FPGAs im Rechenzentrum

Marco Platzner · Christian Plessl

Einleitung

Field Programmable Gate Arrays (FPGA) sind integrierte elektronische Bausteine, die durch Programmierung beliebige digitale Schaltungen realisieren können. Die Programmierung erfolgt dabei auf der strukturellen Ebene durch eine anwendungsspezifische Verschaltung der physikalisch auf dem FPGA vorhandenen Hardwareelemente. Dieser Vorgang wird als *Konfiguration* bzw. *Rekonfiguration* bezeichnet.

Das Konzept flexibler Hardware geht auf Gerald Estrin zurück, der in den 1960er-Jahren an der UCLA den Vorschlag machte, Computersysteme mit festen und variablen Anteilen auszustatten [4]. Die variablen Anteile waren Module für arithmetische Operationen und die Konfiguration erfolgte manuell durch Einstecken und Entfernen von Modulen in das Motherboard des Rechners.

Erst Mitte der 1980er-Jahre wurde von Ross H. Freeman von der Firma Xilinx eine praktikable Technologie zur Automatisierung des Ansatzes von Estrin als FPGA vorgestellt und patentiert [5]. In einem FPGA werden konfigurierbare Logikblöcke („configurable logic blocks“, CLB) in einer zweidimensionalen Matrixstruktur angeordnet und mit einem ebenfalls konfigurierbaren Verbindungsnetzwerk untereinander sowie mit externen Anschlüssen verbunden (siehe Abb. 1). Die CLBs verwenden Lookup Tables (LUT), um beliebige Logikfunktionen zu realisieren. Sowohl die Logikfunktionen als auch die Verbindungen zwischen den CLBs werden durch das Schreiben von SRAM-Zellen festgelegt. Da SRAM eine volatile Speichertechnologie ist, muss ein FPGA beim Einschalten jeweils neu konfiguriert werden.

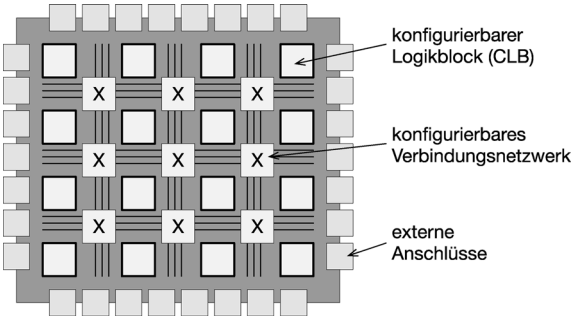

Abb. 1 Grundlegende Struktur eines FPGAs

Anfänglich wurden FPGAs vor allem als programmierbare Logikbausteine in der Elektronikentwicklung genutzt. Mit fortschreitender Integrationsdichte wurden rasch neue Möglichkeiten erschlossen, z. B. zur Emulation digitaler Systeme oder als Ersatz für anwendungsspezifische integrierte Schaltungen (ASIC). Die Volatilität der SRAM-basierten FPGAs ermöglicht aber auch die Rekonfiguration im laufenden Betrieb und damit einen wesentlich disruptiveren Einsatzbereich: anwendungsspezifisch rekonfigurierbare Rechnersysteme. Dieser als *Reconfigurable Computing* bekannt gewordene Ansatz kombiniert die Leistungsfähigkeit spe-

https://doi.org/10.1007/s00287-019-01187-w

Marco Platzner · Christian Plessl
Institut für Informatik und Paderborn Center for Parallel Computing (PC^2),
Universität Paderborn, Paderborn
E-Mails: platzner@upb.de, christian.plessl@uni-paderborn.de

Alle „Aktuellen Schlagwörter“ seit 1988 finden Sie unter:
http://www.is.informatik.uni-wuerzburg.de/as

Published online: 17 July 2019

zialisierter Hardware mit der Flexibilität von Software.

