

Overview

- Introduction to TCP/IP Protocol Suite
- Embedded TCP/IP Applications
- TCP Termination Challenges
- TCP Acceleration Techniques

Overview

- **Introduction to TCP/IP Protocol Suite**
- Embedded TCP/IP Applications
- TCP Termination Challenges
- TCP Acceleration Techniques

Why do we care about TCP/IP?

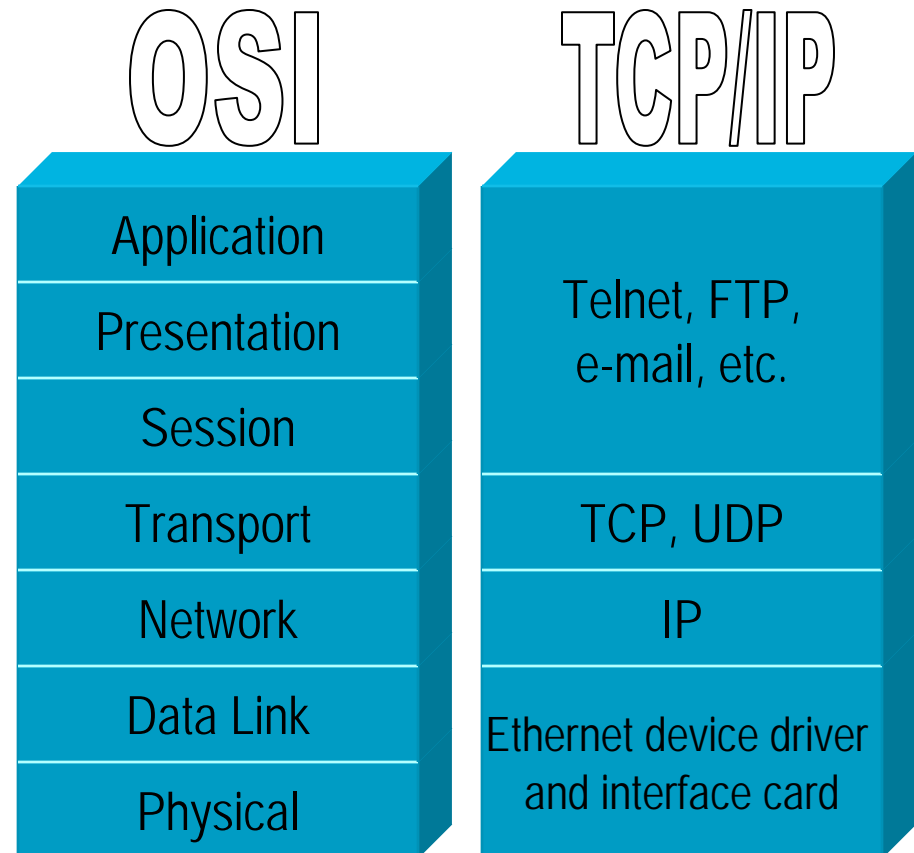
- TCP/IP is the most popular networking protocol
 - TCP has become the de-facto standard for data communications between computer systems
- Ethernet is the most popular LAN technology
 - 95%* of all LAN implementations use Ethernet
 - Over 750 million* Ethernet ports deployed worldwide

* Source: Instat/MDR



TCP/IP Defined

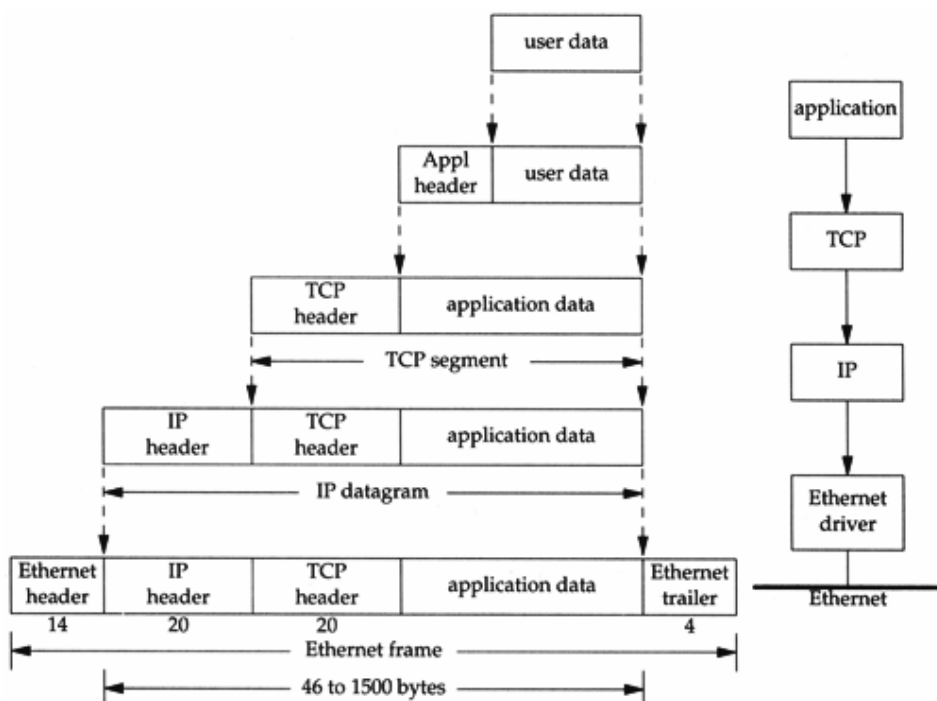
- TCP/IP is a suite of protocols that allow computer systems to communicate with each other
 - TCP/IP is an open standard
 - Internet Engineering Task Force
 - <http://www.ietf.org/>
- Protocol suite is layered
 - **Application:** handles details of particular application
 - **Transport:** provides flow of data
 - **Network:** handles movement (routing) of packets around the network
 - **Data Link:** device driver + network interface



TCP/IP Protocols

- TCP : Transmission Control Protocol [RFC 793]
 - Connection oriented, reliable, byte stream transport
- UDP : User Datagram Protocol [RFC 768]
 - Simple datagram oriented, unreliable transport
- IP : Internet Protocol [RFC 791]
 - Connectionless datagram delivery service
 - TCP and UDP data are transmitted as IP datagrams

