Reed Electronics Group

MICROPROCESSOR REPORT

www.MPRonline.com

THE INSIDER'S GUIDE TO MICROPROCESSOR HARDWARE

TEJA'S FPGA PLAY

New Tools Build Packet Processors Using ANSI C and FPGAs

By Tom R. Halfhill {4/3/06-02}

If off-the-shelf network processors don't fit the bill, but designing a custom part is too costly or intimidating, Teja Technologies has a fresh alternative: Teja FP (FPGA Platform). It's a package of development tools, software, and hardware intellectual property (IP) that allows software engineers to build a packet processor in an FPGA without using a hardware description language (HDL) or fabricating custom silicon.

With Teja FP, programmers can start with existing data-plane code written in ANSI C or write new code in that language. After profiling and analyzing the code, the next step is to partition the application. The most compute-intensive parts can execute in the FPGA's programmable-logic fabric, while other parts can run on soft processor cores synthesized in the fabric. Although, at present, the profiling, analysis, and partitioning are largely manual tasks, Teja FP can automatically compile ANSI C into logic for the hardware partition or into assembly language for the software partition. In addition, Teja offers some prewritten hardware and software IP for accelerating common packet-processing tasks.

Teja FP is primarily intended for optimizing data-plane code at wire speeds in the 1Gb/s to 2Gb/s range. Control-plane code can run on hard processor cores embedded in the FPGA or on an external processor. For now, Teja FP works only with Xilinx Virtex-4 FX devices, so the lone integrated-processor option is the PowerPC 405. Customers preferring an external processor can compile their control code for virtually any CPU architecture.

Because Teja FP targets FPGAs for both development and deployment, it offers several advantages over conventional solutions. Programmers can bend the hardware to match the software, instead of writing software that must conform to the fixed architecture of a standard-part processor. Development is a feedback-driven iterative process, allowing programmers to fine-tune the application's hardware and software partitions for optimal performance. Deployment is rapid and less risky, because it eliminates the need to spin custom silicon. Upgrades are easier, even in the field, because the software partition is fully programmable, and the hardware partition is implemented in reprogrammable logic. Result: a highly customizable, flexible packet processor.

Of course, high-end FPGAs like the Virtex-4 FX aren't cheap. Even in volume, unit costs can reach hundreds of dollars. Project managers must balance those costs against the nonrecurring engineering expenses of developing and manufacturing a custom chip, or of buying and programming standard parts that aren't as flexible. In some cases, depending on the size of the software, a packet processor implemented in an FPGA can compete with the unit costs of standard-part network processors and high-performance general-purpose processors. Teja expects most customers to turn to Teja FP because they must extend standard networking protocols or create proprietary implementations of standard protocols.

Microprocessors: The New Macro Blocks

One of the most interesting aspects of Teja FP is that it builds a packet-processing pipeline by stringing together 32-bit soft processor cores. These cores—Teja calls them engines—are the MicroBlaze processors licensed by Xilinx exclusively for synthesis in Xilinx FPGAs. MicroBlaze is a general-purpose embedded-processor core based on Xilinx's own 32-bit RISC

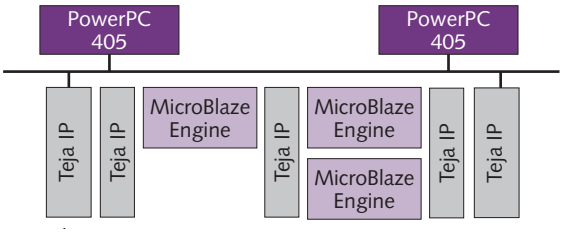

Figure 1. A low-cost packet processor designed with Teja FP might use only two or three MicroBlaze soft processor cores. Each MicroBlaze core is an engine in the packet pipeline. In this diagram, packets flow through the pipeline from left to right. Individual stages can have one or more MicroBlaze engines, depending on the software's complexity and the performance target (e.g., the wire speed). The Teja FP development platform includes special IP for linking the engines into a functional packet pipeline with dispatch logic and queues.

architecture. The latest version has an optional FPU, an optional 32-bit integer multiplier, new pattern-compare instructions, improved debug logic, and faster clock speeds when synthesized for high-end Xilinx FPGAs. (See *MPR 5/17/05-02*, "MicroBlaze Can Float.")

