UNIVERSITY OF CAMBRIDGE

P51: High Performance Networking

Introduction to NetFPGA

Dr Noa Zilberman
noa.zilberman@cl.cam.ac.uk

Lent 2018/19

Practical Assignment

Practical Assignment

- Goal: Design a high performance networked application
- Who: 2 students in each team
- What:
  - Start from a reference design (e.g. Reference Switch)
  - Pick an application
  - Implement in a network device and provide line-rate performance
- When:
  - Due date: First day of Easter term, 23/4/2019
    Via Moodle

Practical Assignment

- Submission contents:
  - Source code and bit files
  - Any scripts used for testing + outputs
  - Documentation
    - Architecture
    - Performance profiles
    - Design decisions
    - Evaluation plan and evaluation results
    - Full details will be provided later

Project Selection

- Project ideas appear on Moodle
- Pick an application – prefer a topic you are familiar with.
- Pick a platform:
  - NetFPGA – default, provided by the course's team
  - Otherwise – bring your own platform, subject to approval
    Access to the platform required for the assessment
- Pick a starting point – Choose an easy starting point!
  - Default – NetFPGA Reference Switch
  - Can be any network application
- Pick a design flow:
  - Verilog – stable, more support and existing code.
  - P4 – less hardware knowledge required, easier to implement.

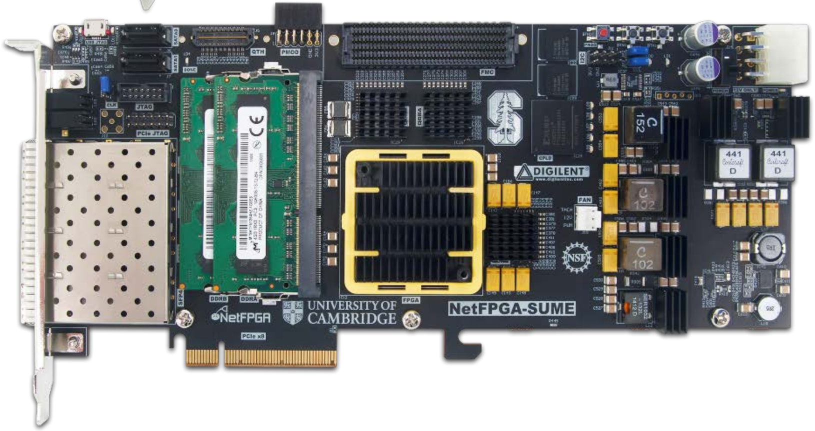
Project – For Next Week

- Choose your partner
- Choose your project
- Submit a project proposal via Moodle by 29/1/2019 16:00
  - Use the template on the course's website
  - Indicate if you are interested in publishing your work
- Projects will be discussed in the next lab (30/1/2019)
- The project proposal can be updated following 2:1 discussion (next week)

Section I: The NetFPGA platform

NetFPGA = Networked FPGA

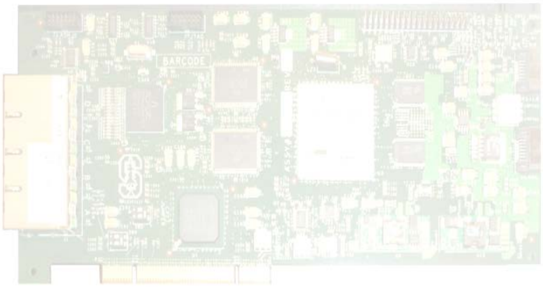
A line-rate, flexible, open networking platform for teaching and research



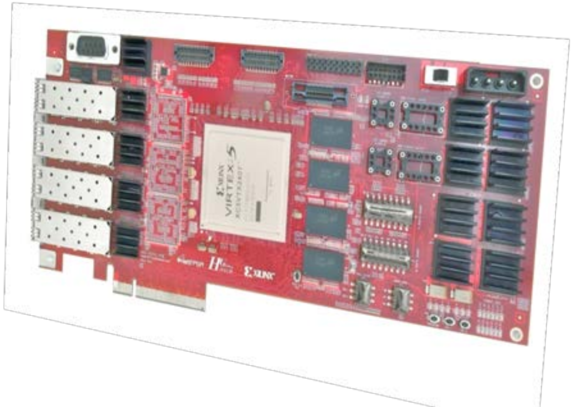
=

- Network Interface Card
- Hardware Accelerated Linux Router
- IPv4 Reference Router
- Traffic Generator
- Openflow Switch
- More Projects
- Add Your Project