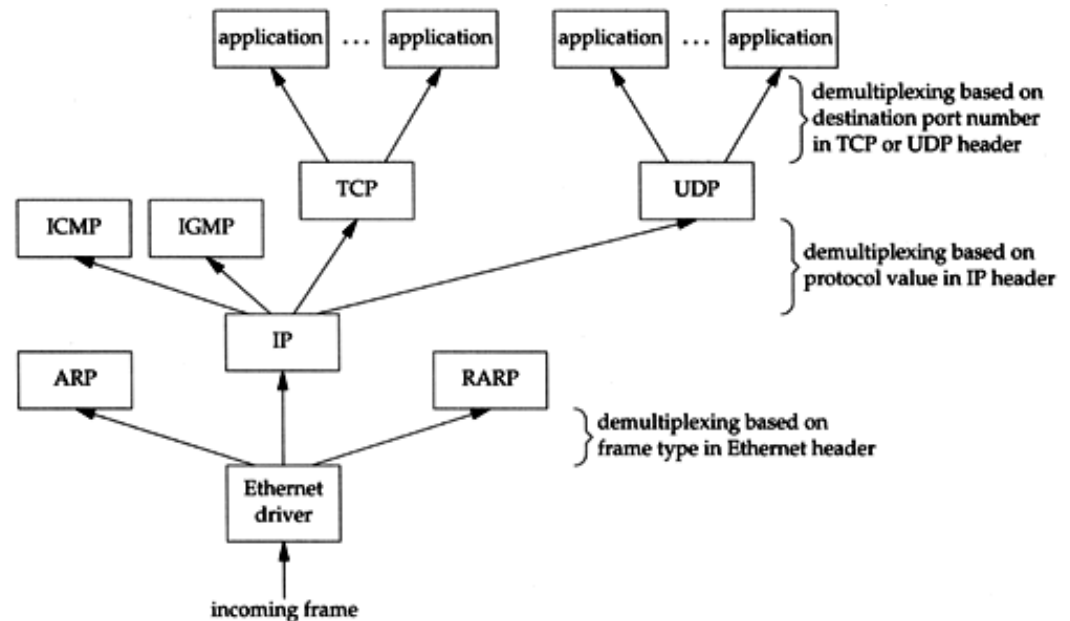
Protocol Encapsulation



- Data is sent down protocol stack through each layer until it is finally sent over Ethernet media
- Each layer prepends a header that helps to provide the service of the each protocol
- **TCP Segment**: data that TCP layer sends to IP layer
- **IP datagram**: data that IP layer sends to network interface
- **Ethernet Frame**: stream of bits sent over Ethernet media

Protocol Demultiplexing

- Ethernet frames received on network interface travel up the protocol stack
- Headers are removed and parsed by each successive protocol layer
- Identifiers in the headers determine which next upper layer should receive the data
- Each TCP Connection can uniquely be identified by 4 pieces of information (*a socket*):
 - Client IP address
 - Client port number
 - Server IP address
 - Server port number



TCP/IP Summary

- TCP/IP is a suite of protocols that allow computer systems to communicate with each other
- Protocols are layered (like OSI model)
- Main Protocols that make up TCP/IP Protocol Suite
 - TCP, UDP, IP

Overview

- Introduction to TCP/IP Protocol Suite
- **Embedded TCP/IP Applications**
- TCP Termination Challenges
- Further TCP Acceleration Techniques
- TCP/IP Customer Ready Demo

High Resolution Imaging over TCP



Image Protocol Interface to Imaging device



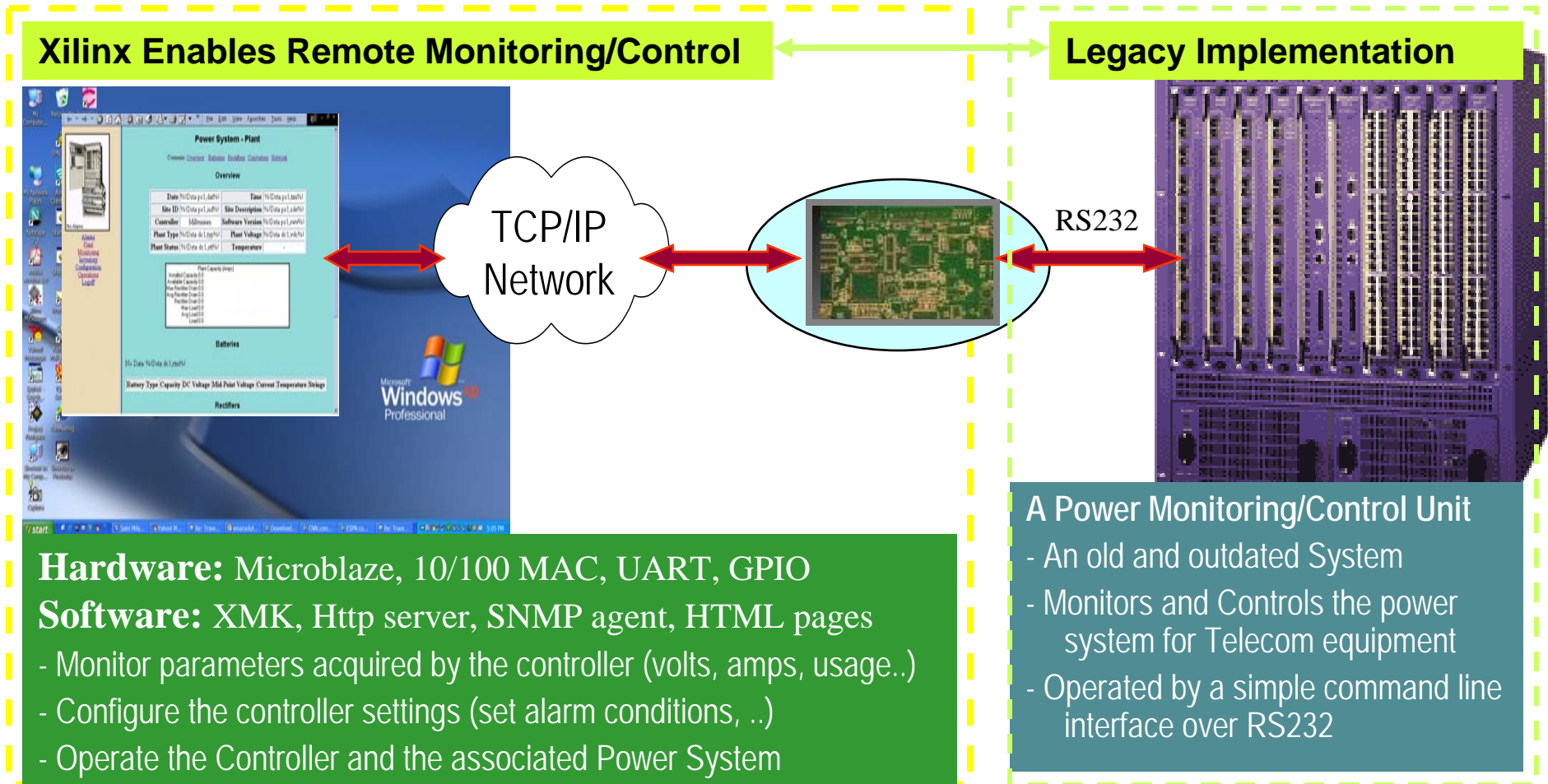
Ethernet interface to Workstation



PowerPC takes data from Imaging device and transfers to remote Workstation using TCP

Workstation stores image data and performs post processing

Remote Monitoring & Control



Overview

- Introduction to TCP/IP Protocol Suite
- Embedded TCP/IP Applications
- **TCP Termination Challenges**
- TCP Acceleration Techniques

