FPGA Communications Based on Gigabit Ethernet

Carlos Serrano & Larry Doolittle

Lawrence Berkeley National Laboratory

ICALEPCS, Grenoble, October 11, 2011

BERKELEY LAB
Lawrence Berkeley National Laboratory

Index

Overview

The problem

Establish communication between commodity computers and highly specialized hardware via EPICS, TANGO, etc.

FPGAs in Accelerators

FPGAs add high performance and flexibility to traditional accelerator instrumentation.

How can specialized HW meet commodity computer world?

GbE keeping highly specialized hardware specialized.

GbE in FPGAs: The Result of Two Success Stories

Field Programmable Gate Arrays

First commercially viable FPGA invented by the Xilinx co-founders in 1985. First time a piece of hardware could have programmable gates and programmable interconnects. Started at 64 gates, to the millions of current technologies accounting for a $2.75 billion market in 2010.

Ethernet

Developed at Xerox Park in 1973-74 to interconnect computers locally inside the company. Standardized in 1980 as IEEE 802.3, and accounts for a $16.3 billion market in 2010 for switches alone.

FPGAs in Accelerators

Architecture designed on-demand

FPGA boards can be quite generic and still leave room for a custom, application-optimized hardware design in the FPGA.

FPGAs in Accelerators

Applications

Fast feedback, timing, high speed communications, data acquisition, etc.

Why FPGAs?

Flexible, reliable, low latency, large throughput, deterministic.

Why not computers?

FPGAs perform much better for DSP, real-time, low latency applications .

Hardware Communication Standards

Many options

PCI, USB, CAN, VME, VXI, cPCI, PCIe, IEEE-488 (HP-IB), IEEE-1392 (Firewire), SATA (or eSATA), etc.

Why Ethernet?

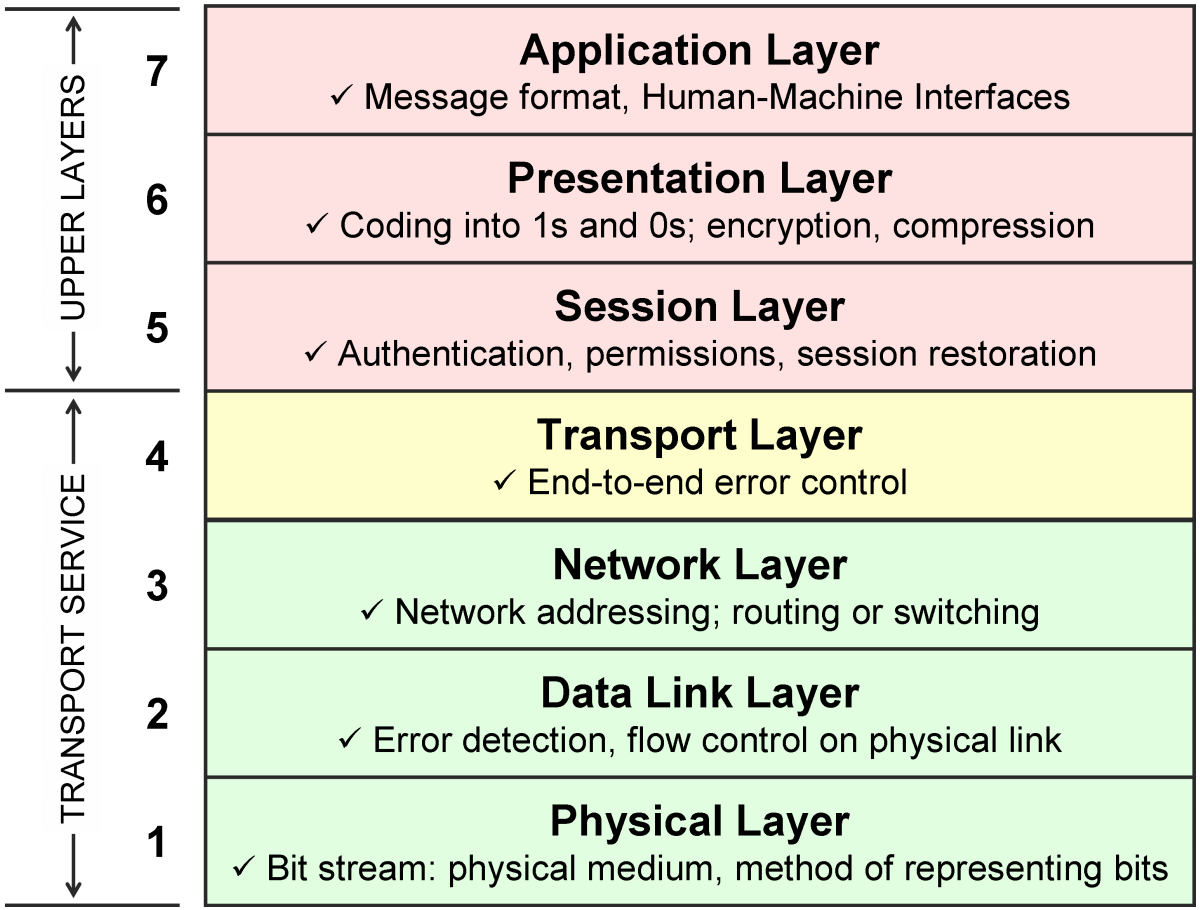
Simple enough for highly specialized hardware, well supported by commodity computers, does not seem it will leave us any time soon, and provides more bandwidth than computers can handle anyway.

