Konfigurationstechniken

Proseminar FPGAs
SS 2003
Christian Egger

1. Wieso Konfiguration?

Programmierung eines FPGA-Bausteins ist ein unzutreffender Begriff, denn die Daten, die dem Chip seine Funktion geben, sind selbst nicht ausführbar. Deswegen spricht man bei FPGAs anstatt von einem Programm geeigneter von einer Konfiguration.
FPGAs lassen sich nach ihrer Konfigurierbarkeit in Klassen einordnen (s. Abb. 1-1).

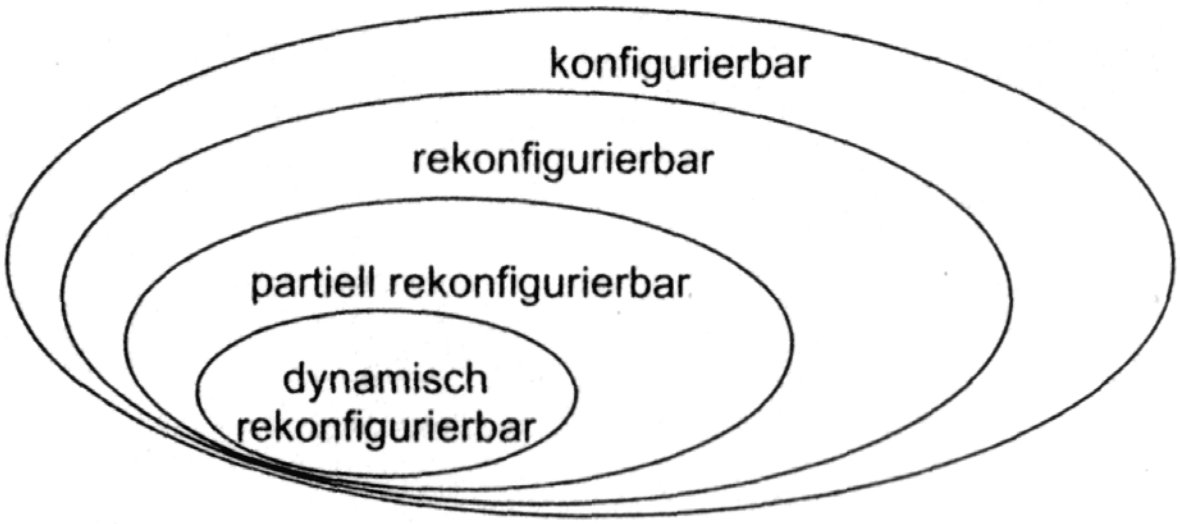

Abb. 1-1: Klassen der Konfigurierbarkeit

Die äußerste Klasse bilden die Antifuse-FPGAs; sie sind lediglich einmal konfigurierbar und werden im Weiteren nicht behandelt. Rekonfigurierbare FPGAs sind Look-Up-Table-basierte Bausteine. Die LUTs bestehen heute meist aus SRAM oder früher bevorzugt auch aus anderem wiederbeschreibbaren Speicher, wie z.B. in EEPROM Technologie. Partiell rekonfigurierbar heißt, dass einzelne Speichertabellen separat beschreibbar sind. Dynamisch rekonfigurierbar bedeutet während des Betriebs partiell rekonfigurierbar.

2. Techniken

2.1. Konfigurationsspeicher

Ein SRAM-basierter FPGA verliert jedes Mal seine Konfiguration, wenn die Spannungsversorgung kurz ausfällt oder einbricht. Die dann zur Rekonfiguration benötigten Daten befinden sich in einem externen Speicher. Früher waren PROMs wie z.B. EPROMs verfügbar, später auch EEPROMs und heute vor allem Speicher in Flash Technologie. Diese Speicher sind entweder seriell oder parallel an den FPGA angeschlossen und übertragen ihre Daten bitweise (seriell) oder byteweise (parallel). Es bieten sich auch andere Möglichkeiten, den Chip zu konfigurieren.
Ein im System vorhandener Prozessor kann diese Aufgabe ebenfalls übernehmen. Dabei könnten die Daten von Festplatte, Hauptspeicher oder aus dem Netz stammen. Sehr flexibel sind auch Konfigurationsspeicher mit integrierter Logik.

