

Wenn mehrere Herzen schlagen

Soft-Core-Multiprozessor-Lösungen zur Implementierung in FPGAs

FPGA-basierende Embedded-Systeme, die sich mit einer Vielzahl von Soft-Core-Prozessoren ausstatten lassen, eröffnen Systementwicklern neue technische Möglichkeiten. Damit können nicht nur ASIC-Entwickler ihre System-on-Chips mit speziellen Funktionen auf optimale Leistung trimmen, sondern auch Anwender programmierbarer Logik. Die Aufteilung der Verarbeitungslast einer Applikation auf mehrere Soft-Cores bringt nicht nur mehr Flexibilität beim Entwurf mit sich, sondern auch die Möglichkeit, mit steigenden Leistungsanforderungen mithalten zu können.

Von Bob Garrett

Entwickler von Embedded-Systemen können durch den Einsatz von mehreren Soft-Core-Prozessoren die Gesamtsystemleistung steigern oder auch Aufgaben aufteilen und damit gegebenenfalls einen Standard-Prozessor entlasten. Typischerweise werden mit 400 bis 800 MHz getaktete diskrete Prozessoren eingesetzt, um die unzähligen Aufgaben – relativ einfache wie auch sehr aufwendige – abzuwickeln. Indem die Aufgaben nach Zeit- und Leistungs-Anforderungen partitioniert werden, ermöglichen mehrere Soft-Prozessoren eine effizientere Nutzung der Rechenleistung, während zusätzlich die gleiche oder sogar eine höhere Gesamtverarbeitungsleistung erreicht wird.

Die Anzahl von Soft-Core-Prozessoren, die in einem einzigen FPGA implementiert werden kann, hängt nur ab

von den Ressourcen des Bausteins (zum Beispiel Logik und Speicher). Komplexe FPGAs beispielsweise können prinzipiell Hunderte von Soft-Core-Prozessoren integrieren. Es lassen sich auch verschiedene Soft-Core-Prozessoren – 16 oder 32 bit – implementieren, um ein Design in Bezug auf Rechenleistung, Logikfläche, Leistungsaufnahme etc. zu optimieren.

Codierungsalgorithmen können auf mehrere Prozessoren verteilt werden, je nachdem, welche Aufgaben zusammengehören. Und zeitkritische Tasks lassen sich dedizierten Prozessoren zuweisen, während weniger anspruchsvolle Aufgaben von einer oder mehreren anderen CPUs übernommen werden. Diese Flexibilität ermöglicht eine logische Gruppierung der Aufgaben und somit höhere Leistungsniveaus bei gleichzeitig reduzierter Taktfrequenz, was wiederum die Leistungsaufnahme des Systems senkt.

Embedded-Prozessoren in FPGAs

Heute stehen dem System-Entwickler Embedded-IP-Funktionen (Intellectual Property) wie CPUs, Signalverarbeitungseinheiten, Peripherie und standardmäßige Kommunikationsschnitt-

stellen, die speziell für FPGAs entwickelt wurden, zur Verfügung. Der Multiprozessor-Ansatz senkt die Bauelementekosten, indem mehrere Prozessorkerne in ein FPGA integriert werden, wodurch z.B. die Leiterplatte kleiner werden kann. Damit sind auch weniger Signalleitungen zwischen den Prozessoren auf Leiterplattebene erforderlich, und diese einfacheren Prozessorkerne laufen noch dazu mit einer niedrigeren Taktfrequenz. Beides führt dazu, dass auch die Anzahl der Leiterplatten-Schichten reduziert werden kann.

Wenn eine Aufgabe auf mehrere Prozessoren aufgeteilt werden kann, ist es einfacher, die einzelnen Programme zu schreiben, zu debuggen und zu warten. Damit sind enorme Kosteneinsparungen möglich. Die Code-Entwicklung sowie das Debuggen gehen schneller, und wenn das Produkt eine gewisse Reife erreicht hat, kann der Code leichter modifiziert werden, weil er wesentlich einfacher zu analysieren ist.