System Level Considerations

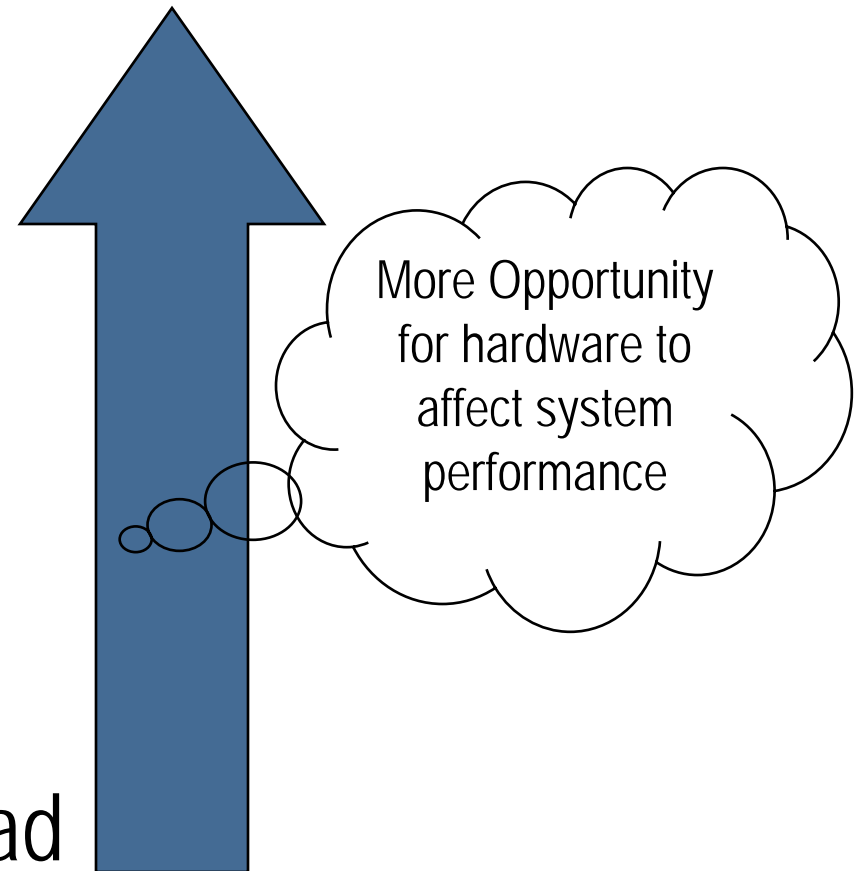
- Memory Bandwidth
 - Lots of memory bandwidth is required to support wire-speed TCP/Ethernet plus processor instruction/data fetches with reasonable latency
- CPU Performance
 - PowerPC in Xilinx FPGA ~300MHz
 - General Rule of thumb – 1 MHz of CPU per 1 Mbits of TCP
 - PPC can't touch payload data bytes
 - PPC can not produce or consume TCP payload
 - So... compression or any other byte touching operation is out

What Limits TCP Performance?

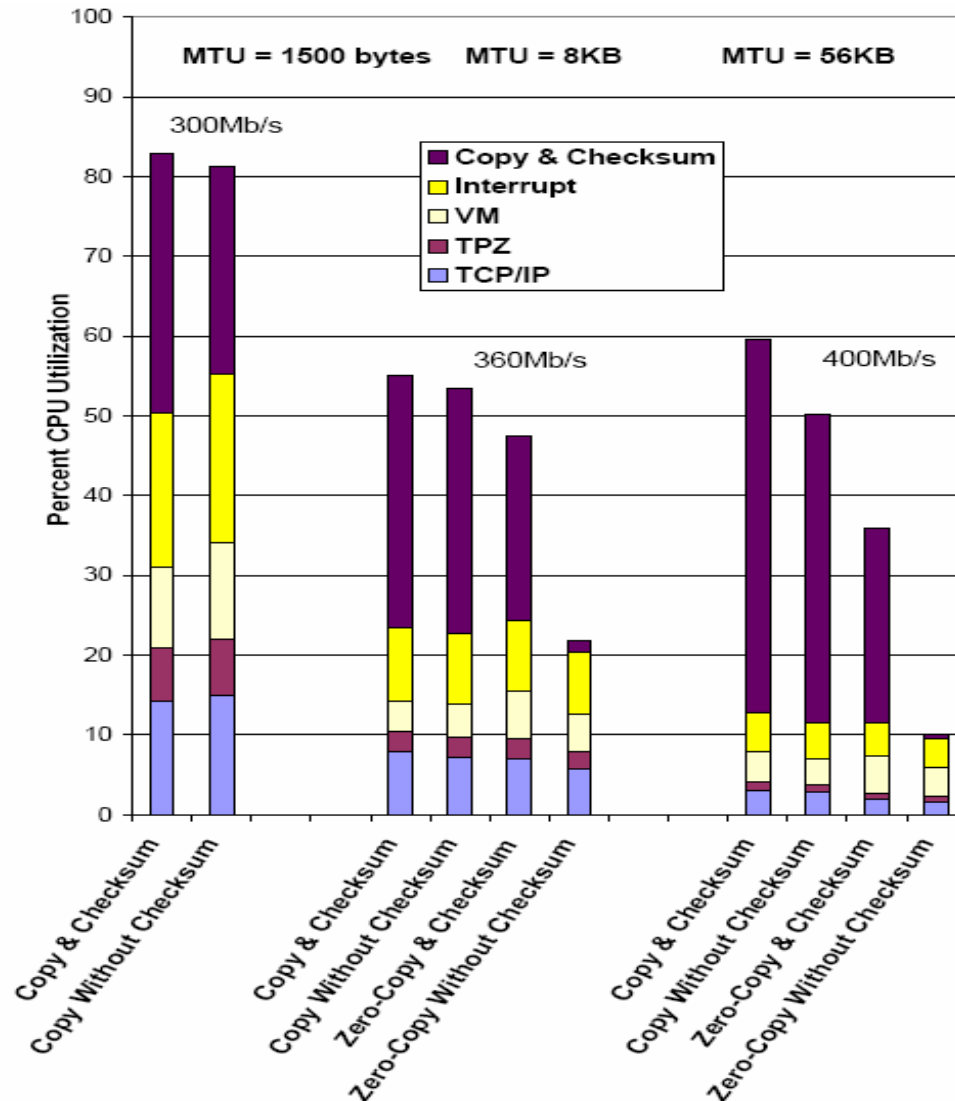
- Supporting functions are source of most overhead
- TCP protocol processing is not to blame
- Research Papers:
 - *End-System Optimizations for High-Speed TCP*
 - <http://www.cs.duke.edu/ari/publications/end-system.pdf>
 - *TCP/IP at Near-Gigabit Speeds*
 - <http://www.cs.duke.edu/ari/publications/tcpgig.pdf>
 - *TCP Performance Re-Visited*
 - <http://www.cs.duke.edu/~jaidev/papers/ispass03.pdf>

Classification of TCP/IP Overheads

- Per-Byte Overhead
 - Data buffer copies
 - Checksum calculation
- Per-Packet Overhead
 - Interrupt overhead
 - Buffer Management
 - Protocol Processing
- Per-Connection Overhead