Ethernet is simple

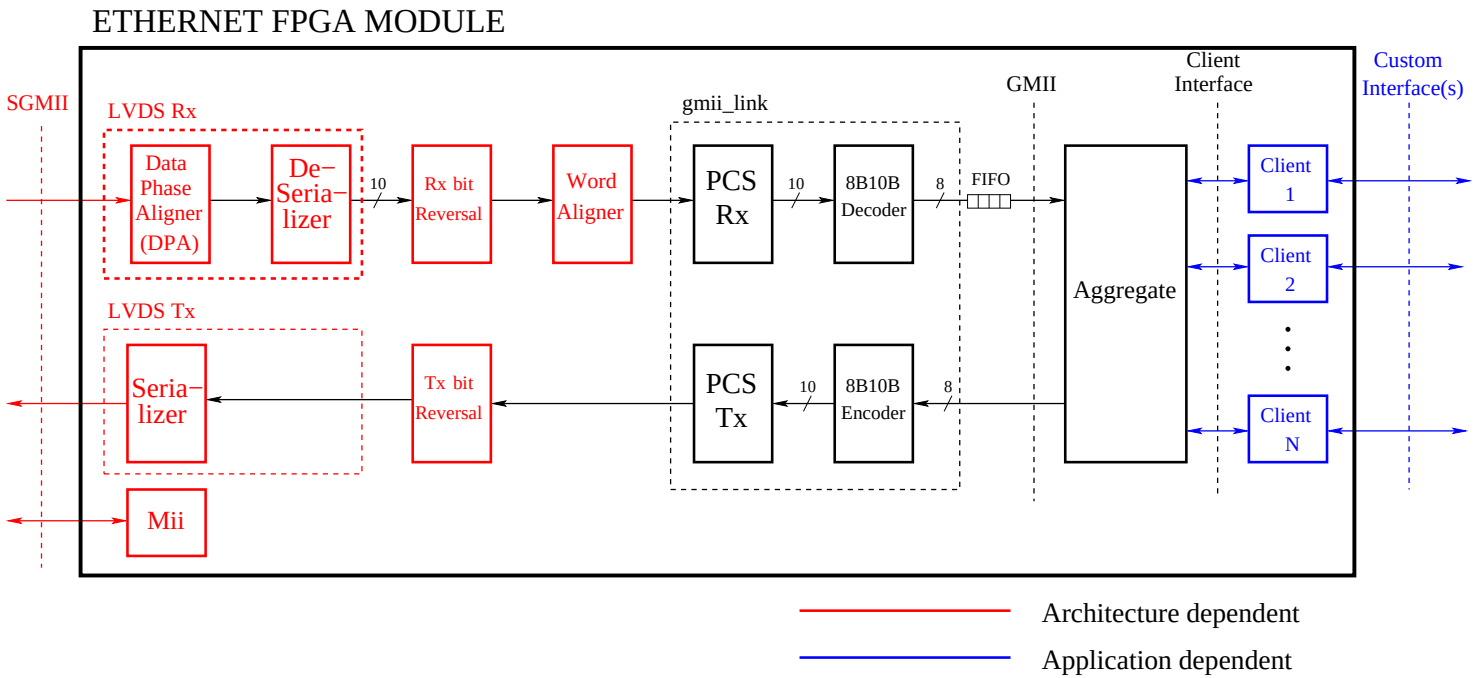
Why is it so successful?

The key is simplicity: Well thought out standard pushing complexity to upper layers. It does what it has to do, it does it well, and for cheap.