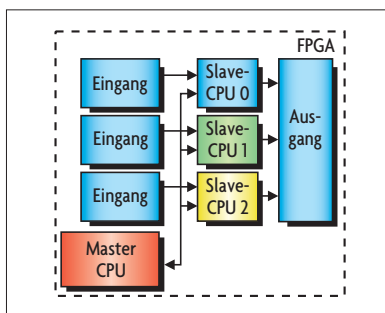
Multikanal-Anwendungen

Multikanal-Anwendungen können für den erforderlichen Systemdurchsatz mit Hilfe von mehreren Prozessoren auf einem Chip skaliert werden, wobei jeder Prozessor einen Teil des gesamten Kanaldurchsatzes übernimmt. Jeder Prozessor kann entweder genau den gleichen Code abarbeiten oder einige können die Algorithmen „on the fly“ wechseln, um sich den Systemanforderungen anzupassen. In manchen Fällen ist zusätzlich ein Master-Prozessor vorhanden, der die allgemeinen Aufgaben wie Systeminitialisierung, Statistik und Fehlerbehandlung übernimmt (Bild 1).

Serielle Verbindung von Prozessorkernen

Sind einzelne Prozessoren seriell in einer Kette verbunden, behandeln Systemarchitekten jeden einzelnen als eine Stufe in einer größeren Verarbeitungs-Pipeline (Bild 2). Jede CPU ist verantwortlich für einen Teil der gesamten Verarbeitungsaufgabe und jede CPU hat Zugriff auf den Datenspeicher (arbitrierte oder dedizierte Speicherschnittstellen, wenn der Speicher extern ist, oder Dual-Ported-Speicher, wenn er sich auf dem Chip befindet),

Bild 1. Für einen höheren Systemdurchsatz lassen sich bei Multikanal-Anwendungen mehrere Soft-Core-Prozessoren auf einen Chip integrieren.



um die Ergebnisse vom Ausgang der einen Stufe zum Eingang der nächsten zu „schieben“.

Prozessor-Companion

Diskrete Prozessor- und DSP-Chips, die mit einem FPGA verbunden sind, können von einer Hardware-Beschleunigung, Peripherie-Erweiterung oder Interface-Bridge profitieren, egal ob das FPGA eine CPU enthält oder nicht. Diverse Schnittstellen-IPs für die Chip-zu-Chip-Verbindung sind verfügbar, die den externen Zugriff auf Peripherie, Beschleunigungslogik und I/O-Schnittstellen in einem FPGA ermöglichen.

Erweiterung des Befehlssatzes

Diverse Prozessor-IP-Anbieter und FPGA-Implementierungen bieten die Möglichkeit, den Befehlssatz des Prozessors hardwaremäßig mit mächtigeren Befehlen für applikationsspezifische Algorithmen zu erweitern. Mit Hilfe der gängigen Lese-/Speicher-Operationen kann ein kundenspezifischer Logikblock mit Daten gefüttert werden, der wiederum zu einem Teil der Prozessor-ALU wird (Bild 3). In einigen Fällen können kundenspezifische Instruktionen Multizyklus-Operationen unterstützen und Zugang zu anderen Systemressourcen wie FIFOs und Pufferspeicher bieten. Typische Anwendungen kundenspezifischer Instruktionen sind beispielsweise Bit-Manipulationen oder komplexe numerische und logische Operationen.

Im Gegensatz zu einem Algorithmus, der mit Hilfe der normalen ALU-Ressourcen abgearbeitet wird, können kundenspezifische Instruktionen signifikante Leistungsvorteile bringen – und das, obwohl die normalen Lese- und Speicheroperationen des Prozessors genutzt werden. Ein Cyclic-Redundancy-Check-Algorithmus (CRC) für einen 64-K-Block mit einer kundenspezifischen Instruktion, die nur wenige Logikelemente benötigt, kann z.B. um den Faktor 27 schneller ausgeführt

werden als mit herkömmlicher Software. Auf diese Weise lässt sich im System auch die Leistungsaufnahme reduzieren: Die gleichen Ergebnisse werden in gleicher Zeit, aber mit geringerer Taktfrequenz erreicht.