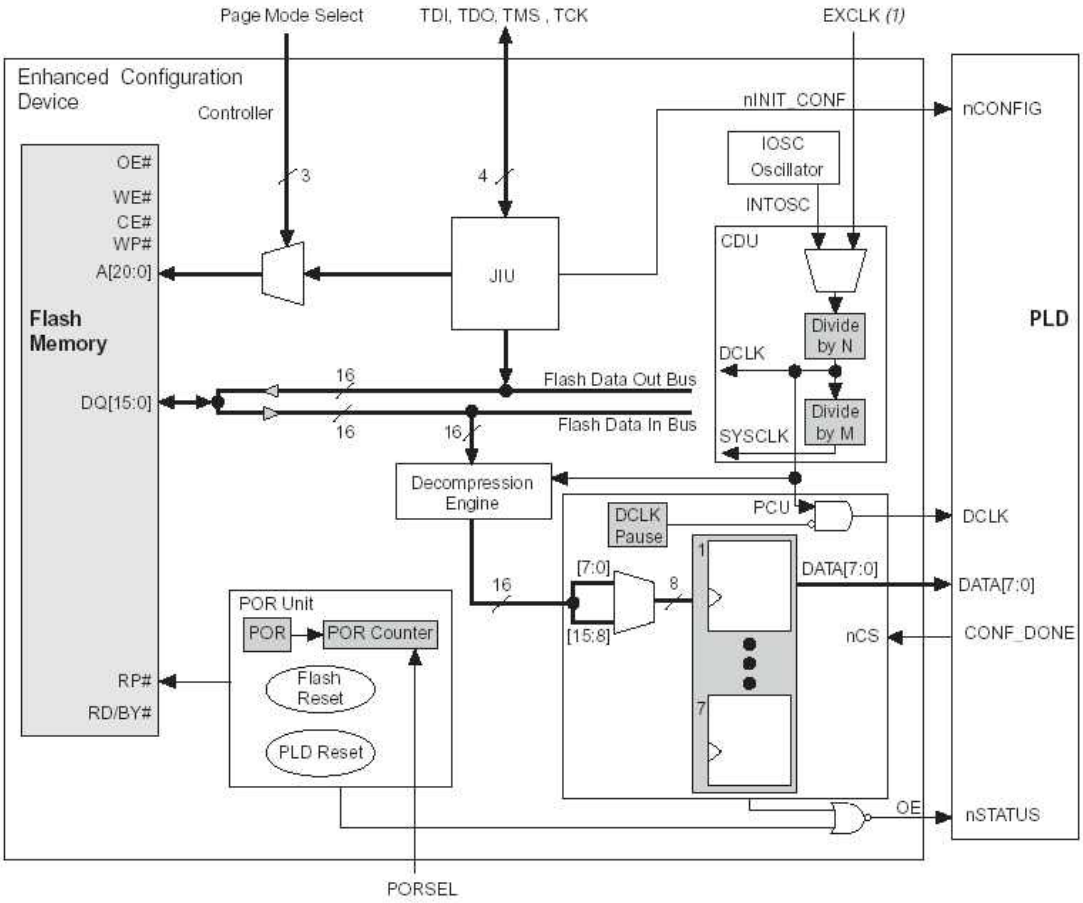

Abb. 2.1-1: Aufbau eines Konfigurationsspeichers aus der EPCxx Serie von Altera

Die Bauteilserie EPCxx (s.Abb. 2.1-1) für Altera's FPGAs Apex II, Apex 20kl, Mercury u.a. bietet beispielsweise folgende Funktionen:

→ 4, 8 oder 16 Mbit Flashspeicher
Der Speicher kann mehrere verschiedene Konfigurationen speichern und kann auch dem FPGA teilweise zugänglich gemacht werden.

→ Power-On reset circuitry (POR)
Diese Einheit erzeugt einen Reset nach dem Einschalten, bis die Spannungsversorgung stabil ist.