Zu Beginn der 1990er-Jahre wurden mehrere Forschungsprojekte gestartet, um das Potenzial des Reconfigurable Computing auszuloten. Ein prominentes Beispiel ist das DECPerLe-1-System. Damit konnten ausgewählte Anwendungen um mehrere Größenordnungen schneller ausgeführt werden als mit dem schnellsten Supercomputer jener Zeit – bei einem Bruchteil der Kosten und elektrischer Energie [1]. Die Nutzung dieses Potenzials erforderte allerdings sehr hardwarenahe Entwurfsmethoden und Expertenkenntnisse in digitaler Schaltungstechnik.

Nach diesen anfänglichen Erfolgen fristeten FPGAs in Rechenzentren lange ein Nischendasein. Erst durch das Ende der Dennard-Skalierung [2] und dem sich abzeichnenden ökonomischen Ende des Moore'schen Gesetzes hat das kommerzielle Interesse an FPGAs und damit deren Verbreitung in großen Rechenzentren stark zugenommen, da mit FPGAs durch Spezialisierung weiter erhebliche Leistungs- und Energieeffizienzgewinne erzielbar sind.

Wie CPUs sind FPGAs programmierbar und als General-Purpose-Devices von vielen Kunden einsetzbar. Die daraus resultierenden hohen Stückzahlen ermöglichen es den Herstellern, FPGAs in den jeweils modernsten Halbleitertechnologien zu realisieren. Aktuelle FPGAs werden z. B. in 14-nm(Intel)- und 16-nm(Xilinx)-Technologien gefertigt. Obwohl FPGA-Schaltungen um eine Größenordnung langsamer getaktet werden als CPUs, kann man Leistungsgewinne durch die massiv parallele Hardwareumsetzung der Anwendung erzielen.

Der Spagat zwischen der Notwendigkeit einer starken Spezialisierung der FPGA-Architekturen für eine konkrete Anwendung oder Anwendungsklasse und dem Wunsch nach abstrakteren und produktiveren Entwicklungswerkzeugen prägt die Forschung und Praxis des Reconfigurable Computing bis heute. Im Folgenden fassen wir die relevantesten Entwicklungen für die Nutzung von FPGAs im Rechenzentrum zusammen und gehen auf neue Architektureigenschaften und die Programmierung ein. Zum Abschluss skizzieren wir aktuelle Forschungsfragen und geben einen Ausblick.

Architekturen

Die Logikkapazitäten von FPGAs sind in den letzten 25 Jahren sehr stark gewachsen. Moderne FPGA-Architekturen verwenden prinzipiell noch immer die in Abb. 1 dargestellte CLB-Struktur, integrieren aber zusätzlich weitere Funktionsblöcke. Für die Anwendung in Rechenzentren sind vor allem folgende drei Typen relevant:

- *Digital Signal Processing (DSP) Blöcke*: Arithmetische Komponenten lassen sich zwar mit CLBs implementieren, führen aber insbesondere für größere Wortbreiten zu beträchtlichem Ressourcenbedarf und hoher Latenz. Zur effizienteren Signalverarbeitung wurden daher DSP-Blöcke eingeführt, die auf breiteren Worten (typ. 18–27 Bit) Multiplikationen und Akkumulationen durchführen können. Neueste FPGAs von Intel unterstützen sogar Fließkomma-Arithmetik in ihren DSP-Einheiten. Durch die Integration mehrerer tausend unabhängiger DSP-Blöcke können heutige FPGAs dadurch mit der Fließkomma-Rechenleistung von High-End-CPUs mithalten.
- *Integrierte RAM-Blöcke*: Um die in Hardware implementierten Rechenpfade mit Daten zu versorgen, reicht die Bandbreite von externem DRAM-Speicher häufig nicht aus. Daher verfügen heutige FPGAs über zahlreiche, unabhängig nutzbare SRAM-Blöcke mit einer Kapazität von typ. 1–36 KB. Diese RAMs können sehr flexibel konfiguriert und aggregiert werden, um eine große Palette von anwendungsspezifischen Speicherarchitekturen zu implementieren, z. B. Scratchpad RAM, FIFO-Puffer für Datenströme oder Lookup-Tabellen. Insgesamt stehen auf aktuellen FPGAs einige Dutzend MB an Speicher mit einer aggregierten Datenrate im Bereich von 10 TB/s zur Verfügung.
- *Serielle Transceiver*: Um FPGAs in Rechenknoten und Cluster einzubinden, stehen konfigurierbare Transceiver für schnelle serielle Verbindungen zur Verfügung. Damit lassen sich mit wenigen externen Komponenten sehr unterschiedliche Kommunikationsverbindungen mit typ. 10–100 Gbit/s realisieren, z. B. PCIe, Ethernet oder optische Punkt-zu-Punkt-Verbindungen.

