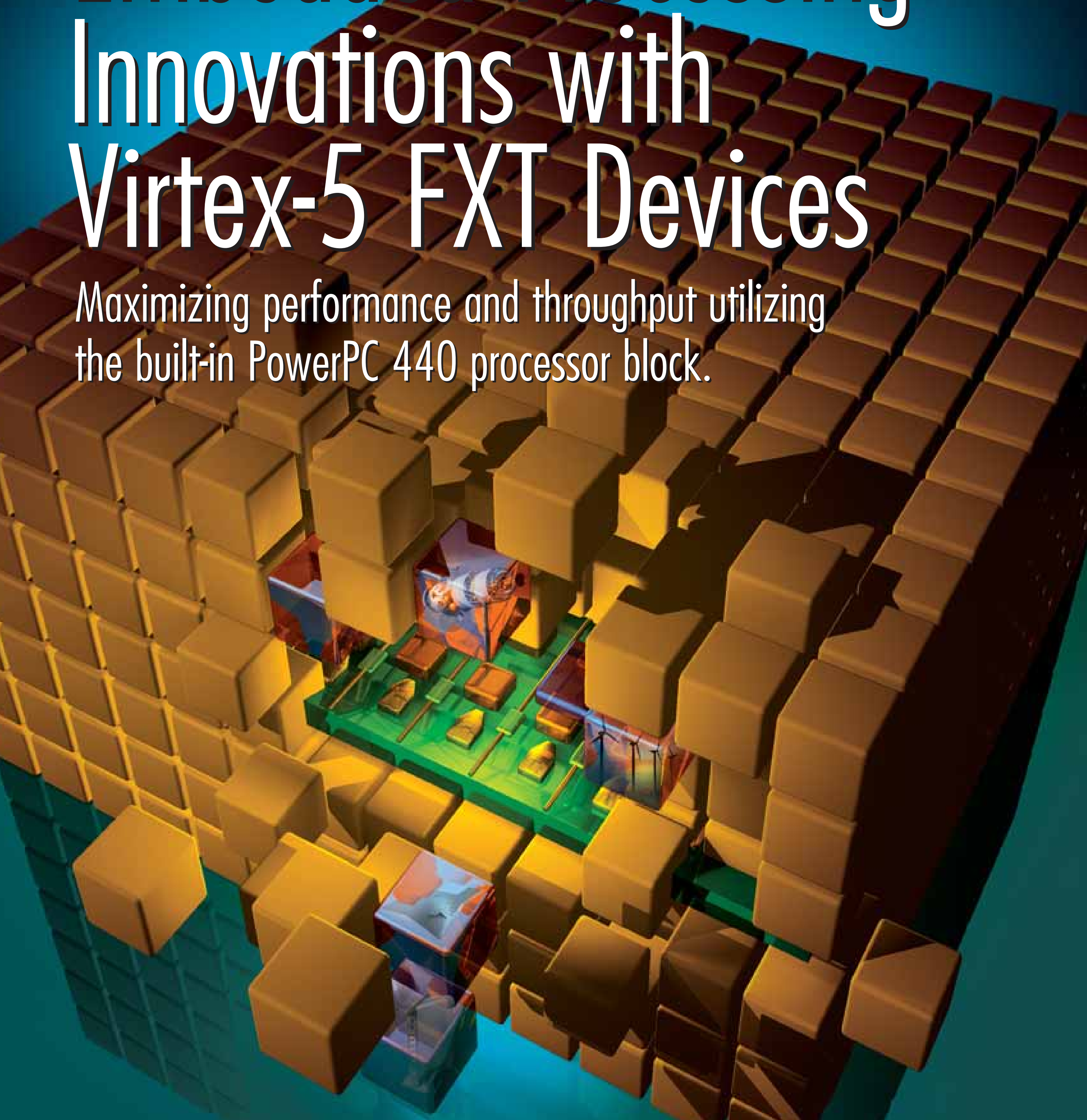
Embedded Processing Innovations with Virtex-5 FXT Devices

Maximizing performance and throughput utilizing the built-in PowerPC 440 processor block.



by Craig Abramson
Product Marketing Manager
Xilinx, Inc.
craig.abramson@xilinx.com

Dan Isaacs
Director of Embedded Processor Marketing
Xilinx, Inc.
dan.isaacs@xilinx.com

Ahmad Ansari
Principal Engineer
Xilinx, Inc.
ahmad.ansari@xilinx.com

With the advent of the Xilinx® Virtex™-5 FXT FPGA, you have an opportunity to get ahead of the embedded system design curve. The need to quickly develop and validate embedded systems has never been more apparent than in the realm of embedded system design.

