An den Vizepräsident für Forschung, Entwicklung und Technologietransfer der Fachhochschule Braunschweig/Wolfenbüttel

Forschungsbericht (Kurzfassung) **SS 2003**

BERMBACH, Rainer, Prof. Dr.-Ing.

Name, Vorname, Titel, ggf. wiss. Einheit

Elektrotechnik

Fachbereich

03.09.2003

Unterschrift, Datum

Entwicklung eines Mikroprozessorkerns

Thema des Vorhabens

Das Ziel des Vorhabens, einen kleinen Mikroprozessorkern zu entwickeln, konnte erreicht werden. Es entstand ein adaptierbarer, auf preiswerten FPGA-Modulen implementierbarer Kern, der mit den Mikrocontrollern der sog. PICmicro Mid-Range Family der Firma Microchip weitestgehend kompatibel ist. Die Realisierbarkeit in programmierbarer Hardware ist ein wesentlicher Unterschied zu anderen frei verfügbaren Mikroprozessorkernen, deren Beschreibungen nicht oder höchstens teilweise implementierbar sind.

Die Auswahl der Architektur fiel auf die PICmicro-Controller, da sie bei mittlerer Leistungsfähigkeit vergleichsweise einfach strukturiert ist und da für sie eine leistungsfähige, kostenlose Entwicklungsumgebung (IDE) verfügbar war. Mit Hilfe dieser IDE kann eine Softwareentwicklung für den Mikrocontroller vorgenommen werden. Den erzeugten Code wandelt ein im Rahmen des Projektes entstandenes Programm in eine VHDL-Beschreibung um. Zusammen mit der Beschreibung des Kerns in VHDL (VHSIC Hardware Description Language = Hardwarebeschreibungssprache) kann der Mikrocontroller implementiert und in ein FPGA (Field Programmable Gate Array = programmierbare Hardware) geladen werden. Danach steht er zum Betrieb zur Verfügung. Auf diese Weise gelangt man in weniger als fünf Minuten inkl. Übersetzen der Software zu einem einsetzbaren Mikrocontroller. Dies entspricht in etwa auch der benötigten Zeit bei der Programmierung realer PIC-Bausteine.

Der entworfene Controller verfügt über alle Einheiten des realen PICmicro-Kerns inkl. Interruptverarbeitung sowie Befehlsspeicher und RAM. An Peripheriekomponenten stehen derzeit die Funktionalität des Timer 0, des Watchdog sowie die 8-Bit-I/O-Ports A bis G zur Verfügung. Die max. Codegröße des Programms kann 2,5KWorte à 14 Bit betragen. Dabei werden die sog. Block RAMs des ausgewählten FPGAs verwendet. Diese sind hierfür 16 Bit breit organisiert, so daß die für den Code nicht benötigten beiden Bits jedes Speicherwortes für andere Zwecke zur Verfügung stehen. So kann man mit wenig Zusatzhardware z.B. einen Single Step Mode oder Breakpoints realisieren. Auch für das RAM und den Stack findet je ein Block RAM Verwendung.

Über ein Konfigurations- und Definitionsmodul (in VHDL) kann der entwickelte Prozessorkern an unterschiedliche reale Typen angepaßt werden. Im wesentlichen betrifft dies die Ausstattung mit RAM sowie die dazugehörige Adreßlage. Abweichungen zur Architektur realer PIC-Bausteine ergeben sich nur dort, wo die entsprechende Funktionalität nicht oder nicht sinnvoll im FPGA implementiert werden kann. So ist z.B. kein unabhängiger RC-Oszillator für den Watchdog vorhanden, sondern sein Takt wird aus dem Prozessortakt abgeleitet. Der sog. Sleep Mode (Power Down) bringt nur die interne Verarbeitung zum Stillstand, eine Stromersparnis ist nicht realisierbar.