Per-Byte vs. Per-Packet Overheads



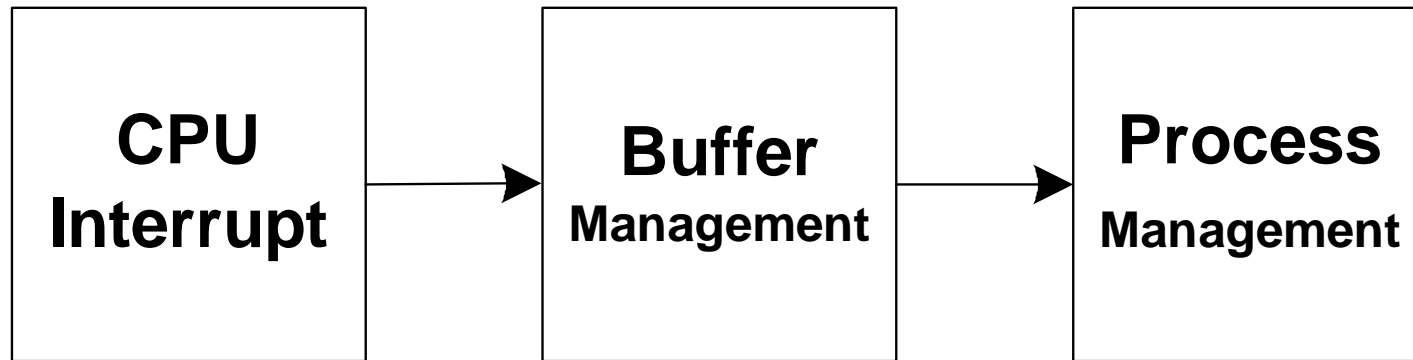
<http://www.cs.duke.edu/ari/publications/tcpgig.pdf>



Per-Byte Overhead

- “Sockets” API uses buffer copying to queue user application data (user-to-kernel space)
- Current TCP stacks have zero-copy APIs
 - For example, Linux has `sendfile()`
- Additional buffer copies are also required if Ethernet Interface’s DMA engine restricts alignment of data buffers
- TCP checksum requires 16-bit calculations over header and **payload data bytes**

Per-Packet Overhead



- RX and TX Interrupts impose high overhead
- Gigabit Ethernet
 - 1518 byte frame transferred every 12us
 - 3651 CPU cycles between frames (300Mhz CPU)

Gigabit System Reference Design (XAPP536)

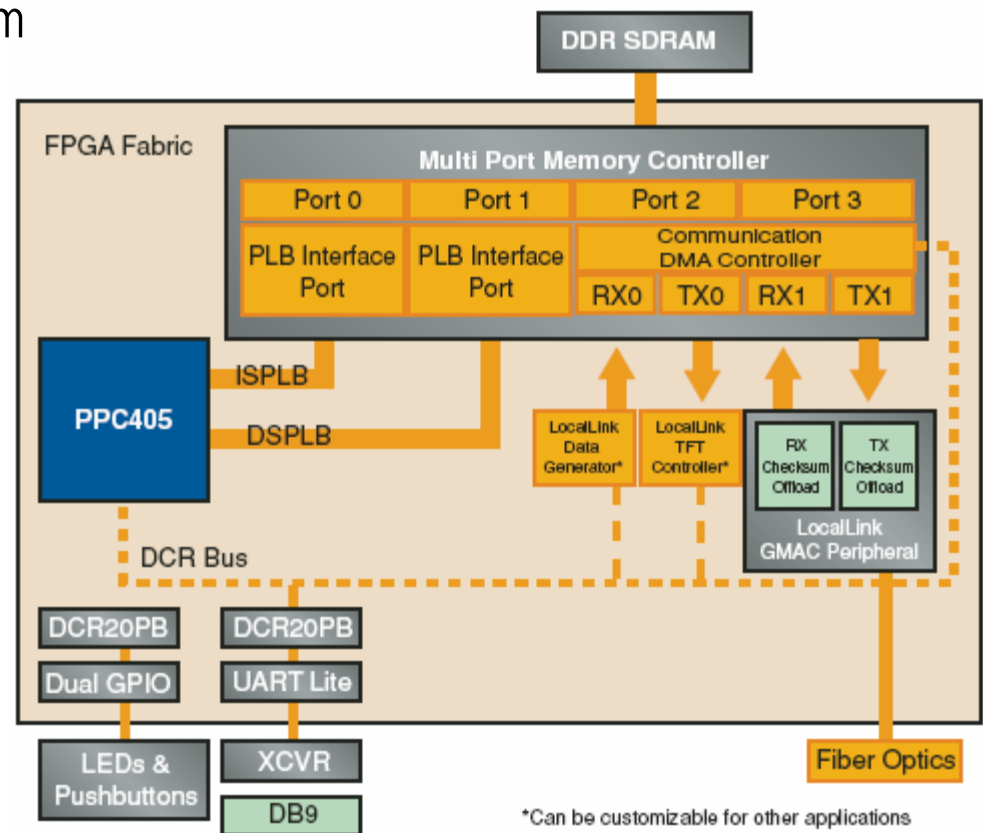
- Processor does not 'touch' payload data
 - TCP Zero-Copy software APIs
 - Transmit and Receive Checksum offload in FPGA fabric
 - DMA with Data Realignment Engine
- Multi Port Memory Controller
 - Efficiently shares DDR SDRAM memory resource between processor and DMA devices
 - Allows point-to-point connection between high bandwidth peripherals and memory – no shared bus

Getting started with GSRD: www.xilinx.com/gsr (XAPP536)



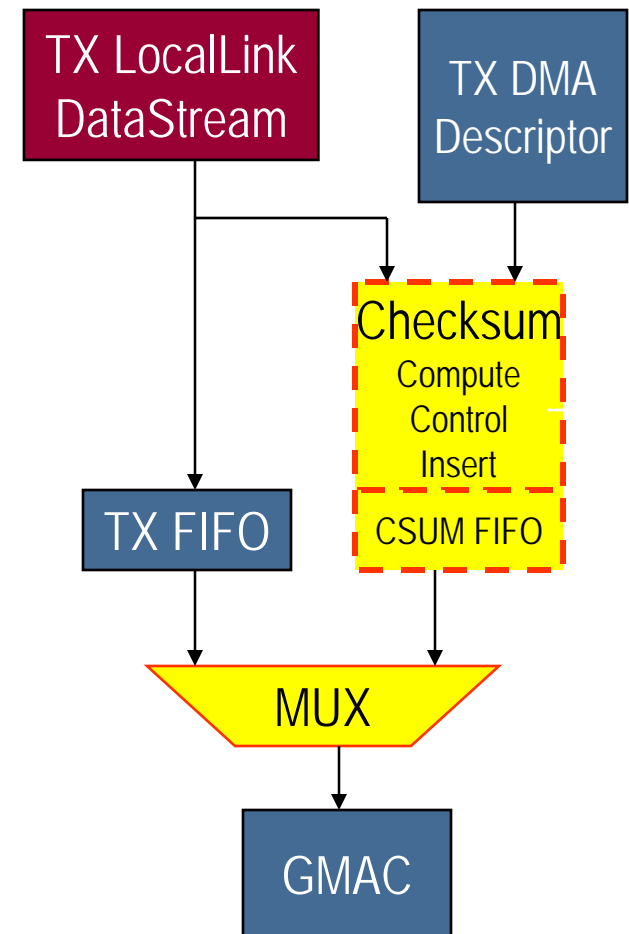
Gigabit System Reference Design

- GSRD is an EDK-based reference system tuned for maximum TCP/IP performance
- GSRD uses custom IP blocks
 - LocalLink Gigabit Ethernet MAC (LLGMAC, XAPP536)
 - Communications DMA Controller (CDMAC, XAPP535)
 - Multi Port Memory Controller (MPMC, XAPP535)
- TCP transmit rates as high as 780 Mbps
 - Ideal for applications requiring high performance bridging between TCP/IP and user data interfaces