Combining software and hardware to demonstrate this at a system level (as quickly as time permits) has become commonplace in the industry. By providing a more tightly coupled, flexible, scalable solution, you have a means to address many hardware and software SOC design challenges.

FPGAs provide a significantly faster path for designers to rapidly develop, prototype, and test their embedded designs. The Virtex-5 FXT device platform, the third-generation FPGA to feature a PowerPC processor, has added an embedded block that will help you meet more demanding design requirements while allowing you to finish your designs quickly and easily.

In this article, we'll provide a detailed description of the embedded processing innovations in the PowerPC 440 processor block and system interconnect. A key area of focus in the Virtex-5 FXT FPGA processor block is simplification through integration.

A corollary to this is ease of development and test. Quickly bringing up a system to allow software developers to get a head start on actual hardware is a major emphasis for the Virtex-5 FXT device's PowerPC 440 processor.

Simplification Through Integration

Integration is key. We have reduced the amount of FPGA logic needed to build a high-performance processing system while still allowing a wide variety of topologies. You still have the flexibility and advantages of an FPGA-based implementation, but you now also have the added benefit of a hardened, integrated interconnect architecture that (among other things) maximizes access to external memory.

As you will see, the result is an embedded block that allows you to develop a wider range of high-performance processing architectures in a shorter period of time.

PowerPC processors generically have three interfaces: instruction read, data read, and data write. In previous Virtex device architectures, which embedded the PowerPC 405, these processor buses would connect to FPGA fabric. The timing closure requirements of this circuitry would vary based on how many and what types of loads the design presented to the buses.

In the Virtex-5 FXT FPGA (where the processor is now the PowerPC 440), these buses are hardened and hooked directly to a new structure, an integrated 5 x 2 crossbar switch – generically referred to as the crossbar. This hardened interconnect provides significantly higher performance (with virtually no consumption of FPGA logic resources and fixed timing) when combined with the rest of the architectural enhancements in the Virtex-5 FXT device's embedded processor block. This results in an overall system cost reduction and invariably a more tightly integrated processor system.

The processor buses only take up three of the five "crossbar master" ports on the 5 x 2 crossbar (see Figure 1). The crossbar includes two additional master ports, because in many real-world applications it's not just the processor that needs access to memory or peripherals. Each of these "crossbar master" ports comprises a processor local bus (PLB) slave interface, as well as two channels of scatter-gather direct memory access (DMA).

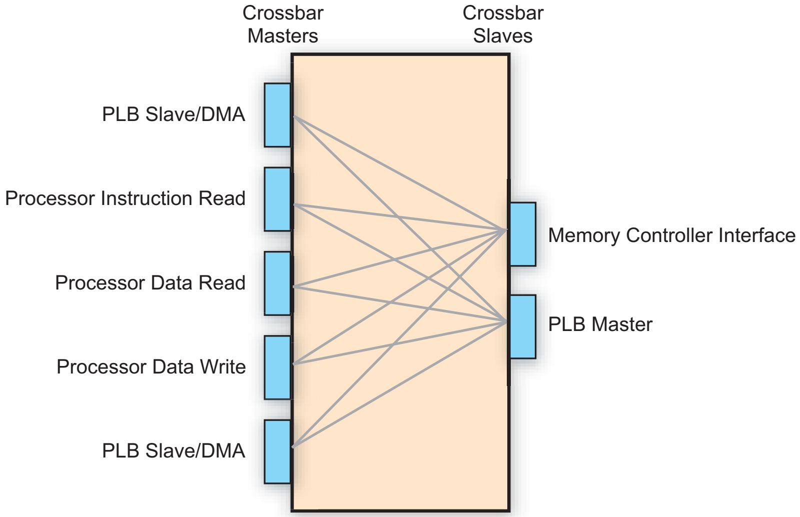


Figure 1 – The crossbar

The "slave" side of the crossbar comprises two ports. One port is a dedicated memory controller interface that provides a high-throughput generic interface to soft memory controllers. The other is a bus for attaching I/O devices and peripherals.