Wie Entwickler kundenspezifische Instruktionen erstellen können, hängt vom jeweiligen Prozessor-IP ab. Auf der einen Seite des Spektrums muss der Anwender einen neuen Compiler erzeugen, wenn er kundenspezifische

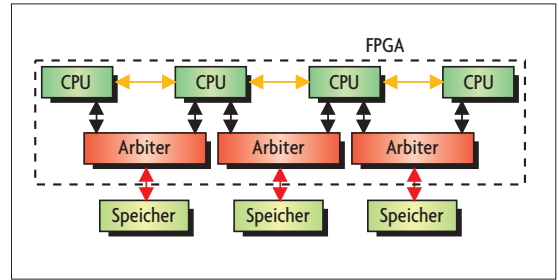


Bild 2. Sind einzelne Prozessoren seriell in einer Kette verbunden, behandeln Systemarchitekten jeden einzelnen als eine Stufe in einer größeren Verarbeitungs-Pipeline.

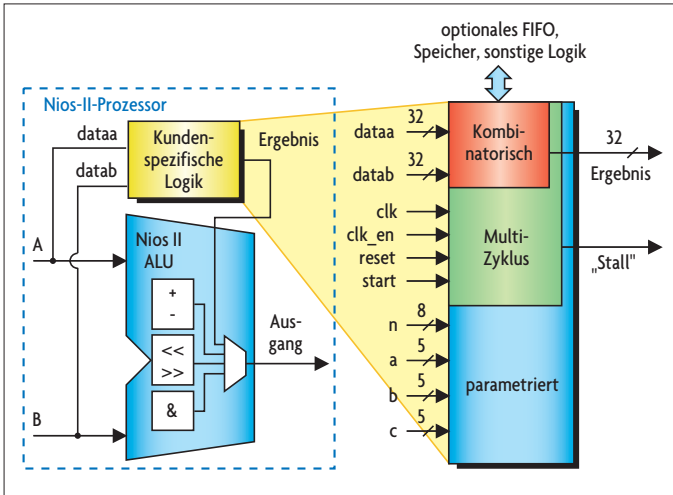


Bild 3. Der Embedded-Soft-Core-Prozessor Nios II kann mit bis zu 256 anwenderspezifischen Befehlen erweitert werden. Damit lassen sich insbesondere zeitkritische Aufgaben lösen.

Befehle hinzufügen will. Dieser kundenspezifische Compiler leitet dann die Aufrufe kundenspezifischer Instruktionen auf Basis einiger Applikationskriterien ab. Ein einfacherer, eleganterer Ansatz besteht darin, die Instruktion im C-Code aufzurufen, genauso wie eine normale Subroutine aufgerufen werden würde.

► **Hardware-Beschleuniger für rechenhungrige Anwendungen**

Andere Ansätze zur Steigerung der Gesamtsystemleistung bestehen zum Beispiel darin, Hardware-Beschleuniger (oder Coprozessoren), Companion-Chips für Prozessoren oder kundenspezifische System-on-Chips zu nutzen. Anders als kundenspezifische Instruktionen arbeiten Hardware-Beschleuniger als unabhängige Logikmodule, die ihre Anweisungen von der Embedded-CPU erhalten und ganze Daten-Puffer ohne Eingriff der CPU abarbeiten. Vereinfacht dargestellt, verfügt ein Hardware-Beschleuniger über zwei halbe DMA-Kanäle (einen, um die Eingangsdaten zu lesen, den anderen, um die Ergebnisse zu speichern) und ein Steuerungs-Interface für die CPU, damit sie die Einheit während des Betriebs aufsetzen, starten, stoppen und „pollen“ kann (Bild 4). Hardware-Beschleuniger sind prädestiniert für Aufgaben, bei denen große Datenblöcke involviert sind und das Laden der Daten und das Speichern der Ergebnisse durch die CPU eventuell zu einem Engpass führen könnten. Sie können gegenüber der Software-Variante die Leistung um einige Größenordnungen steigern.