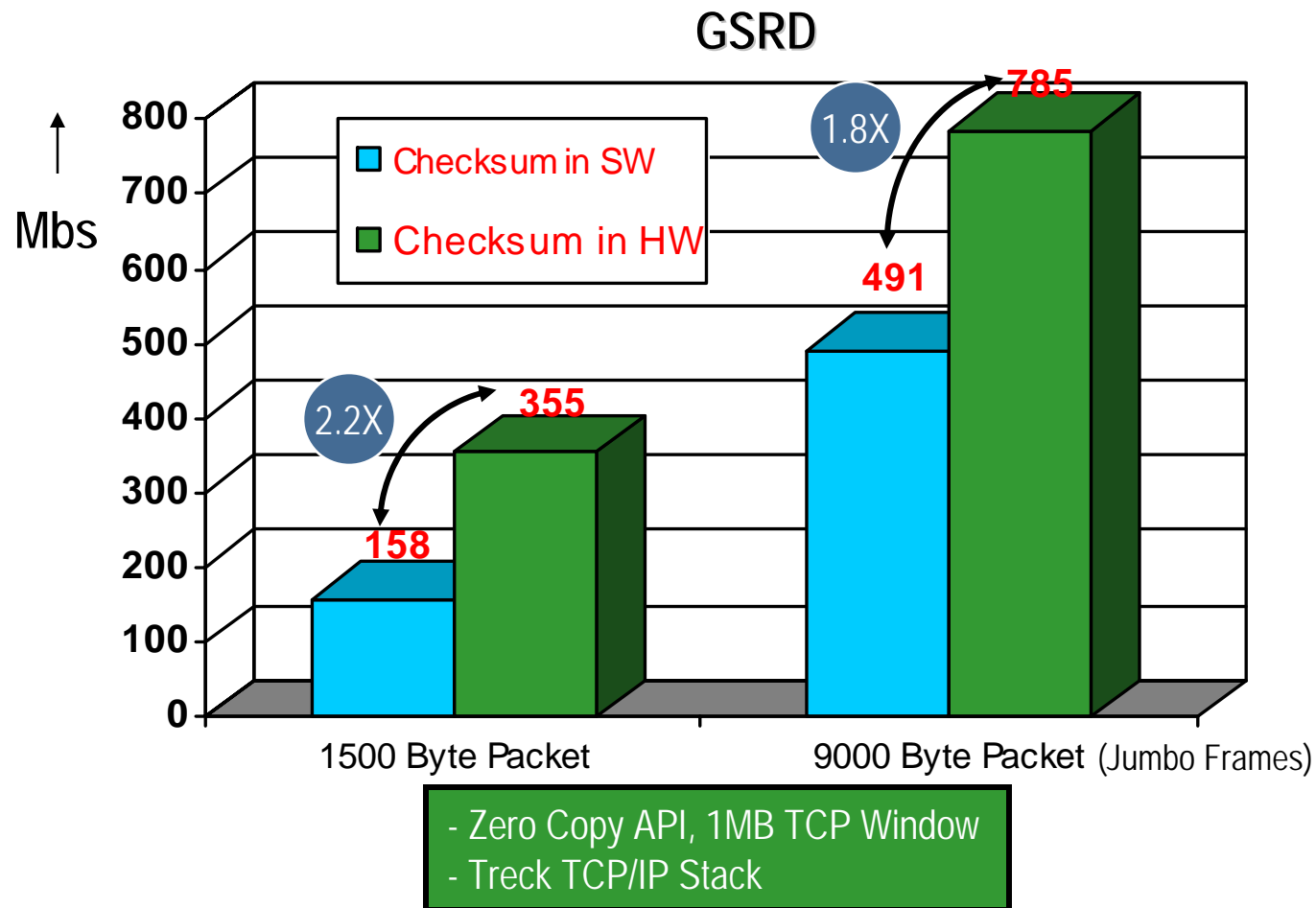


Checksum Offload Implementation

- Processor incurs severe penalty for checksum computation
 - Ethernet packet ingress to memory
 - PPC extraction of header from memory
 - Processing of header with software and data structures from memory
 - Processed packet assembled in memory
 - Egress of packet to the host
- Functionality can easily be offloaded into FPGA fabric
 - Simple state machine implements and inserts the checksum logic
 - Implemented in ~ 100 LUT's
- Software stack needs minor corresponding change
 - Many TCP/IP stacks have pre-built support for checksum offload



Transmit Performance Comparison With Checksum Offload



TCP Termination Challenges Summary

- Memory Bandwidth is important
 - High bandwidth peripherals such as 1 Gigabit Ethernet
 - Processor(s) instruction/data fetches
- PPC can't touch payload data bytes
- TCP Protocol Processing not to blame for low performance
 - Supporting functions like buffer copies, checksums, etc. are to blame

Overview

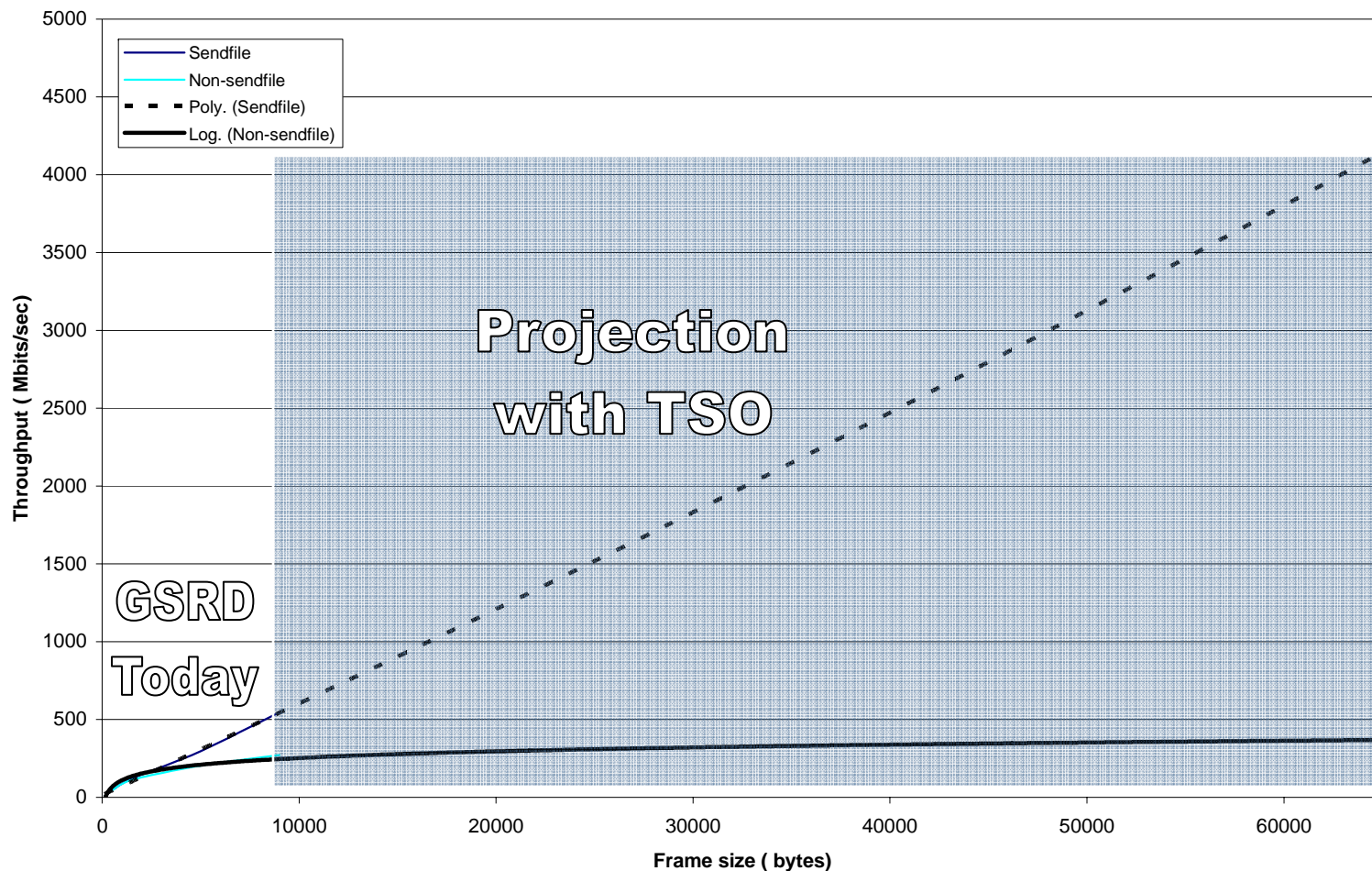
- Introduction to TCP/IP Protocol Suite
- Embedded TCP/IP Applications
- TCP Termination Challenges
- **TCP Acceleration Techniques**