A Better Processor

Providing all of this extra functionality in the embedded processor block would be of little consequence if there were not a processor with the horsepower to take advantage of it. The Virtex-5 FXT FPGA represents the first time anyone has embedded a PowerPC 440-class processor in an FPGA.

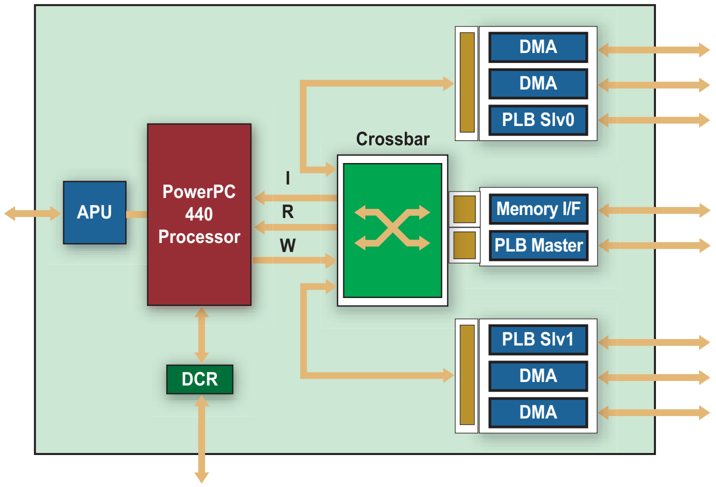


Figure 2 – The PowerPC 440 embedded processor block

The PowerPC 440 offers a significant performance improvement over the PowerPC 405 (which was embedded in previous Virtex families) in a number of areas.

First, the PowerPC 440, when in the fastest speed-grade FPGA, can be clocked at 550 MHz. The PowerPC 405 topped out at 450 MHz. This is almost a 20% performance improvement. But add to that the fact that the I and D cache sizes are doubled, the instruction pipeline is seven instead of five stages, and the execution unit can now execute two instructions out of order and in parallel.

The result? You've got a processor with performance sufficient to handle a great many of today's embedded processing challenges.

There are a number of other advantages to moving from the PowerPC 405 to the PowerPC 440, as shown in Table 1.

The PowerPC 440 embedded block is shown in Figure 2.

High-Throughput Switch Matrix

The 5 x 2 crossbar is more than just a big switch. It provides non-blocking pipelined access from the five crossbar masters to the two crossbar slaves (see Figure 1). It allows concurrent transfers between different agents on the crossbar at the same time.

As shown, we'll call the buses going into the crossbar "crossbar masters" and the buses coming out "crossbar slaves." These interfaces are highly pipelined, thus allowing a large number of transactions to be in progress at the same time.

In fact, up to four concurrent transactions can exist: two for each crossbar slave (such as the memory controller or PLB master). Additionally, each crossbar master (that is, the three processor PLBs and the two PLB slave interfaces) can pipeline four read and four write transactions to the same slave.

Another key feature of the crossbar is its highly programmable memory mapping. You can think of the entire system of having available memory space of 4 GB. Both the memory controller interface and the PLB master can have different memory windows mapped into the memory space of any of the crossbar masters. These memory spaces can be programmed through the FPGA bitstream, by the processor at run time, or even by external logic on the FPGA using the crossbar's sideband bus, called the device control register (DCR) bus.

Integrated PLB Interfaces

As we mentioned earlier, many of the buses connected to the crossbar are processor local buses, also called PLBs.

The PLB is one of the standard CoreConnect buses as defined by IBM. An earlier version of the PLB (version 3.4) was used as one of the standard buses on PowerPC 405 designs in Virtex-II Pro and Virtex-4 FX FPGAs and is also used in the new PowerPC 440 embedded processor block.