→ Interner Oszillator (IOSC)
Der Oszillator kann Frequenzen von 8 bis 53MHz generieren mit denen der FPGA beim Konfigurieren getaktet wird. Die reale Frequenz schwankt dabei abhängig von Temperatur und anderen Faktoren um bis zu 20% nach oben und nach unten. Dies stört allerdings nicht, da es beim Konfigurieren nicht auf eine genaue Einhaltung der Frequenz ankommt. Braucht man jedoch für andere Zwecke eine genauere Frequenz, kann man diese von Außen einspeisen.

→ Konfigurationseinheit (PCU – PLD Configuration Unit)
Diese Einheit ist für die Konfiguration des FPGA zuständig.

→ Dekompressionseinheit
Diese Funktion wird gebraucht, wenn die Konfigurationsdaten vor dem Schreiben per Software komprimiert wurden. Dadurch kann die Speicherkapazität von 4, 8 oder 16Mbit auf bis zu 7, 15 oder 30Mbit erhöht werden. Dies bietet sich vor allem an, wenn mehrere verschiedene Konfigurationen für einen oder mehrere FPGAs gespeichert werden sollen.

→ Frequenzteiler (CDU – Clock Division Unit)

→ JTAG Interface Unit (JTIU)
JTAG ist ein IEEE Standard, der den Zugriff auf den Flashspeicher und einige Befehle, wie z.B. Reset definiert. Wenn in den ISP-Modus (ISP = In System Programming) gewechselt wird, terminiert die JTIU zuerst jeden Zugriff auf den Flashspeicher (z.B. durch eine gerade laufende Rekonfiguration). Danach kann man den Speicher überprüfen, löschen, neu beschreiben oder eine Rekonfiguration des FPGA einleiten. Die JTAG Schnittstelle arbeitet mit dem für den Flash höchstmöglichen Takt von 10 MHz.

2.2. Konfigurationsmodi

Zur Zeit werden vor allem programmierbare flashbasierte Konfigurationsspeicher beworben, die mehre Möglichkeiten der Rekonfiguration bieten. Xilinx bietet Konfigurationschips an, die je nach Komplexität folgende Konfigurationsmodi (s.Abb.2.2-1) bieten. Wenn die Daten seriell übertragen werden, wird nur ein Pin des FPGA benutzt. Die parallele Übertragung ist achtmal schneller, weil pro Takt ein Byte übertragen wird, wie es bei der aktuellen Spartan und Virtex Familie von Xilinx oder dem Apex II von Altera der Fall ist. Xilinx nennt dieses Feature SelectMap, bei Altera heißt es Fast Passive Parallel. Früher war der parallele Modus genauso schnell wie der serielle, weil jedes Byte erst durch ein Schieberegister serialisiert wurde.