Improving TCP TX Performance

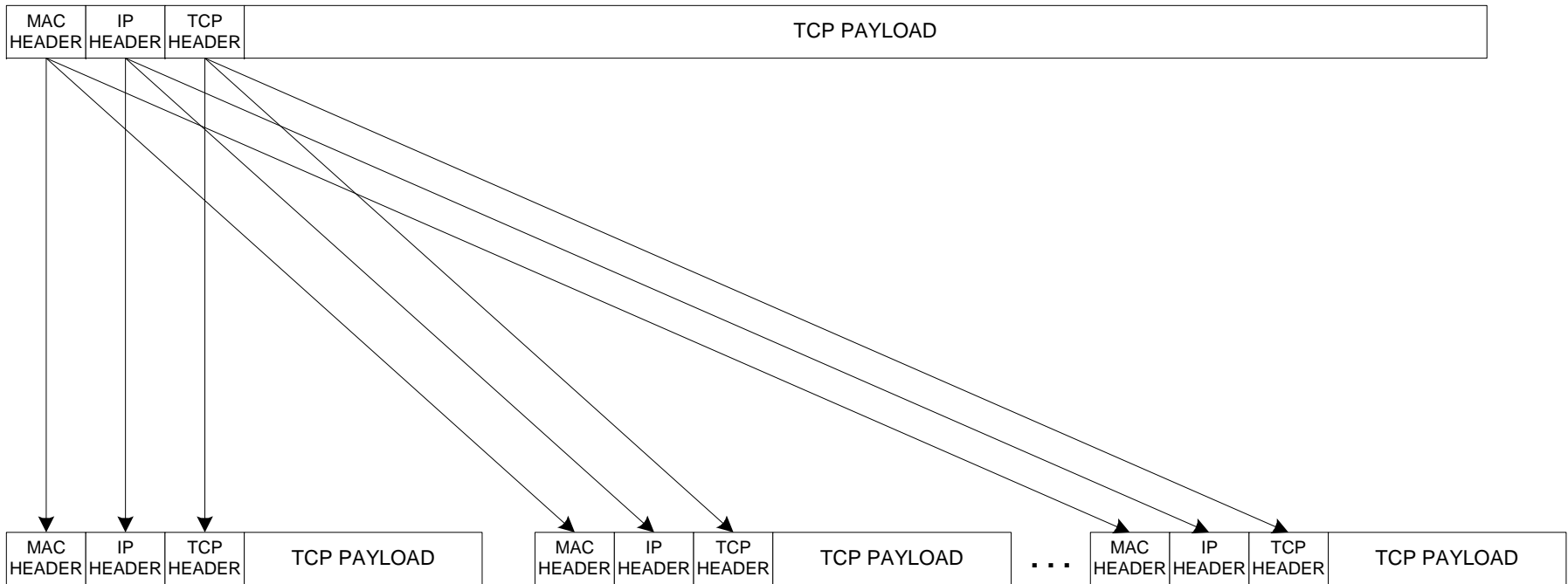
- Reduce occurrence of per packet Overhead
 - Decrease number of CPU generated packets
- TCP Segmentation Offload (TSO)
 - CPU generates large TCP frames (64 KB)
 - Hardware segments into smaller frames (1.5 KB)
 - LLGMAC optionally coalesces received ACKs

TX Performance Projections

Linux Netperf Transmit Performance (Sendfile vs Non-sendfile)



TSO Algorithm (1)



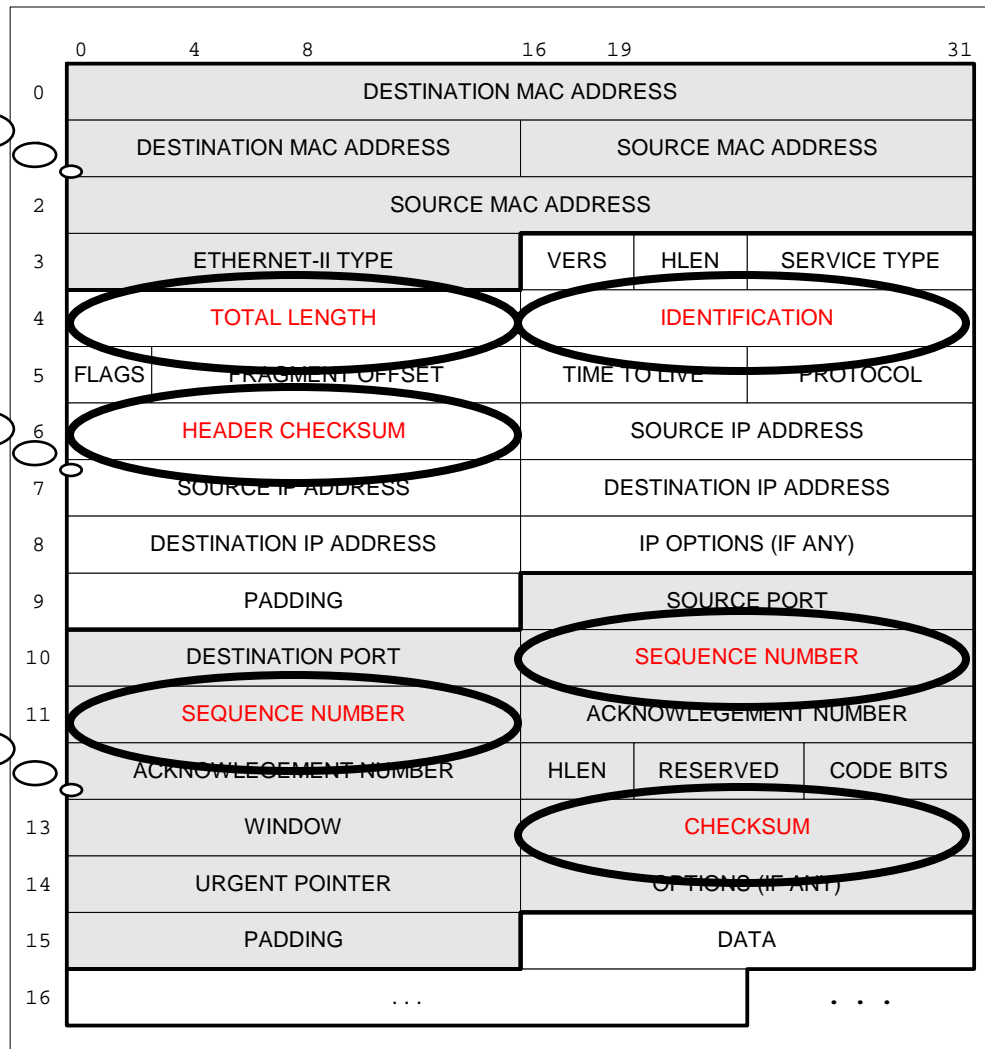
- LLGMAC uses TCP/IP frame header as template
- LLGMAC generates multiple standard sized frames