In the PowerPC 440 embedded processor block, the PLBs connect the processor's internal caches to the input side of the crosspoint. The buses are:

- ICURD: instruction cache unit read
- DCURD: data cache unit read
- DCUWR: data cache unit write

	PPC405 **(Virtex-4 FX FPGA)**	**PPC440** **(Virtex-5 FXT FPGA)**	**Benefit**
Architecture	32-bit instruction, 32-bit address, 64-bit data	32-bit instruction, 36-bit address, 128-bit data, Book E compliant	Access more physical memory, higher speed data movement
Pipline	Single instruction/cycle, five-stage pipeline, in-order issue	Two instructions/cycle, seven-stage pipeline, out-of-order issue	More efficient instruction execution
Caches – I/D	16K/16K, two-way set associative, no locking	32K/32K, 64-way set associative, locking	Less memory access latency
MMU	Page size: 1 KB to 16 MB	Page size: 1 KB to 256 MB	Less page swapping
DMPS Estimate	700+ DMIPS	1000+ DMIPS	Better benchmarks equal higher performance

Table 1 – Virtex PowerPC integrated processor feature comparison

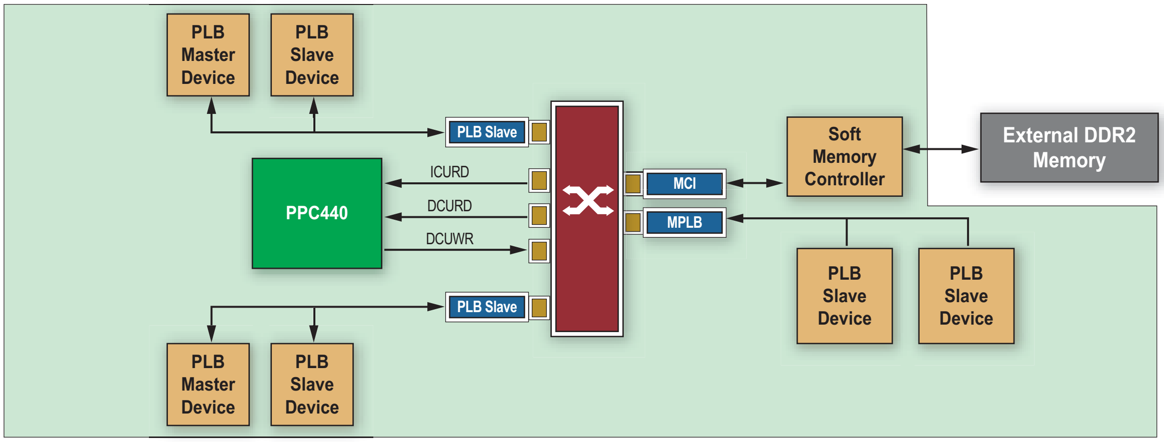


Figure 3 – PLB masters and slaves

The PLB used in the Virtex-5 FXT device is version 4.6 (PLB46). The PLB46 bus architecture brings with it a number of new capabilities that give it a nice performance boost over its predecessor. The most obvious is the fact that while PLB34 was 64 bits, PLB46 is 128 bits. But not to worry – if the IP connected to the bus is less than that, the bus will perform dynamic bus sizing as required to accommodate 32- and 64-bit transactions.

We should also point out that the PLB46 version is a Xilinx implementation of the IBM-defined PLB46, optimized to take advantage of FPGA resources.

PLB46 – and indeed all versions of PLB – have the concept of master and slave. This should not be confused with crossbar master and crossbar slave. (Again, refer to Figure 1.) As we stated earlier, there are two PLB slave port interfaces on the crossbar; they are crossbar masters. These slave ports are connected to the FPGA fabric.

In a processor system there is often the need to allow something besides the processor to access external memory or on-chip peripherals. The PLB slave interfaces allow just that. PLB masters, built from FPGA logic, connected to the PLB Slave ports (see Figure 3) can access either the MCI or the MPLB through the crossbar.

Similarly, the function of the PLB master (the one that is the crossbar slave) is to have a PLB to hook to I/O devices and soft peripherals. And because the PLB master is a crossbar slave, anything hooked to a crossbar master port can access it.