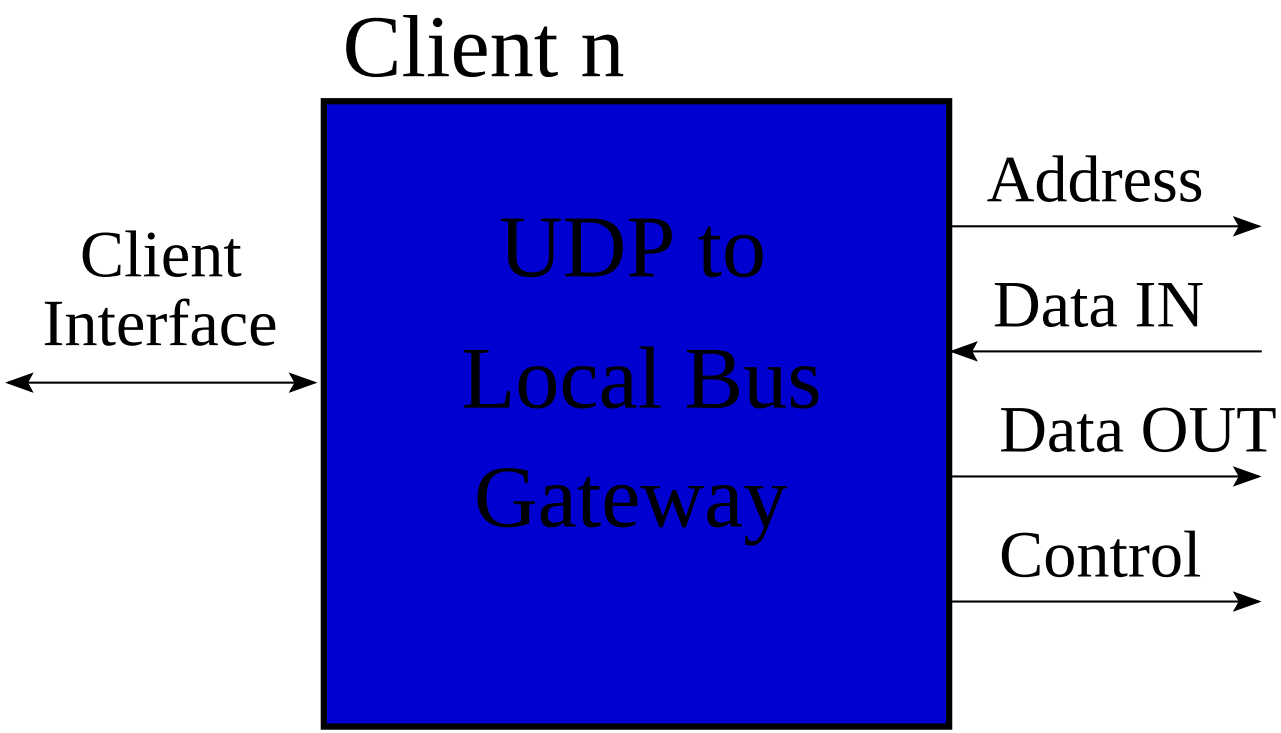
What does Ethernet do?



Implementation: 1GbE for Stratix-IV on Copper



Client Example: UDP to Local Bus Gateway



Local Bus Custom Frame Format

Local bus encoded 64−bit frame

CTL	ADDRESS	DATA
8	24	32

Conclusions

GbE and FPGAs mix well in nature

Both share simplicity and performance as high values.

Full bandwidth used

Real 1 GbE, where computer is the bottleneck.

Low FPGA resources used: Keep resources for applications!