In der Regel ist es aus Effizienz- und Kostengründen nicht sinnvoll, komplette Anwendungen auf FPGAs auszuführen. Stattdessen werden rechen-

intensive Programmteile auf FPGAs beschleunigt, während die übrigen auf den Host-CPUs verbleiben. Die effiziente Host-FPGA-Kopplung war daher von Anbeginn eine zentrale Frage im Reconfigurable Computing.

Als geläufigste Form der Kopplung kommen seit jeher die jeweils aktuellen standardisierten Peripheriebusse zum Einsatz (z. B. TURBOchannel, SBus, PCI, PCI-X, PCIe). Peripheriebusse stellen allerdings häufig einen Engpass für datenintensive oder latenzkritische Anwendungen dar.

Um das Jahr 2005 herum haben mehrere HPC-Systemanbieter FPGAs als interessante Technologie identifiziert und enger gekoppelte Systeme speziell für den Rechenzentrumsmarkt entwickelt. Dazu wurden FPGAs direkt an Multiprozessorbusse von CPUs (z. B. NUMAlink bei SGI RASC oder Hypertransport bei XtremeData XD1000) angebunden, wodurch sich ein Zugriff auf gemeinsamen Speicher mit hoher Bandbreite realisieren lässt. Ein alternativer Ansatz ist die Anbindung an die Hochgeschwindigkeitsnetzwerke von HPC-Clustern (z. B. RapidArray in der Cray XD1). Diese Lösungen waren aber durch die Zweckentfremdung von Schnittstellen, die nicht für die Anbindung von Beschleunigern konzipiert waren, technologisch fragil und als proprietäre Technologien unverhältnismäßig teuer.

Die effizientesten Kopplungen eines FPGA an die Host-CPU sind die Integration als rekonfigurierbare funktionale Einheit direkt im Datenpfad der CPU oder als System-in-Package-Coprozessor. Beide Ansätze wurden als Prototypen oder Produktvarianten evaluiert, z. B. der Stretch-S5-Prozessor mit rekonfigurierbaren Funktionseinheiten oder die On-Package-Integration der Intel-Xeon-Broadwell-CPU mit einem Arria 10 FPGA. Diese Systeme konnten sich bisher aus technischen und wirtschaftlichen Gründen nicht durchsetzen.

Mit der zunehmenden Verbreitung von FPGAs in Rechenzentren geht der Trend klar zur Nutzung standardisierter Kommunikationsschnittstellen für die FPGA-Host-Kopplung, aktuell PCIe. In Zukunft werden voraussichtlich neue Schnittstellen zum Einsatz kommen, die gerade von Industriekonsortien standardisiert werden, um den wachsenden Anforderungen von Beschleunigern und neuen Speichertechnologien im Rechenzentrum gerecht zu werden (z. B. CCIX, Gen-Z oder OpenCAPI, CXL). Teilweise sind diese Schnittstellen rückwärtskompatibel mit PCIe und um zusätzliche Funktionalitäten erweitert, z. B. den Zugriff auf einen gemeinsamen Adressraum oder die Speicherkohärenz.