Although MicroBlaze isn't specifically designed for packet processing, Teja FP uses it as a basic macro block. A simple FPGA-based packet processor designed for low cost might arrange two or three MicroBlaze engines in a serial pipeline. A complex FPGA-based packet processor designed for high throughput might arrange a half-dozen or more MicroBlaze engines in a pipeline configuration having both serial and parallel dimensions. In effect, each MicroBlaze engine becomes a stage—or part of a stage—in the packet-processing pipeline.

The MicroBlaze engines connect to each other's Fast Simplex Links (FSL), not their on-chip peripheral buses. FSLs are unidirectional point-to-point data-streaming interfaces that are better suited for this application than conventional shared buses are. Although the resulting packet processor is technically a multicore design built with symmetric processor cores, it's not a symmetric multiprocessor (SMP) chip.

Figure 1 shows an example of a low-cost packet processor designed with Teja FP. This illustration is both a block diagram and a pipeline diagram. As a block diagram, it shows a high-level view of the packet processor's major components inside the FPGA, including three MicroBlaze soft cores in the data plane and two PowerPC 405 hard cores in the control plane. As a pipeline diagram, the figure depicts a two-stage packet pipeline in which each MicroBlaze engine is part of a pipe stage. Packets enter the pipeline at the left and exit at the right. Teja FP includes the hardware IP required to connect the engines together, to implement logic for dispatching the packets, and to build queues that temporarily hold packets between stages.

A crucial step in the development project is determining the number of MicroBlaze engines the packet processor will require, as well as the best configuration of engines within the pipeline. These decisions depend on the complexity of the packet-processing software and the minimum wire speed the processor must maintain. If the developer underestimates the processing power needed, the processor will either drop packets at the target wire speed or be restricted to slower wire speeds. If the developer overestimates the processing power needed, the finished design will cost more, because it requires a larger FPGA to accommodate more MicroBlaze engines and their associated logic.

Stringing Processors Into Pipelines

Consider a few examples of designing a packet processor with Teja FP. In the simplest case, assume that a MicroBlaze engine in the programmable-logic fabric runs at a clock frequency sufficient to perform 50 clock cycles at the target wire speed. If the packet-processing software requires only 50 or fewer cycles to execute, then a single MicroBlaze engine with a 50-cycle budget would be enough to sustain the data rate.

But chances are that 50 clock cycles aren't enough to get the work done. If processing a packet requires 100 cycles, two MicroBlaze engines arranged serially could pipeline the task. The first engine could do half the work, then forward the partially processed packet to the second engine, which would finish the job. Meanwhile, the first engine begins processing the next packet. That's classic pipelining, just like the instruction pipelines inside a microprocessor.

Pipelines can grow wider as well as longer. By arranging two or more MicroBlaze cores side by side, so to speak, the pipeline can use multiple engines to simultaneously perform a task that exceeds the clock-cycle budget of one engine. As Figure 2 shows, a pipeline can be both serial and parallel. Another technique, also illustrated in Figure 2, is to create a "bulge" in the pipeline by adding extra engines to some stages. This is particularly useful if the software doesn't break down into evenly divided tasks.

As a reference point, Teja says an IPv4 packet processor capable of sustaining a line rate of 1Gb/s would require four MicroBlaze engines arranged in a 2×2 matrix. In other words, the packet pipeline would be two

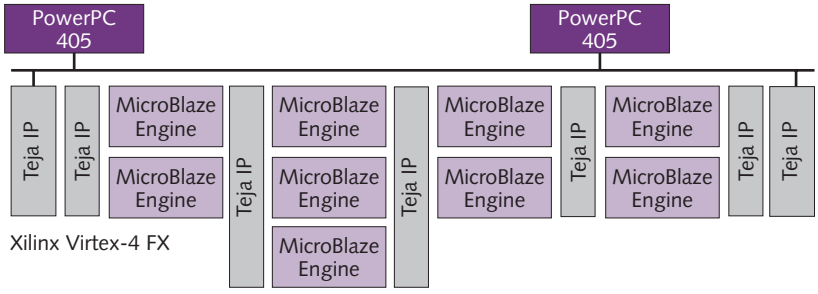

Figure 2. A high-performance packet processor designed with Teja FP requires several MicroBlaze processor cores. This example has nine cores, including one extra-wide stage that provides additional processing horsepower.

engines wide and two stages deep. Naturally, the scale of any particular design depends on the amount of processing needed. Presumably, a packet processor created with Teja FP is doing something specialized; otherwise, a standard part would be more suitable.