Note that there can be no more than four PLB masters connected to each PLB slave bus. Few systems are likely to need more than four masters, but if you did need more, you could always use the PLB/PLB bridge IP provided with the Embedded Development Kit (EDK) (see *www.xilinx.com/support/documentation/ipembedprocess_coreconnect_plbbusstruct.htm*).

Figure 3 is a simplified system diagram showing how PLB peripherals can be hooked to crossbar master and crossbar slave ports. Note that if you have multiple masters on any PLB, arbitration is handled by the IP for the bus. You do not need a separate arbiter.

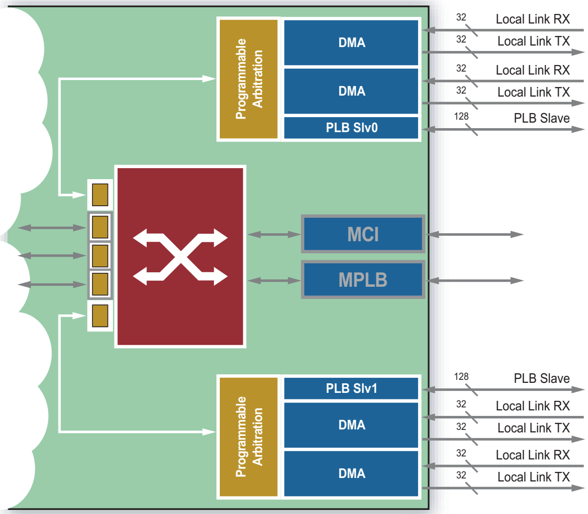
Optimized DMA Engines

There are four additional crossbar masters; they are the four DMA channels. Each DMA channel has separate 32-bit transmit and 32-bit receive interfaces. They share crossbar arbitration with PLB slave interfaces, as shown in Figure 4.

All DMA ports can be operating at the same time. Each one has a dedicated FIFO, so as one DMA is accumulating data, the other DMA can be pumping data through the crossbar. Each DMA channel operates asynchronously to the processor clock.

The interface into the DMA channels is through an interface called LocalLink. Xilinx uses the LocalLink interface in a

Figure 4 – PLB slave buses and DMA channels share crossbar arbitration.



> To maximize throughput and performance, the PowerPC 440 embedded block employs scatter/gather DMA. To make using this capability as easy as possible, Xilinx provides wrappers for the various pieces of IP and embedded blocks it offers.

number of IP blocks. LocalLink is a point-to-point interface that sends packets to, or receives packets from, some external device.

The most notable type of processor IP that uses the LocalLink interface is the hard embedded tri-mode Ethernet media access controller (TEMAC) block. The TEMAC has a wrapper that allows it to communicate directly with the PowerPC 440 DMA.

Although all data paths through the crossbar are 128 bits, the LocalLink interface into and out of the DMA channels are all 32 bits. As such, there is built-in logic between the DMA controller and the crossbar that realigns data.

To maximize throughput and performance, the PowerPC 440 embedded block employs scatter/gather DMA. To make using this capability as easy as possible, Xilinx provides wrappers for the various pieces of IP and embedded blocks it offers.

The first one targeted specifically toward the PowerPC 440 is the soft wrapper for the embedded TEMAC blocks. This wrapper, combined with the functionality of the DMA engine in the PowerPC 440 embedded block, allows you to easily build a processing system with a high-performance TEMAC connected directly to the PowerPC 440 DMA channels. Figure 5 is a simplified system showing how both DMA and PLB peripherals can be hooked to crossbar master and crossbar slave ports.

The DMA channels are controlled by descriptors, small blocks of memory that are set up by the PowerPC 440 processor before commencing DMA operations. The descriptors control how much data is transferred and where data is located in system memory.