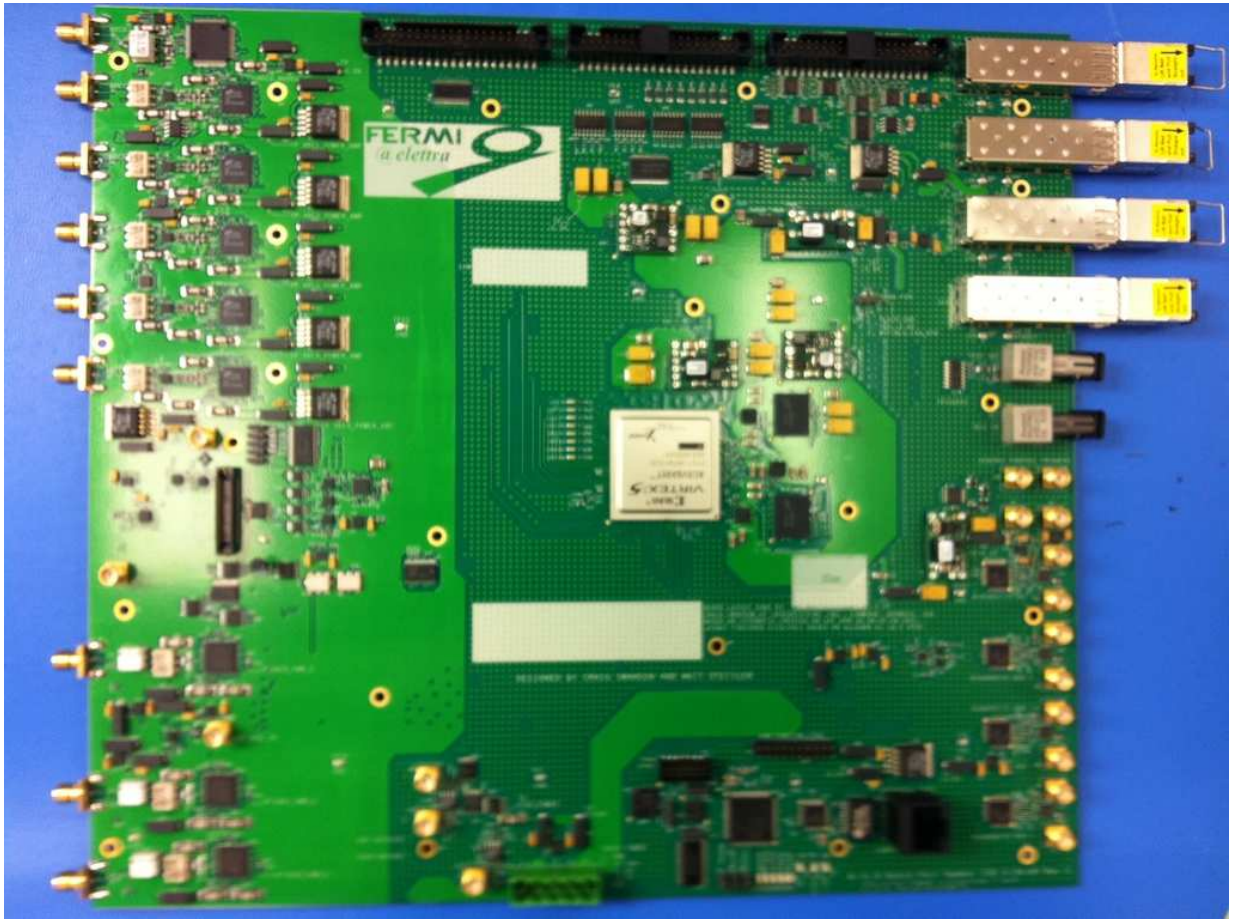
Uses 930 logic cells, and 3 block RAMs on a Xilinx Spartan-6 XC6SLX45T (1.7% and 2.6% of the available resources).