For another reference point, consider that the base configuration of a MicroBlaze core requires at least 950 logic cells (programmable-logic lookup tables, or LUTs). It grows significantly with the addition of optional features. The largest option is an FPU, which doubles the size of the core, but packet processors rarely need floating-point math. (One exception might be a specialized packet-filtering function.) The smallest Virtex-4 FX device is the XC-4VFX12, which has 12,312 logic cells, and the largest is the XC-4VFX140, which has 142,128 logic cells.

Therefore, even the smallest device that Teja is targeting with this development platform has enough capacity for perhaps a half-dozen MicroBlaze engines, after leaving room for the glue logic and some offload logic. The largest Virtex-4 FX device has enough logic cells for dozens of MicroBlaze engines. Of course, the prices of these huge FPGAs get pretty steep, so developers can't run wild with their designs. Table 1 shows six members of the Virtex-4 FX family and their features.

Translating ANSI C Into Logic Gates

Teja FP includes profiling tools that help identify hot spots in the application code, but it doesn't automate the process of designing a packet pipeline. Although Teja plans to automate some lower-level tasks in future versions of Teja FP, developers must do the high-level work themselves. The most important task is analyzing the software to determine the number of MicroBlaze engines needed and how best to arrange the engines.

Nor does Teja FP automate hardware/software partitioning. Some compute-intensive tasks may be unsuitable for software execution. (Even when synthesized for the fastest Xilinx FPGA, a MicroBlaze core reaches a top speed of only about 200MHz.) Some operations, such as memory accesses, may have long latencies that stall the packet pipeline. When software execution isn't practical, the alternative is to offload a task onto a logic block implemented directly in the FPGA's fabric. Teja provides some hardware-IP blocks for offloading common tasks, including blocks for calculating checksums, copying memory, and examining prefixes to find the packet's next destination address.

One thing Teja FP does automate, to a large degree, is designing custom logic for the programmable fabric. Normally, this requires significant labor with an HDL. With Teja FP, software programmers can rapidly create custom logic using ANSI C. Teja CC (C Compiler) can automatically convert ordinary C code into register-transfer-level (RTL) logic for the FPGA. Although Teja CC can't automatically partition an application into hardware and software components, it eases the onerous task of implementing the hardware and software components after developers have decided where the partition belongs.

Many other companies and researchers have struggled for years to transmute C into an HDL, and one current result is System C, which resembles a mash-up of C and Verilog. Teja is using a different alchemy. Teja CC can compile plain old ANSI C into logic, because the target is a bounded case and Teja's tools piggyback on the Xilinx development tools. Teja CC targets only the programmable logic of a Xilinx FPGA, and only for projects developed with Teja FP. It can't generate RTL for any other target. Likewise, for the software partition, Teja CC targets only one CPU architecture: Xilinx MicroBlaze. In the actual design flow, Teja CC creates a project for the Xilinx Embedded Development Kit (which Teja assumes the customer has already licensed). The output from Teja CC feeds into the Xilinx logic compiler, which generates the gate-level logic for the FPGA.

Within those limitations, Teja CC offers a fair degree of flexibility. Customers with existing hardware IP written