Descriptors can be chained together if need be, effectively creating a sequence of commands to control a DMA channel. The DMA controller is covered in its entirety in the reference guide, entitled "Embedded Processor Block in Virtex-5 FPGAs" (*http://www.xilinx.com/support/documentation/user_guides/ug200pdf).*

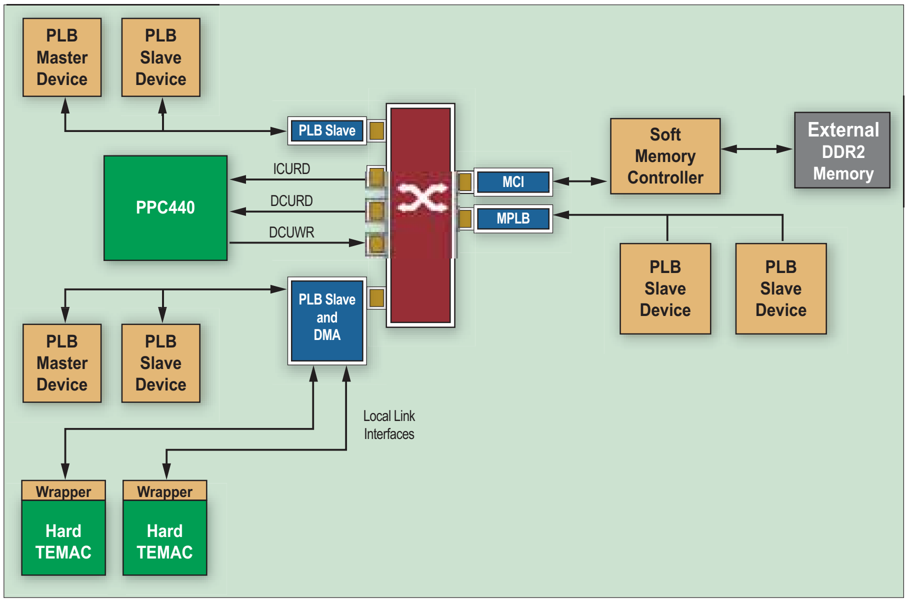


Figure 5 – Sample system with both PLB and DMA peripherals

High-Performance Dedicated Memory Interface

Rounding out the new processor block is the dedicated memory controller interface. The purpose of this interface is to provide a dedicated link out to external memory, but at the same time not be tied to any specific memory technology.

At this time, the memory controller interface supports a stand-alone DDR2 controller and MPMC4 controller, all available through Xilinx Platform Studio, EDK 10.1. This interface provides the flexibility to connect to virtually any memory technology now or in the future.

The memory controller interface is streamlined, comprising address/data/control. It can be programmed to support 128-, 64-, 32-, or even 16-bit memory. It does width and burst realignment, so while the DMA may be bursting one size packet, the memory controller can buffer and realign the packet data to maximize the bandwidth to the memory. Burst size is programmable and can be 1, 2, 4, or 8, and the memory controller interface will automatically adjust the address to accommodate the various burst widths.

The majority of soft memory controller handshaking signals are generated by the interface on behalf of the memory controller. They are provided ahead of time such that the soft memory controller can generate throttling signals back to the memory interface. The memory controller interface – on behalf of the soft memory controller – can also be programmed to detect bank and row misses ahead of time and will inform the soft memory controller to anticipate a bank or row miss. All of these features together provide a solution whose primary goal is to maximize memory throughput.

Tuning the System

In some situations, a PLB or DMA interface just may not be the right solution. For