► **Kundenspezifische Anpassung im Feld**

Muss ein Produkt schnell auf den Markt gebracht werden, hat das oft zur Folge, dass die erste ausgelieferte Version weniger Funktionen umfasst und mit einer geringeren Leistung läuft als es ursprünglich geplant war. Die System-Firmware zu erneuern, um Fehler zu beseitigen, Funktionen hinzuzufügen und/oder die Leistung zu verbessern, ist da-

her heute eine gängige Praxis. Systeme, die mit programmierbarer Logik aufgebaut sind, bieten dagegen die Möglichkeit, auch die Hardware-Charakteristik relativ einfach zu ändern. Denn der gleiche Flash-Speicher, in dem die System-Software gespeichert ist, kann auch die Konfiguration für ein oder mehrere FPGAs speichern. Ein Prozessor in einem FPGA übernimmt dann die Aufgabe, den Flash-Speicher mit der Systemkonfiguration neu zu beschreiben. Anschließend kann er eine Rekonfiguration initiieren, die die System-Hardware in Millisekunden auf den neuesten Stand bringt.

Bei netzwerkfähigen Produkten lässt sich die Hardware auch über das Internet rekonfigurieren, um beispielsweise neue Funktionen zu implementieren, die Leistung zu steigern oder Fehler zu beseitigen. Das war vor ein paar Jahren noch ein Novum, heute ist es bei FPGA-basierenden Designs weit verbreitete Praxis. Fehlersichere Entwürfe, die auf mehreren, im Flash-Speicher abgelegten FPGA-Konfigurationen basieren, wobei im Fall eines Datenfehlers ein standardmäßiges, sicheres Design geladen wird, garantieren einen funktionierenden Systembaustein. Das heißt, dass im schlimmsten Fall – wenn der Baustein an der Neukonfiguration scheitert – das standardmäßige Design geladen, ein Report über den Fehler geschrieben wird und das System mit der alten Konfiguration weiterläuft.

► **Sicherer Systemaufbau**

Prozessoren müssen mit dem Speicher, der Peripherie, Coprozessoren und ex-

ternen Bausteinen kommunizieren. Software-Treiber nutzen verfügbare Mail-Boxen, damit Multiprozessorsysteme über geteilte Speicher kommunizieren können. Verschiedene Blöcke komplexer IP-Funktionen zu verbinden, kann zu Problemen führen, die die Systementwicklungszeit und die Leistungsfähigkeit negativ beeinflussen können. Zu den hierzu üblicherweise benötigten Logikfunktionen gehören Adress-Decodierer, Daten-Multiplexer, Interrupt-Controller, „Wait-State“-Generatoren, Arbitrierungslogik, Busbreitenanpassung, Signal-Timing und Pegelanpassung sowie Takt-domänen-spezifische „Crossing“-Logik.

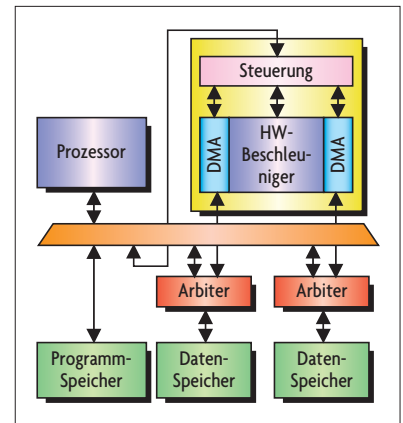


Bild 4. Hardware-Beschleuniger sind prädestiniert für Aufgaben, bei denen große Datenblöcke verarbeitet werden müssen und das Laden der Daten und das Speichern der Ergebnisse durch die CPU eventuell zu einem Engpass führen könnten.

