Considerations for High-Bandwidth TCP/IP PowerPC Applications

The Xilinx Gigabit System Reference Design maximizes TCP/IP performance.

by Chris Borrelli
Embedded Networking Manager
Xilinx, Inc.
chris.borrelli@xilinx.com

The TCP/IP protocol suite is the de facto worldwide standard for communications over the Internet and almost all intranets. Interconnecting embedded devices is becoming standard practice even in device classes that were previously stand-alone entities.

By its very definition, an embedded architecture has constrained resources, which is often at odds with rising application requirements. Achieving wire-speed TCP/IP performance continues to be a significant engineering challenge, even for high-powered Intel™ Pentium™-class PCs.

In this article, we'll discusses the per-byte and per-packet overheads limiting TCP/IP performance and present the techniques utilized in the Xilinx Gigabit System Reference Design (GSRD) to maximize TCP/IP over Gigabit Ethernet performance in embedded PowerPC™-based applications.

GSRD Overview

The GSRD terminates IP-based transport protocols such as TCP or UDP. It incorporates the embedded PowerPC and RocketIO™ blocks of the Virtex-II Pro™ device family, and is delivered as an Embedded Development Kit (EDK) reference system.

The reference system as described in Xilinx Application Note XAPP536 leverages a multi-port DDR SDRAM memory controller to allocate memory bandwidth between the PowerPC processor local bus (PLB) interfaces and two data ports. Each data port is attached to a direct memory access (DMA) controller, allowing hardware peripherals high-bandwidth access to memory.

A MontaVista™ Linux™ port is available for applications requiring an embedded operating system, while a commercial standalone TCP/IP stack from Treck™ is also available to satisfy applications with the highest bandwidth requirements.

System Architecture

Memory bandwidth is an important consideration for high-performance network-attached applications. Typically, external DDR memory is shared between the processor and one or more high-bandwidth peripherals such as Gigabit Ethernet.

The four-port multi-port memory controller (MPMC) efficiently divides the available memory bandwidth between the PowerPC's instruction/data PLB interfaces and a communications direct memory access controller (CDMAC). The CDMAC provides two bi-directional channels of DMA that connect to peripherals through a Xilinx standard LocalLink streaming interface. The CDMAC implements data re-alignment to support arbitrary alignment of packet buffers in memory. A block diagram of the system is shown in Figure 1.

The LocalLink Gigabit Ethernet MAC (LLGMAC) peripheral incorporates the UNH-tested Xilinx LogiCORE™ 1-Gigabit Ethernet MAC to provide a 1 Gbps 1000-BASE-X Ethernet interface to the reference system. The LLGMAC implements checksum offload on both the transmit and receive paths for optimal TCP performance. Figure 2 is a simplified block diagram of the peripheral.