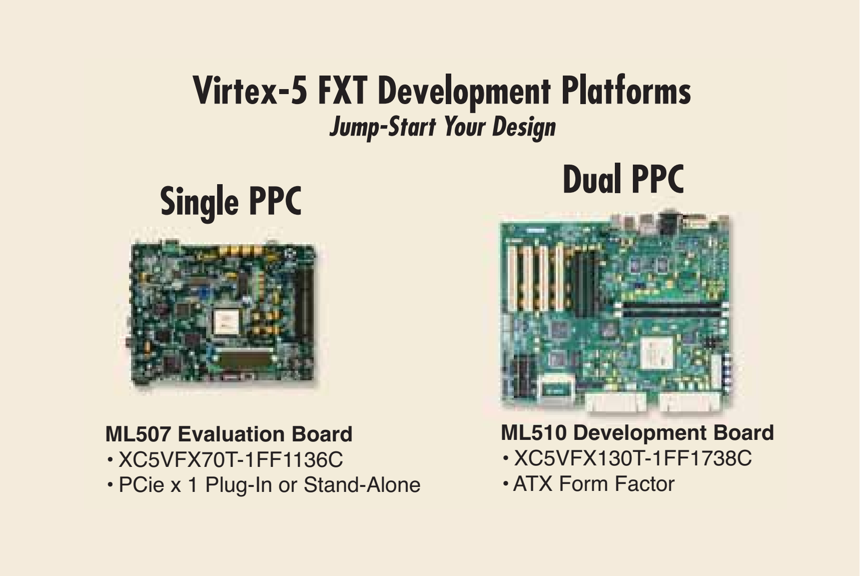


Figure 6 – Xilinx ML507 evaluation and ML510 development boards

instance, you might find that you have a software algorithm that takes too many cycles to execute and is affecting your system bandwidth. That algorithm is a great candidate for implementation in hardware, and the interface to which you may want that hardware connected would be the auxiliary processing unit, or APU interface.

The PowerPC 440 has a second-generation APU interface that is tightly coupled to the execution units of the processor. The interface is controlled by 16 user defined instructions (UDIs). The data path of the APU interface is 128 bits.

Perhaps the most common use of the APU interface is for connecting to a floating-point unit (FPU). The FPU is IEEE754-compatible and supports both single- and double-precision operations for the PowerPC 440.

The FPU is implemented in the FPGA soft logic fabric and utilizes the DSP48E blocks. The soft logic implementation operates up to half the frequency of the hard embedded processor.

Other uses of the APU interface include hardware algorithm acceleration, as well as an alternative high-bandwidth link to block RAM.

Configuring the Embedded Block

By integrating the PowerPC 440 block in the FPGA, the processor block can be configured in multiple ways. Virtually every interface is programmable.

For example, when you build your processing system in the Xilinx Platform Studio development environment and a bitstream is created, all of the specifications of the processing system are in the bitstream. Thus, when the FPGA starts up, your processor is up and running.

Now, let's say the processing system is up and running and you want to modify the operation of one of the DMA channels. You would do that through the DCR interface. There are DCR registers to control every aspect of DMA operation.

In fact, there is DCR access to virtually every other subsystem of the embedded block: the PLBs and crossbar, memory controller interface, and the APU controller. Refer to Figure 2 for more details.

Putting It All Together

This innovation would be for naught if Xilinx did not provide a comprehensive infrastructure to take advantage of all of the architectural enhancements. We should point out that the Virtex-5 FXT FPGA with the PowerPC 440 block represents our eighth year in embedded processing and our third-generation FPGA with a hardened processor.

Throughout that time we've been constantly updating EDK, our award-winning Embedded Development Kit. EDK includes Platform Studio, with its comprehensive library of IP for hardware design, and Platform Studio SDK, a software development environment familiar to many embedded software engineers.

With the introduction of the Virtex-5 FXT family of devices, we continue to further strengthen our third-party alliances with support from industry-leading operating system providers, including WindRiver Systems with VxWorks and Green Hills Integrity.

Linux support is provided through LynuxWorks, Monta Vista, and WindRiver Systems. In addition, Xilinx recognizes the importance of open-source Linux, and we're moving forward on that front.

Xilinx and its partner companies are also developing a wide variety of boards. Xilinx has multiple boards for the Virtex-5 FXT device: the ML507 with the XC5VFX70T and the ML510 with the XC5VFX130T, as shown in Figure 6. The ML507 evaluation platform enables your team to quickly begin developing hardware, software, or both. When multiple processors or a motherboard-type platform are required, the ML510 with the dual-processor XC5VFX130T is ideal.

Conclusion

A high-performance processing solution with optimized data throughput is high on the wish list of embedded designers everywhere. This is true whether you are running critical algorithms at the heart of the latest wireless base station, switching high-bandwidth data through a video switch, performing advanced signal processing for guidance systems using coprocessor acceleration, or handling complex control and system management tasks.

The Virtex-5 FXT embedded processor block, with a multi-ported non-blocking integrated processor interconnect and high-performance integrated DMA, offers a solution that allows you to focus on the key elements of your embedded design.

With a virtually unlimited number of ways to harness these embedded capabilities, the Virtex-5 FXT FPGA embedded processing solution provides a highly integrated platform for high-performance, high-throughput SOC designs.