Erfahrene Hardware-Entwickler können diese Module zwar leicht selbst entwickeln, aber solche eher einfachen Funktionen per Hand in Verilog oder VHDL zu codieren, ist eine Verschwendung wertvoller Entwicklungsressourcen. Routine-Aufgaben wie diese sind perfekte Kandidaten für die Computer-Automatisierung. Moderne Systemerzeugungs-Tools für FPGAs übernehmen diese Aufgaben problemlos und machen es möglich, dass sich Entwickler auf die Aspekte ihrer Projekte konzentrieren können, mit denen sie sich vom Wettbewerb abheben wollen.

In traditionellen Embedded-Systemen teilen sich die Prozessoren einen Systembus mit anderen Master-Komponenten, die in periodischem Abstand immer wieder den alleinigen Zugriff auf den geteilten Bus fordern. Diese

Verbindungstopologie stellt einen potentiellen Engpass dar, nämlich genau dann, wenn zwei Master gleichzeitig auf den Bus zugreifen müssen. FPGAs mit einem hohen Maß an Routing-Ressourcen ermöglichen Konzepte, bei denen die verschiedenen Master über dedizierte, gemultiplexte Datenpfade zu den Slave-Komponenten verfügen.

Der Wechsel von einer Master- zu einer Slave-seitigen Arbitrierung ermöglicht den Ablauf gleichzeitiger Transaktionen, außer zwei Master versuchen gleichzeitig, auf eine Slave-Komponente zuzugreifen (Bild 5). Interne Schalt-Matrix-Elemente steuern alle Verbindungen zwischen den Mastern und Slaves mit unterschiedlichen Busbreiten, mit unterschiedlichen Takt-Domänen und Verzögerungszeiten. Mit dieser „Switch“-artigen Topologie können alle Master mit ihren dedizierten Slave-Komponenten gleichzeitig kommunizieren, was den gesamten Systemdurchsatz enorm erhöhen kann.

gs

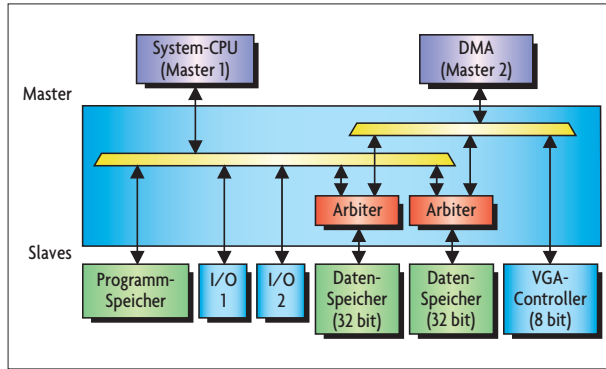


Bild 5. FPGAs mit umfangreichen Routing-Ressourcen ermöglichen Konzepte, bei denen verschiedene Master über dedizierte, gemultiplexte Datenpfade mit den Slave-Komponenten kommunizieren können.

Literatur

- [1] Pocock, G.: Ein starkes Team. Cyclone-II-FPGAs mit dem Soft-Core-Prozessor Nios II ermöglichen kostengünstige und gleichzeitig leistungsfähige Systeme. *Elektronik* 2004, H. 15, S. 40ff.
- [2] Homepage von Altera: www.altera.com



Bob Garrett

ist Senior Marketing Manager für die Nios-Soft-Core-Prozessoren bei der Altera Corporation in San Jose, USA. Bevor er 1999 zu Altera kam, hatte er verschiedene Vertriebs- und Marketing-Positionen bei Fluke Manufacturing und Boulder Creek Engineering inne.
 ▶ E-Mail: BGarrett@altera.com