608	5100	≈ 200	523

Tabelle 1: Benchmark: Taktzyklen und Leistungsverbrauch

Die in der Timing-Simulation des Chip-Layouts (Netzliste mit Timing Back-Annotation) erreichte ARRIVE Taktfrequenz lag bei 100MHz. Unter der Annahme, daß der ARM922T Prozessorkern mit 200MHz betrieben werden kann [4], dann ergibt sich daraus ein 9 bis 19facher Performancegewinn zugunsten des ARRIVE Chip-Layouts. Entsprechend den Angaben in [4] kann weiterhin eine näherungsweise Verdopplung des Chipflächenbedarfes beim vorliegenden ARRIVE Layout (20 mm^2) gegenüber dem ARM922T Kern (8.1 mm^2) angenommen werden.

5 Zusammenfassung und Ausblick

Ziel unseres Beitrag war das Aufzeigen der Leistungsfähigkeit von dynamisch rekonfigurierbaren DSP-Datenpfaden bei gleichzeitig geringem Ressourcenbedarf. Durch die simulative Evaluation eines auf einem 0.18μm CMOS-Prozeß basierenden Chip-Layouts der ARRIVE Architektur konnten dabei Erkenntnisse zum Chipflächenbedarf und Leistungsverbrauch gewonnen werden. Trotz einer durch die begrenzen Anzahl getesteter Benchmark-Applikationen eingeschränkten Aussagekraft der Benchmark-Ergebnisse kann aufgrund vorangegangener Untersuchungen [3] davon ausgegangen werden, daß der erzielbare Performancegewinn durch den Einsatz dynamisch rekonfigurierbarer DSP-Datenpfade um ein vielfaches größer als der Anstieg des Chipflächenbedarfes

und des Leistungsverbrauches ist. Weitergehende Arbeiten haben daher vor allem eine detailliertere Untersuchung weiterer Applikationen zum Ziel.

6 Danksagung

Die Autoren danken den Mitarbeitern des Lehrstuhles für Hochparallele VLSI-Systeme und Neuromikroelektronik der TU Dresden für die fachliche und infrastrukturelle Unterstützung des ASIC-Entwurfsprozesses.

Literatur

[1] S. Furber. ARM System-on-Chip Architecture. Addison-Wesley, 2000.

[2] M. Zabel, S. Köhler, M. Zimmerling, T. B. Preußer, R. G. Spallek. Design Space Exploration of Coarse-Grain Reconfigurable DSPs. In: Proceedings of the International Conference on Reconfigurable Computing and FPGAs (ReConfig '05). IEEE Press, 2005

[3] S. Köhler, M. Zabel, M. Zimmerling, T. B. Preußer, R. G. Spallek. ARRIVE - eine rekonfigurierbare DSP-Architektur mit geringem Rekonfigurationsoverhead. Dresdner Arbeitstagung Schaltungs- und Systementwurf (DASS2006).

[4] AMI Semiconductor. AMI Semiconductor to License ARM7 and ARM9 Microprocessor Cores, Provide Foundry Services. Design & Reuse Headline News. Las Vegas, 2001 http://www.us.design-reuse.com/news/news605.html

[5] S. Fuller. Motorola's Altivec Technology. Seminconductor Product Sector, Austin, TX, 1998.

[6] Silicon Graphics Inc. MIPS Digital Media Extension, 1997

[7] D. Talla,L. K. John,D. Burger. Bottlenecks in Multimedia Processing with SIMD Style Extensions and Architectural Enhancements. IEEE Transactions on Computer, Vol. 52, Issue 8, 2003.

[8] UMC. The UMC 0.18μm CMOS SoC Process. http://www.umc.com/English/process/d.asp

[9] Faraday Technology Corporation. UMC Free Library. http://freelibrary.faraday-tech.com/ips/018library.html