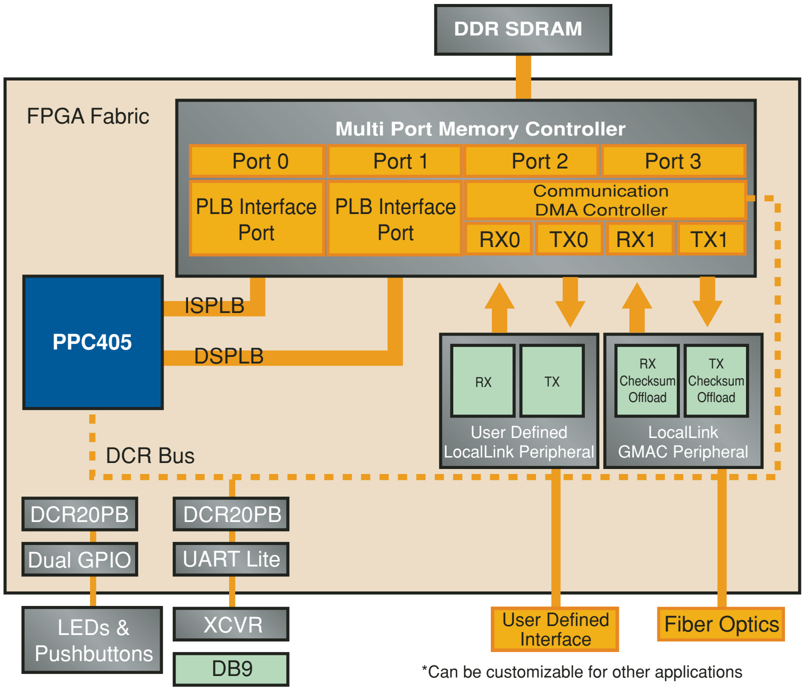

Figure 1 – GSRD system block diagram

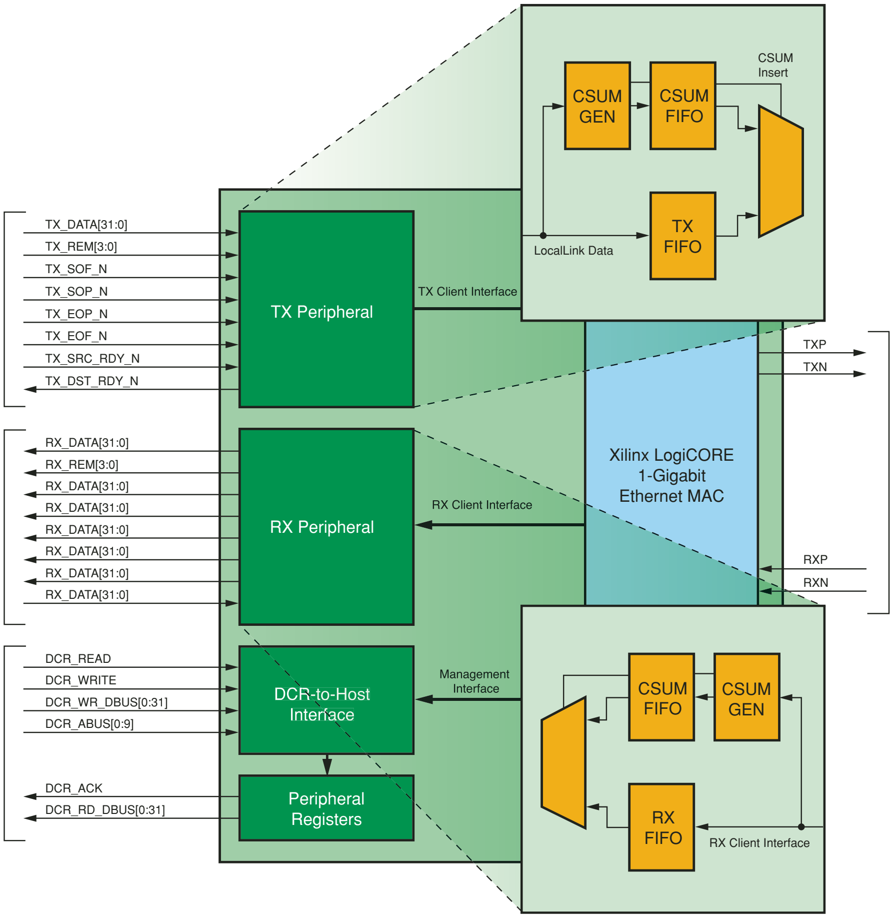

Figure 2 – LocalLink Gigabit Ethernet MAC peripheral block diagram

TCP/IP Per-Byte Overhead

Per-byte overhead occurs when the processor touches payload data. The two most common operations of this type are buffer copies and TCP checksum calculation. Buffer copies represent a significant overhead for two reasons:

1. Most of the copies are unnecessary.
2. The processor is not an efficient data mover.

TCP checksum calculation is also expensive, as it is calculated over each payload data byte.

Embedded TCP/IP-enabled applications such as medical imaging require near wire-speed TCP bandwidth to reliably transfer image data over a Gigabit Ethernet network. The data is generated from a high-resolution image source, not the processor.