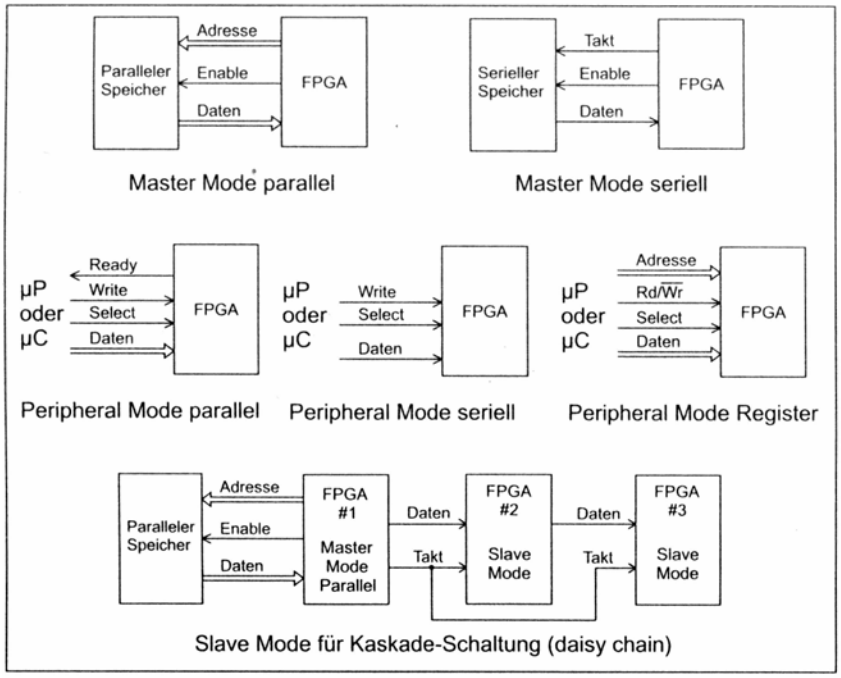

Abb. 2.2-1: Konfigurationsmodi

→ Master Mode seriell / parallel
Bei diesen Modi ist der FPGA selbst für Adressierung und Ansteuerung des angeschlossenen Konfigurationsspeichers zuständig. Als Speicherbausteine können hierbei simple (Flash-/EE-/E-)PROMs eingesetzt werden.

→ Peripheral Mode seriell / parallel
Die Daten werden in diesem Modus von der Peripherie, wie zum Beispiel einem anderen FPGA oder auch einem Prozessor, der ebenfalls im System sitzt, gesteuert übertragen. Dies ermöglicht sehr flexible Konfigurationsmöglichkeiten, denn der Prozessor kann seine Daten von vielen Quellen haben, z.B. einer Festplatte oder aus dem Netz. Altera vertreibt auch ein serielles Linkkabel

→ Slave Mode (Passive Mode) seriell /parallel
Dieser Modus ist heute der Gebräuchlichste. Bei ISP werden Speicherbausteine mit integrierter Logik verwendet, um den FPGA neu zu konfigurieren. Falls mehrere Bausteine identisch konfiguriert werden sollen, bietet sich eine Kaskadierung (Daisy-Chaining) an. Im Falle von unterschiedlichen Konfigurationen können bis zu acht FPGAs gleichzeitig von einem Speicher angesprochen werden, indem jeder FPGA mit einem der acht Bitleitungen des parallelen Ausgangs verbunden wird.

2.3. ISP (In-System Programming)

Die Rekonfigurierbarkeit von Lookuptable-basierten FPGAs wie SRAM oder auch Flash-FPGAs hat wesentliche Vorteile gegenüber ASICs oder Antifuse-FPGAs. Die Wiederbeschreibbarkeit von Flashzellen ist zwar im Gegensatz zu SRAM-Zellen auf bis zu 10.000 Schreibzyklen begrenzt, reicht aber aus. Rekonfigurierbare FPGAs lassen sich mittlerweile auch im System programmieren. Dies bringt gleich mehrere Vorteile:

→ ein zusätzliches Programmiergerät entfällt.
→ Schnelles Testen der fertig aufgebauten Schaltung mit mehreren verschiedenen Konfigurationen zur Entwicklung
→ ISP Bausteine können zum Entwickeln so konfiguriert werden, dass die aktuelle Konfiguration inklusive der im Chip abgelegten Anwendungsdaten ausgelesen werden können, was ein Debuggen eines fehlerhaften Designs erleichtert.
→ Es kann eine spezielle Testkonfiguration geladen werden, bei der der FPGA die anderen Bauteile und Verbindungen testet, indem er verschiedene Testsignale erzeugt und das Ergebnis mit einem Sollergebnis verglichen wird. Dieser Test benutzt die JTAG-Schnittstelle und wird als JTAG Boundary Scan bezeichnet.

Nachteile ergeben sich aber auch, z.B. wenn sich der Entwickler wegen der leichten Rekonfigurierbarkeit zu einem schlechten Entwurfsstil (Trial and Error) verleiten lässt.