Feature	Virtex-4 FX XC-4VFX12	Virtex-4 FX XC-4VFX20	Virtex-4 FX XC-4VFX40	Virtex-4 FX XC-4VFX60	Virtex-4 FX XC-4VFX100	Virtex-4 FX XC-4VFX140
Logic Cells	12,312	19,224	41,904	56,880	94,896	142,128
PowerPC 405	1 core	1 core	2 cores	2 cores	2 cores	2 cores
Ethernet MACs (10/100/1,000Gb)	2	2	4	4	4	4
Block RAM/FIFOs +ECC (18Kb each)	36	68	144	232	376	552
Block RAM (Total)	648Kb	1,224Kb	2,592Kb	4,176Kb	6,768Kb	9,936Kb
Configuration Memory	5.01Mb	7.64Mb	15.8Mb	22.2Mb	35.1Mb	50.9Mb
Digital Clock Mgrs	4	4	8	12	12	20
Phase-Matched Clock Dividers	—	—	4	8	8	8
Differential I/O Pairs	160	160	224	288	384	448
XtremeDSP Slices	32	32	48	128	160	192
Rocket I/O Serial Transceivers	—	8	12	16	20	24
Max. Select I/O	320	320	448	576	768	896
Price (Volume)	$30 (50k)	$50 (25k)	n/a	n/a	n/a	n/a
EasyPath Available	—	Yes	Yes	Yes	Yes	Yes

Table 1. For now, the Teja FP development system works only with the Xilinx Virtex-4 FX family of programmable-logic devices. Even the smallest device in this family has enough logic cells to accommodate a few MicroBlaze cores in a pipelined packet-processor design. The highest-end parts are among the largest, most expensive FPGAs on the market, although the Xilinx EasyPath program can cut prices by as much as 80% for high-volume purchases. (n/a: pricing not available)

in VHDL or Verilog can use Teja CC to wrap their custom logic and incorporate it into a Teja FP project. Whether developers create their offload logic that way or write it directly in ANSI C, special hooks allow the software to invoke the hardware whenever necessary. As Figure 3 shows, Teja's development flow lets programmers repeatedly test their design and modify it to improve performance.

Teja FP is suitable for designing packet processors that run at wire speeds of 1–2Gb/s. That's not fast enough for the highest-end applications, but it's adequate for many other purposes, including anything with Gigabit Ethernet. Note that Virtex-4 FX devices integrate Gigabit Ethernet media-access controllers (MAC) with physical-layer (PHY) interfaces, eliminating the need for external MACs and PHYs. Therefore, it's possible to build a complete packet processor with control-plane processing, data-plane processing, and packet-I/O interfaces in a single FPGA.

Cheap Processor Cores Make It Possible

Xilinx wants to sell FPGAs—the bigger the better. To Xilinx, the MicroBlaze processor core is a loss leader. The more MicroBlaze cores in a design, the more programmable-logic cells the cores require, so the bigger the FPGA must be. That's why Xilinx practically gives away a royalty-free perpetual-use MicroBlaze license for only $495. The license permits developers to build single- or multicore designs, and it includes the synthesizable processor, a development kit, and documentation.

Practically speaking, there is no competitively priced alternative to MicroBlaze. ARM and MIPS Technologies, the leading processor-IP vendors, discourage or prohibit licensees from deploying their synthesizable processors in programmable-logic devices. (All processor-IP vendors allow licensees to synthesize their cores in FPGAs for testing and verification.) One exception is ARM's special arrangement with Actel, which allows Actel to offer an encrypted version of the ARM7TDMI-S for ProASIC3 and Fusion FPGAs. (See *MPR 12/19/05-02*, "Actel Releases First Fusion Chip.") ARC International and Tensilica are more open to FPGA deployment than ARM and MIPS are, but a multicore perpetual-use license for their synthesizable processors could cost hundreds of thousands of dollars.

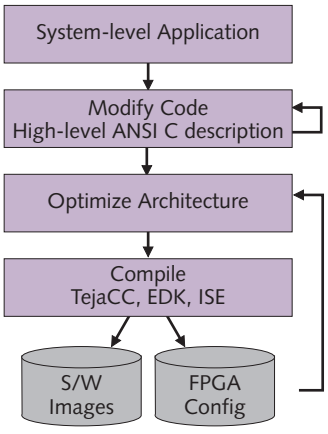

Figure 3. Hardware and software development with Teja FP is highly interactive. Using feedback from code profilers and actual execution in the target FPGA, developers can rapidly refine their design. The hardware/software partition is a movable wall: some operations that don't work well in software can be re-implemented in hardware, while other operations unsuitable for logic can be recompiled as software.