In this case, introducing a zero-copy software API and offloading the checksum calculation into FPGA fabric completely removes the per-byte overheads. "Zero-copy" is a term that describes a TCP software interface where no buffer copies occur. Linux and other operating systems have introduced software interfaces like sendfile() that serve this purpose, and commercial standalone TCP/IP stack vendors like Treck offer similar zero-copy features. These software features allow the removal of buffer copies between the user application and the TCP/IP stack or operating system.

The data re-alignment and the checksum offload features of GSRD provide the hardware support necessary for zero-copy functionality. The data re-alignment feature is a flexibility of the CDMAC that allows software buffers to be located at any byte offset. This removes the need for the processor to copy unaligned buffers.

Checksum offload is a feature of the LocalLink Gigabit Ethernet (LLGMAC) peripheral. It allows the TCP payload checksum to be calculated in FPGA fabric as Ethernet frames are transferred between main memory and the peripheral's hardware FIFOs. GSRD removes the need for costly buffer copies and processor checksum operations, leaving the PowerPC 405 to process only protocol headers.

TCP/IP Per-Packet Overhead

Per-packet overhead is associated with operations surrounding the transmission or reception of packets. Packet interrupts, hardware interfacing, and header processing are examples of per-packet overheads.

Interrupt overhead represents a considerable burden on the processor and memory subsystem, especially when small packets are transferred. Interrupt moderation (coalescing) is a technique used in GSRD to alleviate some of this pressure by amortizing the interrupt overhead across multiple packets. The DMA engine waits until there are n frames to process before interrupting the processor, where n is a software-tunable value.

Transferring larger sized packets (jumbo frames of 9,000 bytes) has a similar effect by reducing the number of frames transmitted, and therefore the number of interrupts generated. This amortizes the per-packet overhead over a larger data payload. GSRD supports the use of Ethernet jumbo frames.

The components of GSRD use the device control register (DCR) bus for control and status. This provides a clean interface to software without interfering with the high-bandwidth data ports. The per-packet features of GSRD help make efficient use of the processor and improve system-level TCP/IP performance.

TCP/IP Stack	Ethernet Frame Size	Optimization	TCP Transmit Bandwidth
MontaVista Linux	9000 bytes (jumbo)	None	270 Mbps
MontaVista Linux	9000 bytes (jumbo)	Zero-copy, checksum offload	540 Mbps
Treck, Inc	9000 bytes (jumbo)	Zero-copy	490 Mbps
Treck, Inc	9000 bytes (jumbo)	Zero-copy, checksum offload	780 Mbps

Table 1 – TCP transmit benchmark results

Conclusion

The Xilinx GSRD is an EDK-based reference system geared toward high-performance bridging between TCP/IP-based protocols and user data interfaces like high-resolution image capture or Fibre Channel. The components of GSRD contain features to address the per-byte and per-packet overheads of a TCP/IP system.

Table 1 details the GSRD TCP transmit performance with varying levels of optimization for Linux and standalone Treck stacks.

Future releases of GSRD will explore further opportunities for TCP acceleration using the FPGA fabric to offload functions such as TCP segmentation.

The GSRD Verilog™ source code is available as part of Xilinx Application Note XAPP536. It leverages the MPMC and CDMAC detailed in Xilinx Application Note XAPP535 to allocate memory bandwidth between the processor and the LocalLink Gigabit Ethernet MAC peripheral. The MPMC and CDMAC can be leveraged for PowerPC-based embedded applications where high-bandwidth access to DDR SDRAM memory is required.

For more information about XAPP536 and XAPP535, visit *www.xilinx.com/gsrd/*.

Associated Links:

Xilinx XAPP536, "Gigabit System Reference Design"
http://direct.xilinx.com/bvdocs/appnotes/xapp536.pdf

Xilinx XAPP535, "High Performance Multi Port Memory Controller"
http://direct.xilinx.com/bvdocs/appnotes/xapp535.pdf

Treck, Inc. (*www.treck.com*)

MontaVista Software (*www.mvista.com*)

"End-System Optimizations for High-Speed TCP" (*www.cs.duke.edu/ari/publications/end-system.pdf*)

"Use sendfile to optimize data transfer" (*http://builder.com.com/5100-6372-1044112.html*)