Programmierung

Für eine produktive Programmierung von FPGAs im Rechenzentrum sind heute Hochsprachenansätze unverzichtbar. Dabei wird in einer höheren Programmiersprache das gewünschte Verhalten des FPGA-Beschleunigers beschrieben und mit einem Hardwaresynthesewerkzeug in eine Schaltung übersetzt. Die Herausforderung bei der Hardwaresynthese besteht darin, Datenpfade zu erzeugen, die tausende parallel arbeitende Recheneinheiten und Speicherblöcke in jedem Taktschritt sinnvolle Arbeit verrichten lassen. Zusätzlich muss auf die Balance zwischen Berechnungen und Datenkommunikation geachtet werden, d. h. die Datenpfade müssen geeignet repliziert werden, sodass die verfügbaren externen Speicher- und Kommunikationsdatenraten maximal ausgenutzt werden, ohne Engpässe zu schaffen.

Im Gegensatz zu CPUs oder GPUs, die eine daten- oder thread-parallele Verarbeitung unterstützen, ist für FPGAs häufig tiefes Pipelining die bevorzugte Technik zur Erzielung einer hohen Leistung und Energieeffizienz. Abbildung 2

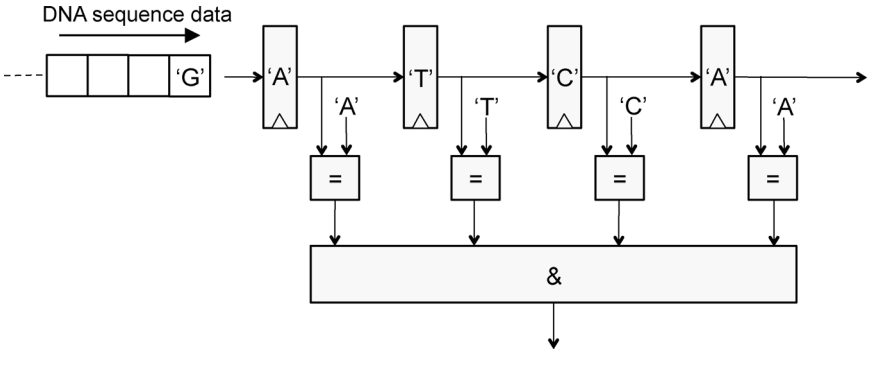

Abb. 2 Anwendungsspezifische Hardwarearchitektur

zeigt als Beispiel eine Architektur zum Erkennen des Substrings „ATCA“ in einer DNA-Sequenz. Die DNA-Basen werden dabei schrittweise in die Pipeline geschoben, wo parallel arbeitende Komparatoren auf Übereinstimmung mit den gesuchten Basen prüfen. Neben der (für längere Substrings sehr tiefen) Pipeline zeigt dieses Beispiel auch die weiteren Charakteristiken anwendungsspezifischer Hardware, wie spezialisierte Datentypen und Operatoren, parallele Ausführung von Operationen, breite Datenpfade und anwendungsspezifische Verbindungsnetzwerke und Speicherarchitekturen.

Für die Hochsprachenprogrammierung von FPGA-Beschleunigern im Rechenzentrum werden heute vornehmlich C-Dialekte eingesetzt, die um Compilerdirektiven zur Steuerung der Hardwaregenerierung erweitert werden. Insbesondere OpenCL findet zunehmend Verbreitung. OpenCL bietet ein etabliertes Beschleunigerprogrammiermodell an und unterstützt zahlreiche für FPGAs relevante Konzepte, z. B. die Unterscheidung von lokalem und globalem Speicher, explizite Vektordatentypen und explizite Datenströme zwischen lose gekoppelten, parallel ausgeführten Kernels.