Flexible, modularized solution

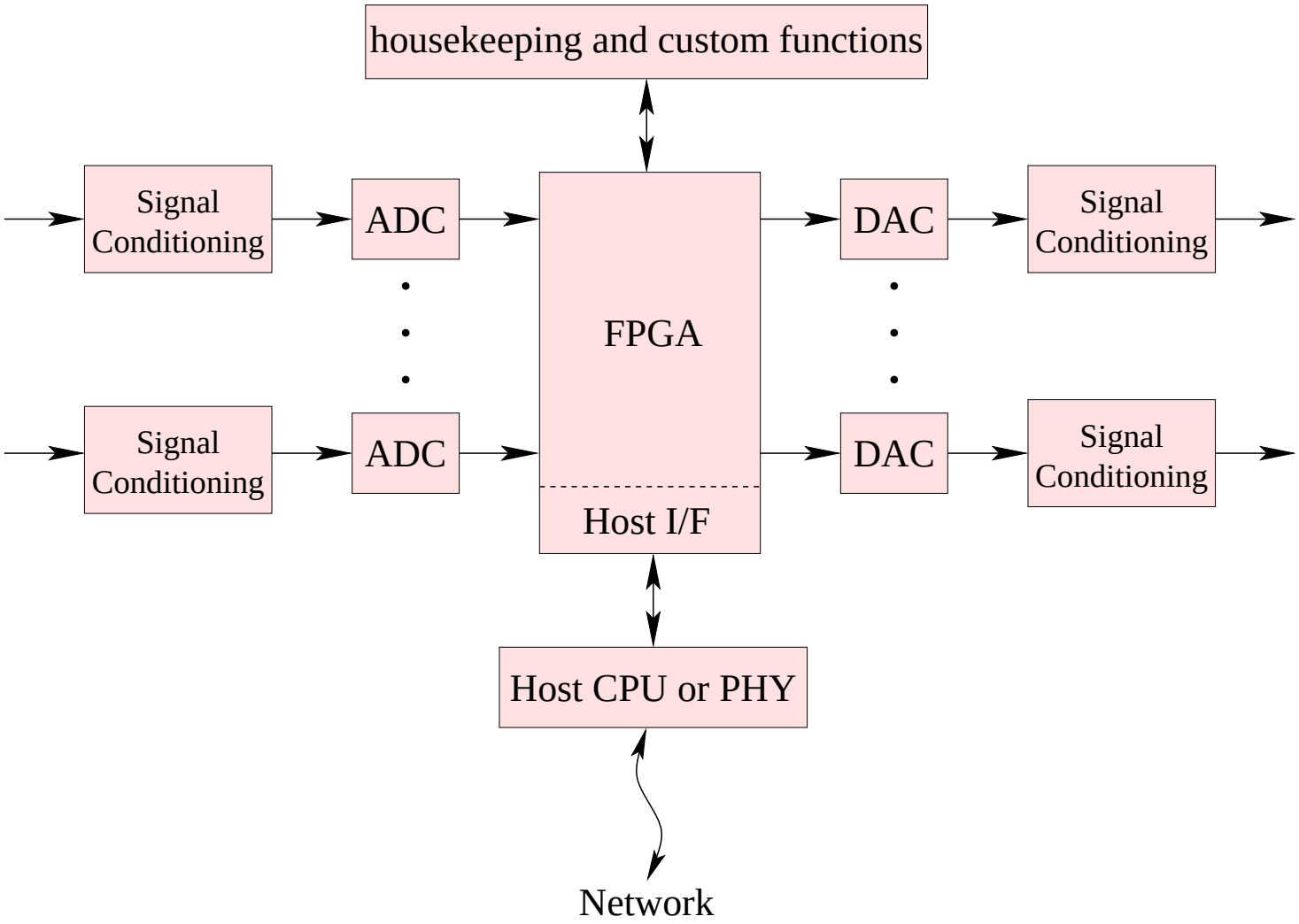
Several combinations of FPGA vendors and physical media supported, with room for upper layer customization.

Voilà..