Der Test des Entwurfs erfolgte per funktionaler Simulation zuerst der Einzelmodule, dann des Gesamtsystems durch Simulation der Ausführung eines umfangreichen PICmicro-Testprogramms. Anschließend, nach Plazierung und Routing, fand eine Timing-Simulation (Post Place & Route) mit real zu erwartenden Verzögerungsparametern statt. Zur besseren Kontrolle der implementierten Hardware wurden wichtige Signale wie Clock, Program Counter, Address und Data Bus auf Pins des FPGAs herausgeführt, dort mit dem Logikanalysator kontrolliert und mit der Simulation verglichen. Außerdem erfolgte der Vergleich mit den (Außen-) Reaktionen eines realen PIC-Controllers, der mit dem gleichen Testprogramm lief. Zur leichteren Überprüfung der Funktionsfähigkeit auf Programmniveau wurde das verwendete FPGA-Modul an ein I/O-Modul angeschlossen, das die Ausgabe über ein LCD, ein LED-Display, Einzel-LEDs, serielle Schnittstelle sowie die Eingabe über Tasten und Schalter ermöglicht.

Gegenwärtig sind zwei Fehler bekannt. Sie erzeugen eine ungewollte Signalflanke bei Änderung der Flankenwahl des Eingangssignals von Timer 0 und externem Interrupt (läßt sich per SW abfangen). Die gesamte Befehlsausführung des Kerns scheint bislang korrekt zu laufen. Der Test zeigte auch Probleme des verwendeten Syntheseprogramms (Xilinx XST, ISE 5.1). Es „optimierte" benötigte Schaltungsteile heraus, wenn Sie nur Verbindungen zum internen Tristate-Datenbus besaßen. Durch das Herausführen von Signalen auf die FPGA-Pins zu Testzwecken konnte dieser Tool-Fehler umgangen werden. Der Hersteller versprach Abhilfe mit der nächsten größeren Software Release.

Der Controller erreicht derzeit eine maximale Taktfrequenz von > 25 MHz. (Die Mehrzahl der realen PIC-Controller läuft mit 20 MHz.) Diese Geschwindigkeit läßt sich voraussichtlich noch steigern, da momentan die Block RAMs auf einer anderen Taktflanke arbeiten als die restliche Schaltung. Zur Umstellung ist eine nicht unerhebliche Designänderung nötig. Erste Versuche zeigten mögliche max. Taktfrequenzen im Bereich > 36 MHz. Das Design benötigt 380 (16%) von 2352 verfügbaren Slices (= 1176 CLBs, Configurable Logic Blocks) im verwendeten Xilinx Spartan II FPGA (XC2S200-5). Erwartungsgemäß „verbrauchen" ALU bzw. Befehlsdekoder die meisten Ressourcen, gefolgt vom Program Counter (inkl. Stack).

Mit Erreichen des Hauptziels des Vorhabens steht ein frei verfügbarer Mikrocontrollerkern zur Verfügung. Für Erweiterungen z.B. mit weiteren Peripheriekomponenten ist eine Standardschnittstelle im Design vorhanden. Für die Einbindung des Controllers in übergeordnete Systeme (SoC, System on Chip), besitzt der verwendete FPGA noch ausreichend „Platz". Auch eine Umsetzung auf eine andere Hardwareplattform (größere FPGAs) ist vermutlich mit geringem Aufwand möglich, da Baustein-spezifische Teile im Design gut isoliert bzw. weitgehend vermieden wurden.

Im Hinblick auf einen optimalen Einsatz des Kerns sind noch weitere Untersuchungen, Entwicklungen und Tests nötig. So sollte die Taktflanke bei den Block RAMs umgestellt werden. Durch Zwischensynchronisierungen (Pipelining) und Überarbeitung wesentlicher Komponenten (ALU) sollte die erreichbare Taktfrequenz weiter zu steigern sein. Natürlich müssen die beiden Fehler bei der Taktflankenwahl noch beseitigt werden. Die nächste Software Release sollte einen Betrieb ohne externe Testsignale ermöglichen. Die Überprüfung der Einbindung weiterer Komponenten bzw. der Integration in übergeordnete Hardware steht noch aus. Für einen praktikableren Einsatz der Ein- und Ausgabe auf dem I/O-Modul ist die Entwicklung einer Software mit standardisierten Softwareschnittstellen nötig.

Ergebnisbericht, insbesondere über neue Erkenntnisse

gesehen und weitergeleitet: Vizepräsident für Forschung, Entwicklung und Technologietransfer