3.Schutzmechanismen

In den letzten Jahren haben sich SRAM-basierte FPGAs immer mehr auf dem Markt verbreitet, der vorher von ASICs beherrscht wurde. Zum Beispiel bewirbt Xilinx seine Spartanfamilie für den Einsatz in Verbrauchergeräten. Am oberen Ende sind FPGAs mit mehreren Millionen Gattern erhältlich. Die Kosten für solch große Entwürfe sind enorm. Diese Entwicklung erfordert einen angemessen Schutz gegenüber dem geistigen Eigentum, der Konfiguration. Ohne Schutz kann man die Konfiguration sehr einfach bekommen, indem man den Bitstream zum FPGA bei einem Update oder nach einem Reset abhört. Es gibt zwei verschiedene Arten von Angriffen:
Die erste besteht daraus, das Gerät simpel zu kopieren, was auch bei der Hardwarepiraterie geschieht. Viel schlimmer ist es aber, wenn die Konfiguration einem Reverse-Engineering unterzogen wird, was fast nicht nachgewiesen werden kann. In der Vergangenheit wurden viele Verfahren benutzt, um dies zu verhindern. Viele davon haben aber die Funktionsfähigkeit des Chips eingeschränkt oder waren angreifbar.

3.1. Security through Obscurity (Sicherheit durch Geheimhaltung)

Der erste Ansatz, der sich allerdings nur gegen Reverse-Engineering aber nicht gegen simples Kopieren richtete, war die Geheimhaltung des Bitstream-Formats. Dies wurde von Xilinx allerdings aufgegeben, als sie ihr Jbits SDK veröffentlichten, das für die dynamische Rekonfiguration von Standardbauteilen verwendet wird und auch eine API für den Bitstream mitbringt.

3.2. Kapselung

Atmel brachte 2000 eine Version eines FPGAs auf den Markt, der den seriellen Flashspeicher schon mit auf dem Chip integrierte. Diese Methode hat den Nachteil, dass die Rekonfiguration relativ restriktiv ist. Dies verbietet dem Designer z.B. einen gemeinsamen Konfigurationsspeicher für mehrere FPGAs einzusetzen.

3.3. CMOS-Batterie

Ein Verfahren, dass theoretisch zwar einen hohen Schutz gewährleistet, ist der Einsatz von einer Batterie, die den FPGA während das Gerät ausgeschaltet ist, mit Strom versorgt. Dies verhindert jedoch den Einsatz vieler Bauteile, weil deren Ruhestrom zu hoch ist. Auch bei Stromspar-FPGAs liegt der Ruhestrom noch bei ca. 10µA was die Lebensdauer eines solchen Geräts auf ca. 5-10 Jahre verkürzt, je nach Einschaltverhalten des Benutzers.

3.4. Benutzerdefinierter Schlüssel

Dieses Verfahren berhuht darauf, dass der FPGA ein Register (EEPROM / Flash) besitzt, in das ein selbstgewählter Schlüssel geschrieben wird. Dieser Wert bleibt nach einem Stromausfall erhalten und der FPGA kann nach einem Reset die verschlüsselte Konfiguration aus einem externen Speicher entschlüsseln und laden. Der Nachteil dieses Ansatzes ist, dass sich nichtflüchtiger Speicher auf dem Chip befinden muss, was standardmäßig bei der Produktion nicht vorgesehen ist und den FPGA verteuert. Für ausreichenden Schutz benötigt zudem jeder Baustein ein anderes Konfigurations-/Schlüsselpaar, was die Produktion aufwendiger macht.
Der Xilinx benutzt eine abgewandelte Methode beim Virtex II, der keine nichtflüchtigen Speicherzellen besitzt. Der Schlüssel wird in ein SRAM Register geladen, das am Gehäuse eigene Pins zur Stromversorgung hat. An diese wird wiederum eine Batterie angeschlossen. Der Vorteil gegenüber dem vorherigen Verfahren (s. 3.3. CMOS-Batterie) ist, dass die Anwendung nicht auf Stromspar-FPGAs beschränkt ist, da nur ein einzelnes Register mit Spannung versorgt werden muss.

3.5. Encrypted Configuration Bitstreams

Bei diesem Verfahren ist man neuerdings auf die Anforderungen des Marktes eingegangen. Häufig werden Flash-EEPROMs zur Rekonfiguration eingesetzt. Zudem werden die meisten FPGA Platinen beim Gerätehersteller konfiguriert. Dadurch ergeben sich keine Sicherheitsbedenken beim ursprünglichen Beschreiben des Konfigurationsspeichers. Ein solcher FPGA besitzt einen geheimen Schlüssel und ein Verschlüsselungswerk, das ein symmetrisches Verfahren, Triple-DES benutzt. Mit dem geheimen Schlüssel verschlüsselt der FPGA die Konfiguration, die er initial via JTAG Schnittstelle im Werk erhält und brennt diese in den Flashspeicher (s. Abb. 3.5-1). Nach einem Reset erhält er diese wieder und entschlüsselt die Konfiguration wieder (s. Abb. 3.5-2).