Template TCP/IP Header

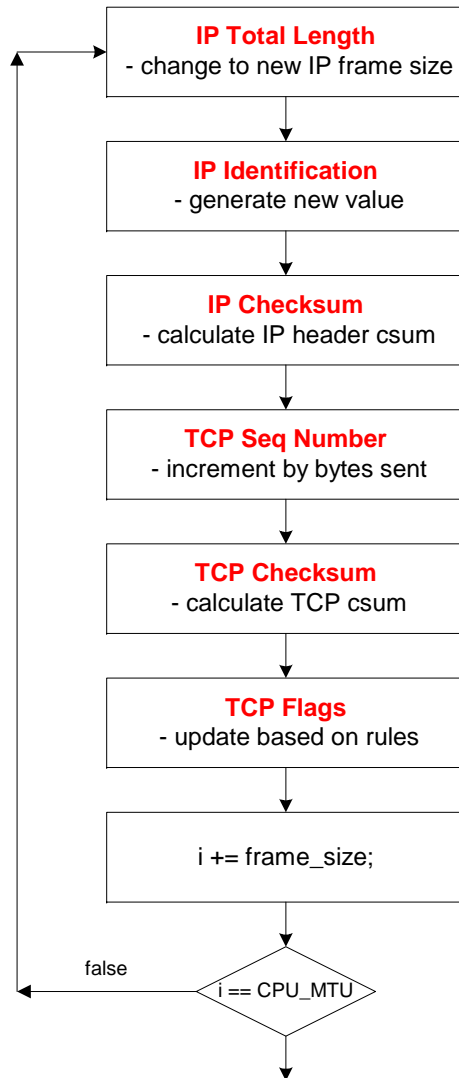
MAC Header

IP Header

TCP Header

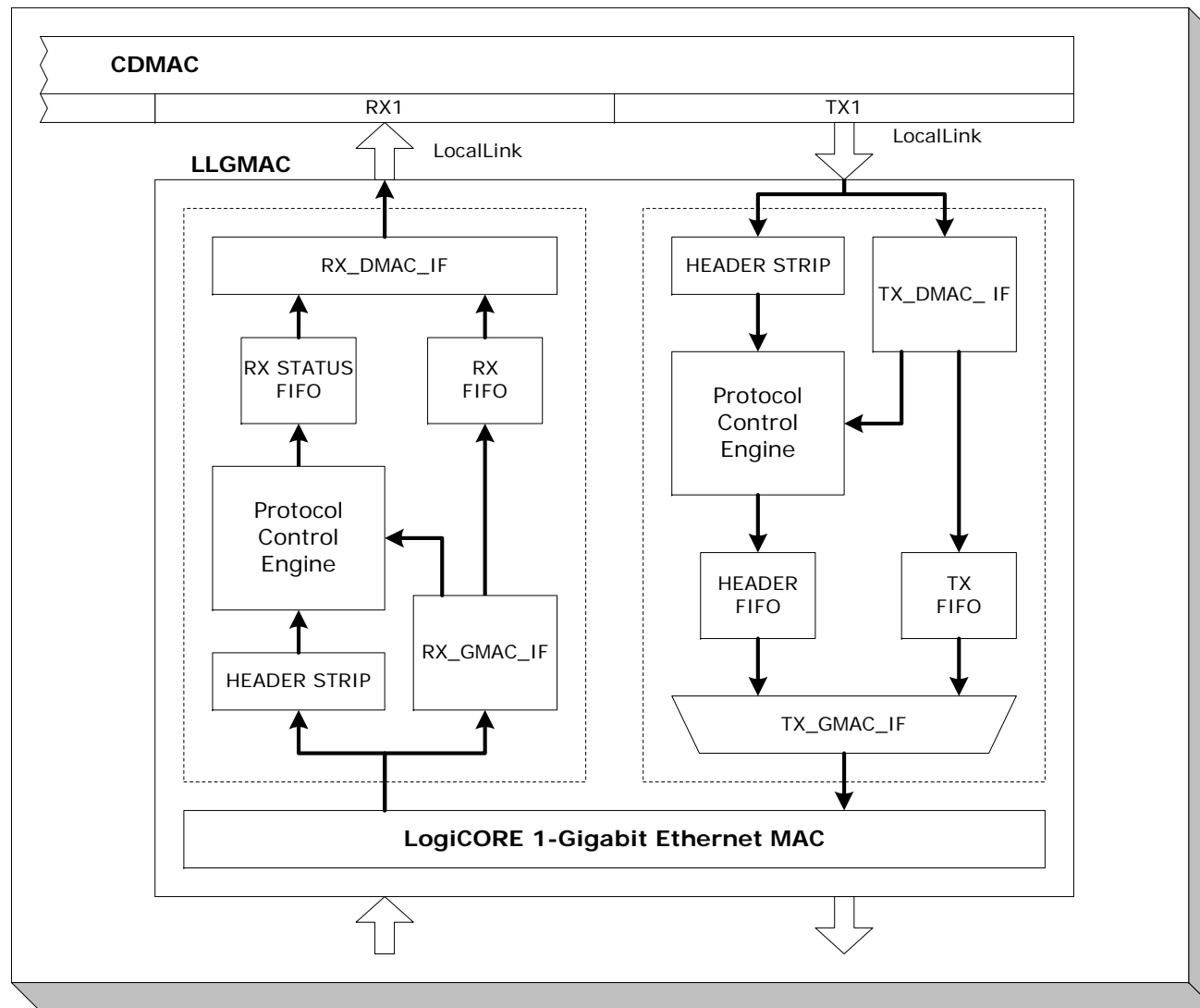


TSO Algorithm (2)



- Each generated frame
 - 200 cycles of processing of assembly code
 - 2uS at 100MHz
- FPGA Implementation
 - FSM controlled data path
 - OR -
 - Microprocessor in fabric
- IEEE Paper
 - Deferred Segmentation for Wire-Speed Transmission of Large TCP Frames over Standard GbE networks