NetFPGA Family of Boards

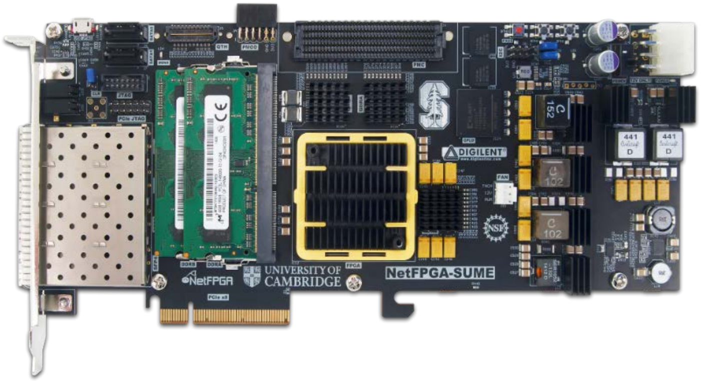


NetFPGA-1G (2006)



NetFPGA-10G (2010)

NetFPGA-1G-CML (2014)



NetFPGA SUME (2014)

NetFPGA consists of…

Four elements:

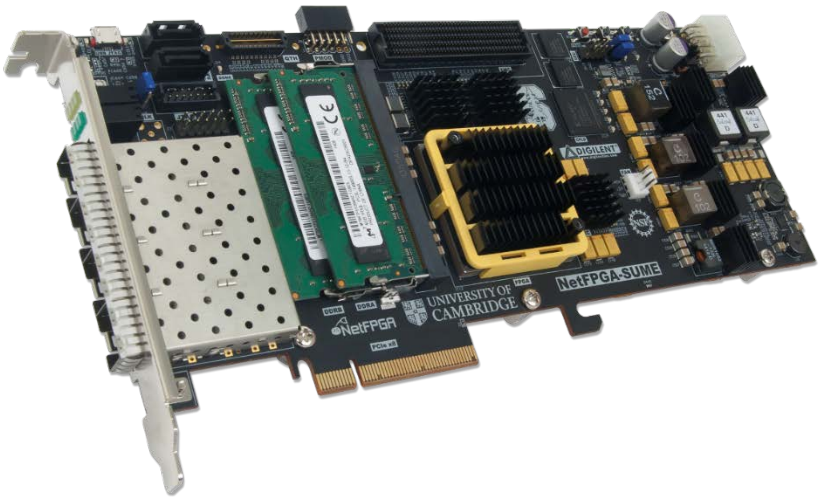
NetFPGA GitHub Organization

The Interwebs http://www.netfpga.org

- NetFPGA board

- Tools + reference designs

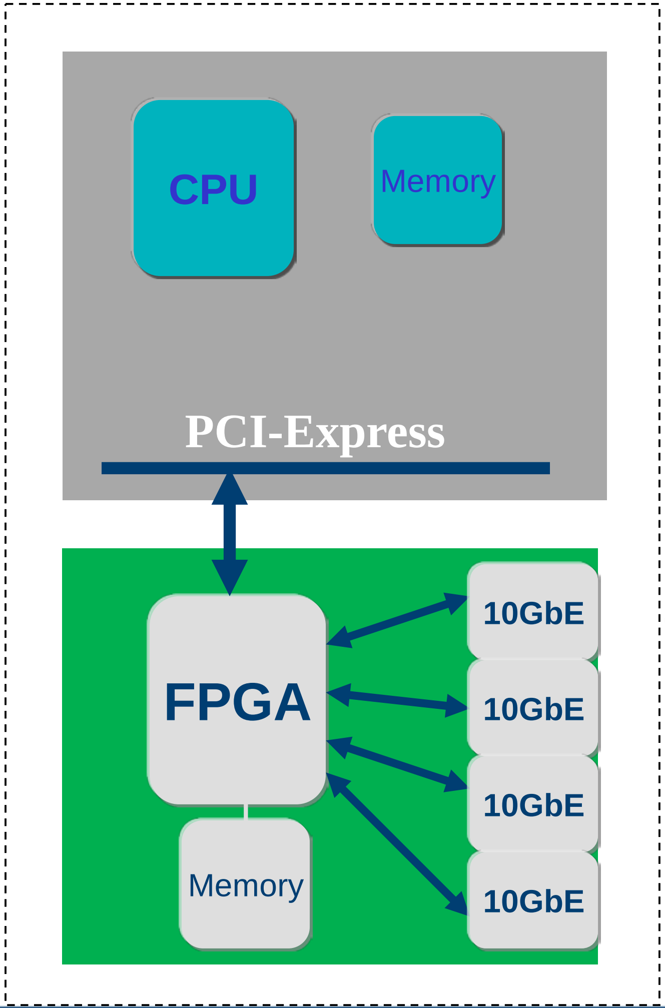
- Contributed projects



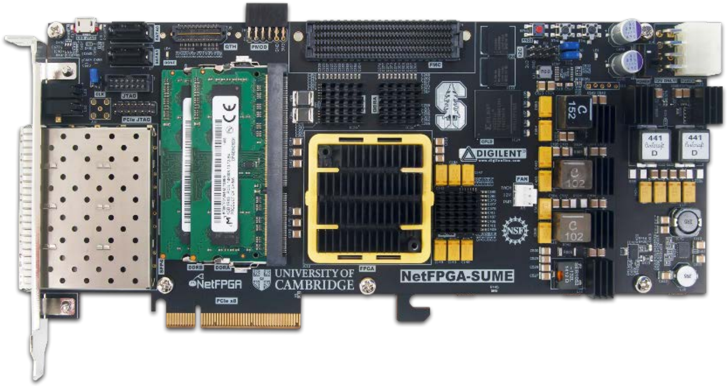
NetFPGA Board

Networking Software running on a standard PC

A hardware accelerator built with Field Programmable Gate Array driving 1/10/ 100Gb/s network links