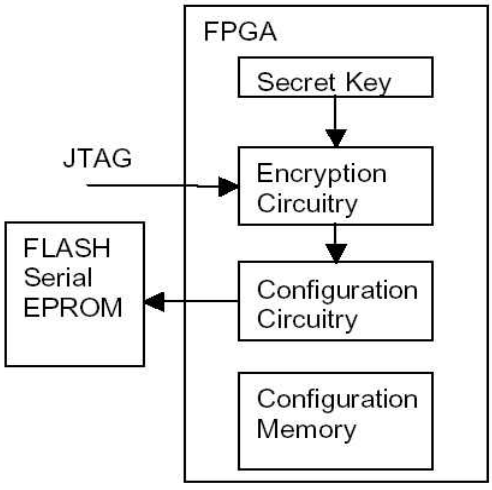

Abb. 3.5-1: Initiale Konfiguration im Werk

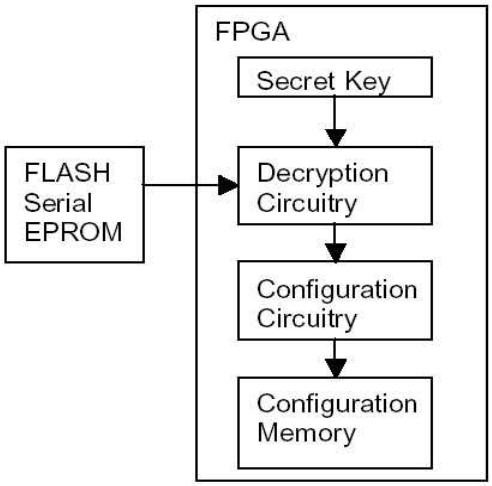

Abb. 3.5-2: Rekonfiguration

Je nach Herstellungsverfahren ist dieser geheime Schlüssel entweder ganz zufällig, oder stammt aus einem begrenztem Schlüsselraum von z.B. 10 Stück. Letzteres verbilligt den Herstellungsprozess, ist aber trotztdem sicher, da ein Angreifer nicht weiß, welcher Schlüssel verwendet worden ist. Es verhindert zwar eine Raubkopie nicht, aber um nun erfolgreich einen anderen FPGA damit zu konfigurieren, müssen im Schnitt 10 Chips gekauft werden, was unwirtschaftlich ist. Wenn die Konfiguration nicht zu seinem Schlüssel passt, ist es außerdem möglich, dass der Chip zuviel Strom verbraucht und dabei Schaden nimmt. Zusätzlich kann man den verwendeten Schlüsselraum zeitlich oder geografisch (ähnlich zu Regionalcodes von DVDs) beschränken.

4. Zusammenfassung

Auf dem Sektor der rekonfigurierbaren FPGAs haben sich seit dem Trend weg von Antifuse-FPGAs zur SRAM-Technologie einige wesentliche Verbesserungen ergeben. So stieg die Flexibilität noch mit der Möglichkeit, intelligenteren Konfigurationsspeicher einzusetzen. Außerdem hat die Branche bemerkt, wie wichtig es ist, für die Sicherheit der Chipkonfiguration zu sorgen und hat sich weg von der Verschwiegenheit über das Bitstream-Protokoll hin zu transparenteren und sichereren Schutzmöglichkeiten bewegt. Dies hat die Position der SRAM Technologie noch mehr gestärkt und hat den Markt für neue Anwendungen geöffnet.

5. Literatur

Wannenmacher, Markus: Das FPGA-Kochbuch, mitp-Verlag
www.xilinx.com: Datenblätter, Verschlüsselung
www.altera.com: Datenblätter, JTAG
www.atmel.com: allgemeine Information
www.algotronix.com: ISP, Verschlüsselung
www.dinigroup.com: Verschlüsselung