Support Slides (I): FERMI@ELETTRA LLRF board



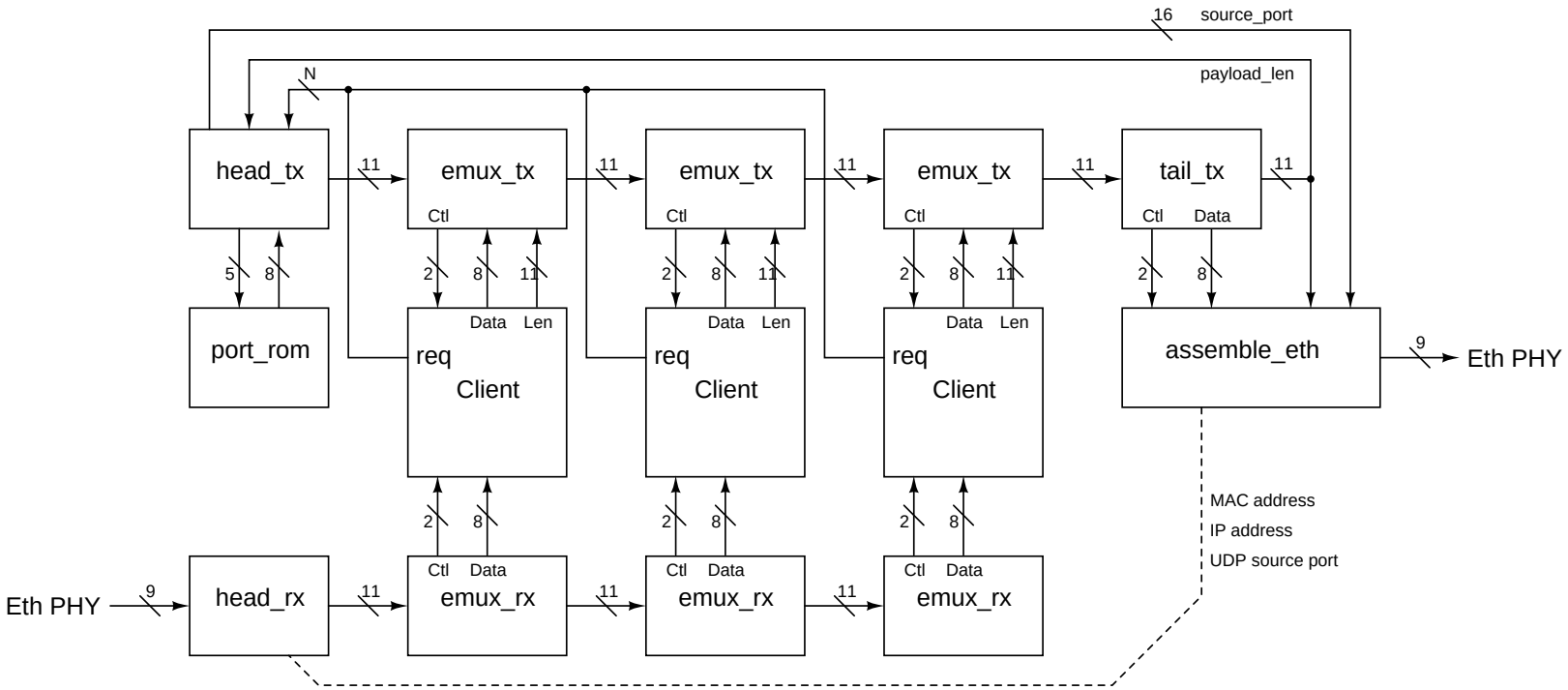
Support Slides (II): Familiar FPGA Block Diagram



Support slides (III): Ethernet module block diagram

Pseudo-Scalable Pseudo-Ethernet Pseudo-Switch

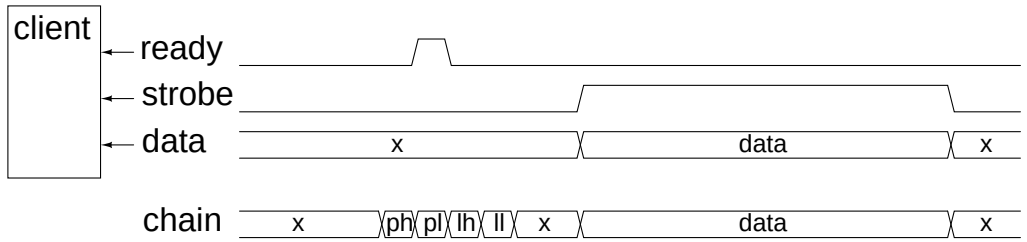
(Short-tick FPGA targeted, 2-16 clients, UDP payload, one to many and many to one switch)



Not shown: ARP, MII

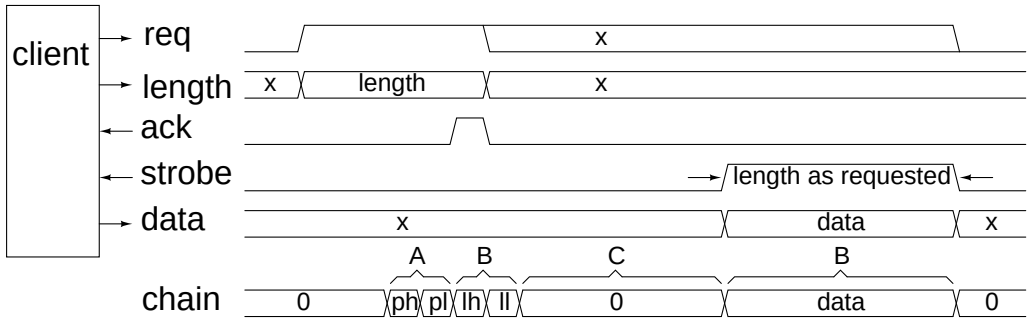
Support slides (IV): Client interface

Rx chain client interface



ph, pl: UDP port high and low octet
lh, ll: UDP datagram length high and low octet
(subtract 8 from datagram length to get number of payload bytes)

Tx chain client interface



A: sent down Tx chain by head_tx
B: inserted in Tx chain by emux_tx, based on control signals
C: time reserved for Ethernet header
ph, pl: UDP port high and low octet
lh, ll: Payload length high and low octet