Ausblick und Forschungsfragen

Seit Kurzem haben FPGAs auch Einzug in Clouds gefunden. So nutzen beispielsweise Amazon, Microsoft und IBM FPGA-Technologie nicht nur für interne Funktionen und eigene Anwendungen in ihren Clouds, sondern bieten ihren Kunden Zugang zu speziellen Cloudinstanzen mit FPGAs an. Die beiden marktführenden FPGA-Hersteller Intel (durch Übernahme der Firma Altera im Jahr 2015) und Xilinx fokussieren aktuell stark auf Rechenzentren und versuchen FPGAs als Alternative zu GPUs zu vermarkten.

Für den erfolgreichen Einsatz von FPGAs im Rechenzentrum muss in den nächsten Jahren Fortschritt in mehreren Bereichen erzielt werden. Im Zentrum steht dabei die Akzeptanz durch die Anwender. Dabei muss geklärt werden, für welche Rechenzentrumsanwendungen FPGAs besonders geeignet sind und wie die benötigten Codes erstellt werden können. Während für die Programmierung C-Dialekte wie OpenCL oder auch das javabasierte Datenflussprogrammiermodell von Maxeler [8] etabliert sind, sind domänenspezifische Ansätze Gegenstand aktueller Untersuchungen. Beispiele dafür sind die Domänen der Stencil-Berechnungen im wissenschaftlichen Rechnen oder der tiefen neuronalen Netzwerke [3].

Aus Sicht der Rechenzentrumsbetreiber und der Technologiehersteller sind besonders Fragen bezüglich der Integration von FPGAs in ein Rechenzentrum relevant. Dazu gehört der gewünschte Mehrmandantenbetrieb von FPGAs, der die Entwicklung geeigneter Virtualisierungsansätze erfordert und zu Sicherheitsfragen führt. Auch die Skalierbarkeit auf eine größere Anzahl von FPGAs ist ein spannendes Thema. Die Verbindung von FPGAs über die CPU-basierten Verbindungsnetzwerke aktueller Cluster ist meist sehr ineffizient. Hier stellen direkte FPGA-FPGA-Netzwerke mit optischen Switches einen interessanten Ansatz dar.

Neben diesen technologisch-wissenschaftlichen Fragestellungen müssen Rechenzentrumsbetreiber aber auch neue Herausforderungen in den betrieblichen Abläufen meistern. Diese reichen von der Technologie- bzw. Herstellerauswahl über die Adaption der Betriebssoftware bis hin zur Wartung. Dabei ist es unerlässlich, fachlich qualifiziertes Personal zu entwickeln bzw. zu rekrutieren.

Diese Fragestellungen werden im Paderborn Center for Parallel Computing (PC^2) [6], einem wissenschaftlichen Institut der Universität Paderborn mit einem Schwerpunkt auf energieeffizientes Hochleistungsrechnen mit FPGAs, bearbeitet. Sowohl durch Forschungsbeiträge, z. B. im Rahmen des Sonderforschungsbereichs 901 On-The-Fly Computing [7], als auch im produktiven Rechnerbetrieb werden Lösungen für die skizzierten Herausforderungen entwickelt und erprobt.

Literatur

1. Bertin P, Roncin, D Vuillemin J (1993) Programmable Active Memories: A Performance Assessment, Technical Report. Digital Equipment Corporation, March

2. Borkar S, Chien AA (2011) The future of microprocessors. Commun ACM 54(5):67–77
3. Chung E et al (2018) Serving DNNs in real time at datacenter scale with project brainwave. IEEE Micro 38(2):8–20
4. Estrin G (2000) Reconfigurable computer origins: The UCLA Fixed-Plus-Variable (F+V) structure computer. IEEE Ann Hist Comput Oct/Dec:3–9
5. Freeman RH (1989) Configurable Electrical Circuit Having Configurable Logic Elements and Configurable Interconnects. U.S. Patent No. 4,870,302. Sep. 26
6. https://pc2.uni-paderborn.de, viewed 21 May 2019
7. https://sfb901.uni-paderborn.de, viewed 21 May 2019
8. Pell O, Averbukh V (2012) Maximum performance computing with dataflow engines. Comput Sci Eng 14(4):98–103