Only two other 32-bit processor cores are inexpensive enough to compete with the $495 MicroBlaze. One is Altera's Nios II, which is priced the same. (See *MPR 6/28/04-02*, "Altera's New CPU for FPGAs.") But the Nios II works only with Altera's programmable-logic devices. Currently, Teja FP isn't compatible with Altera's chips, and Altera doesn't offer an integrated hard core in the same class as the PowerPC 405. An even cheaper alternative is the SPARC-compatible LEON processor, which is freely licensed by the European Space Agency. Although some developers have successfully used LEON, it's not compatible with Teja's automated development tools, and it's not optimized for FPGA integration. (See *MPR 5/2/05-02*, "Storage Processor Leverages LEON.")

For developers, the almost-free MicroBlaze is a great bargain, but also a limitation. MicroBlaze is a plain-vanilla RISC architecture proprietary to Xilinx. ARC, ARM, MIPS, and Tensilica offer more variety and configurability at the core level, although the Xilinx FSL interface does allow developers to attach custom logic. Developers that already have legacy code for a packet processor have probably targeted the popular MIPS architecture or a proprietary network-processor architecture. If the legacy data-plane code is written in generic C, recompiling for MicroBlaze shouldn't be a problem with Teja CC. However, if the code significantly departs from ANSI C or calls any assembly-language routines, some porting will be in order. The same will be true for legacy code that doesn't cleanly segregate the control and data planes.

Likewise, the only embedded hard-core option for Virtex-4 is the PowerPC 405. Although the PowerPC 405 is perfectly capable of running a network operating system and other control-plane software, the only alternative is an external processor, which can be virtually any CPU architecture. The extra cost of an external processor will be partially offset by the lower cost of a Virtex-4 LX chip without the embedded PowerPC core. Another way to cut costs is to take advantage of the Xilinx EasyPath deal. If a particular design doesn't fill a Virtex-4 to the brim, customers can buy chips at a lower price—as much as 80% lower, in large volumes. Some of these chips may have defective logic cells, but Xilinx tests them to ensure that any bad cells aren't used by the customer's application. Customers don't have to change their design or perform any additional qualification.

Despite Teja FP's current limitations—the sole options of MicroBlaze for data-plane processing, the PowerPC 405 for integrated control-plane processing, and the Virtex-4 FX

family for the programmable-logic device—Teja offers a reasonable degree of design freedom and economy. Attempting to duplicate the functionality of Teja FP using other tools and IP would almost certainly cost more and would sacrifice the automation of Teja's tool chain.

A New Direction for Teja

Teja FP represents a new strategy for the privately held company. When Teja was founded in 1998, it focused on a network-processor operating system and high-level development tools. Those tools offered a novel graphical user interface for developing code. (See *MPR 5/14/01-01*, "Teja's Promise of Portability.")

However, as with other attempts to make coding resemble cartooning, the graphical tools didn't become as popular as Teja hoped they would. Indeed, Teja now says that programmers writing network software find that even an object-oriented language like C++ can be too abstract. Although Teja still supports the graphical tools, which are optional with Teja FP, the company is moving in a new direction. Familiar ANSI C will be the preferred front-end interface for writing code. On the back end, Teja's C-to-gates and C-to-binary compilers will ease the task of partitioning an application into hardware and software components.

Teja's new strategy rides an industry cost curve that makes programmable-logic devices a more viable solution with every passing year. FPGAs keep getting denser and cheaper, while custom silicon keeps getting more expensive.

Price & Availability

The Teja FP development platform is available in beta now. Final release is scheduled for 3Q06. The price is $15,000 per seat, but discounts are available for quantity purchases. Customers must separately license the $495 Embedded Development Kit from Xilinx. At this time, Teja FP works only with the Xilinx Virtex-4 FX family of programmable-logic devices, but Teja intends to target other Virtex-4 FPGAs and the less expensive Spartan family in the future. Teja offers ANSI C source code and hardware IP for implementing a Teja FP design that performs IPv4 and IPv6 packet forwarding. For more information about Teja FP, visit *www.teja.com/xilinx*.

Designs that would have been impractical in programmable logic only a short while ago are becoming not only practical but sometimes less costly as well. As FPGA prices drop, they become more economical in larger volumes, overtaking the falling volumes at which custom silicon makes sense. Although Teja is currently riding the highest-priced portion of that cost curve—the initial version of Teja FP works only with large, expensive FPGAs—at least the company is riding the curve in the right direction. ◇