TCP Acceleration Platform



Appendix

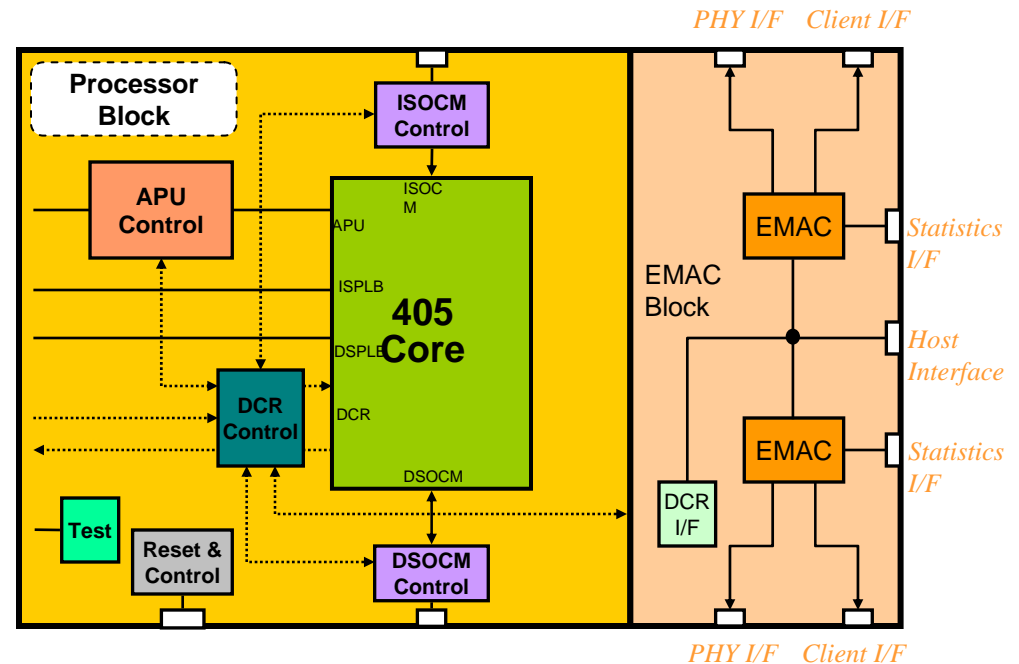
Glossary Of Terms

- XPS: Xilinx Platform Studio. SW to design with Xilinx processors
- EDK: Bundling on XPS with Processor IP
- lwip: Light Weight Internet Protocol TCP/IP Stack
- IP: Intellectual Property, pre engineered blocks of logic implementing specific functions
- EMAC: Ethernet MAC
- GEMAC: Gigabit Ethernet MAC
- TEMAC: Tri-Mode Ethernet MAC
- GSRD Gigabit Ethernet Reference Design
- Protocol: A formal specification of how things should communicate. Think of it as the language used for networks to communicate with each other.
- OSI: Open Systems Interconnection. The standard model to describe how computer networks should work
- Transmission Control Protocol/Internet Protocol (TCP/IP) is a group or suite of protocols that is widely used in networking. The Internet uses TCP/IP.
- TCP Stack: Library of functions that implement various layers of the communication protocol in Software

Virtex-4 Tri-Mode EMAC (TEMAC)

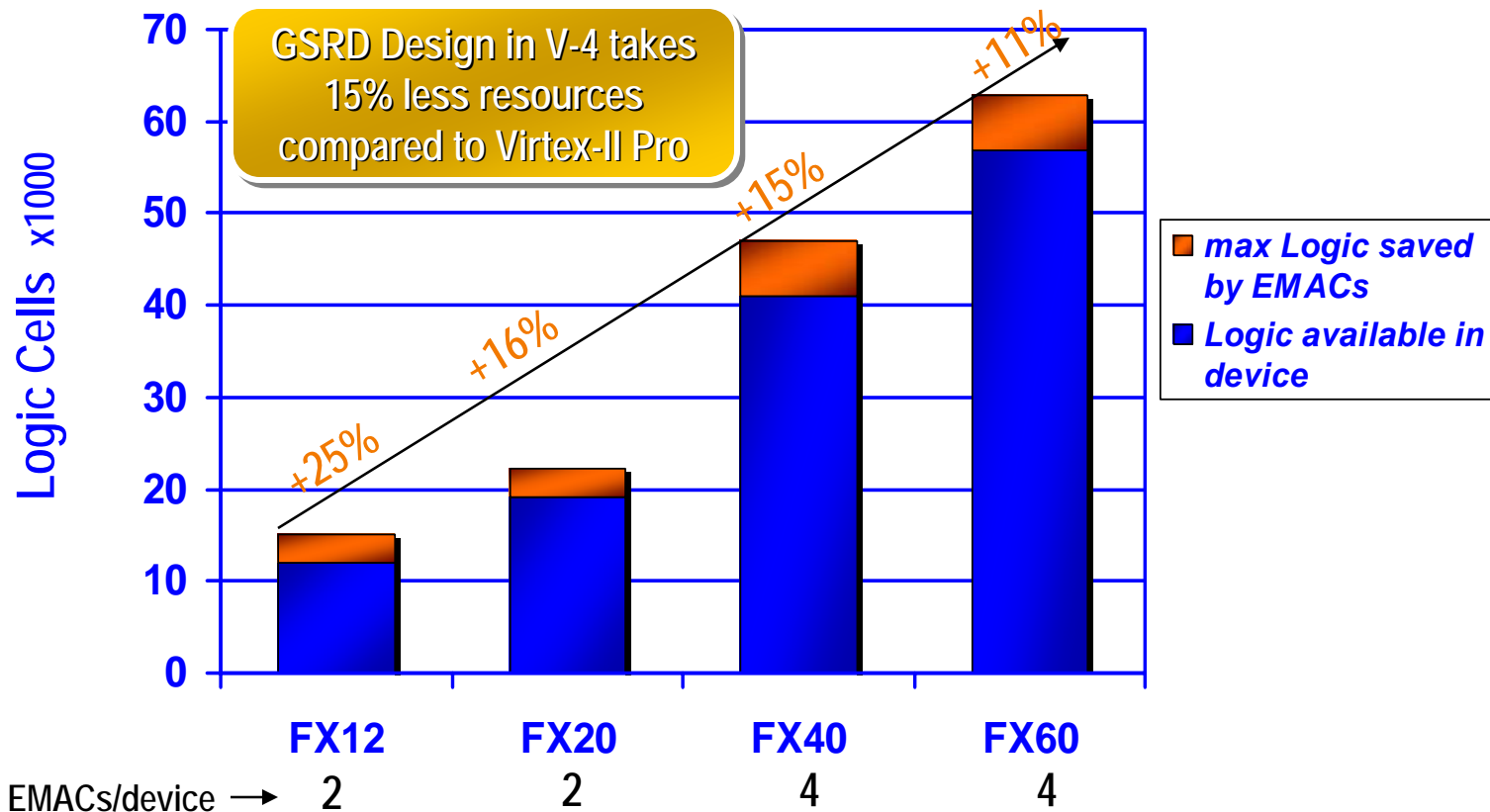
Virtex4-FX devices include 2 or 4 dedicated TEMACs

- 10/100/1000 Mb/s with auto-detect
- Can be used standalone or with embedded PPC405
- Seamless connection to MGT
- TX/RX Statistics gathering
- Jumbo Frame Support
- Processor can control the EMAC via
 - Device Control Register*
 - A memory mapped host interface
- UNH compliance tested
- Accessible through EDK Version 6.3i SP1



*DCR is part of IBM's CoreConnect Bus

Hard EMAC Frees Up Logic Resources In FPGA



** Comparison based on 1G PLB EMAC implementation; Higher savings realized when TEMAC features like Tri-speed, RGMII, SGMII, address match, multicast address table lookup are used