PC with NetFPGA



Tools + Reference Designs

Tools:

- Compile designs
- Verify designs
- Interact with hardware

Reference designs:

- Router (HW)
- Switch (HW)
- Network Interface Card (HW)
- Router Kit (SW)
- SCONE (SW)

Community

Wiki

- Documentation
  - User’s Guide *“so you just got your first NetFPGA”*
  - Developer’s Guide *“so you want to build a …”*
- Encourage users to contribute

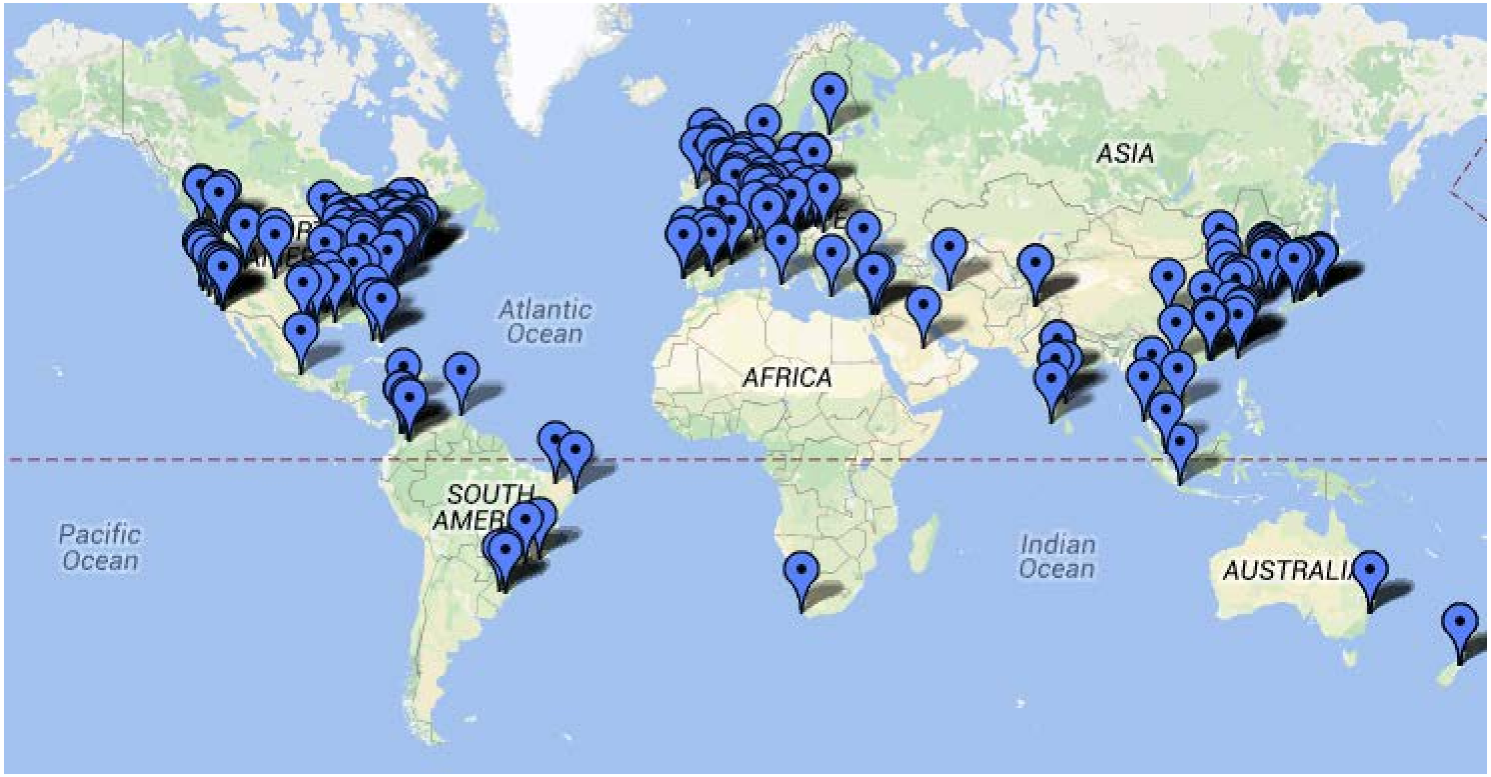
Mailing list

- Announcements
- Support by users for users

International Community

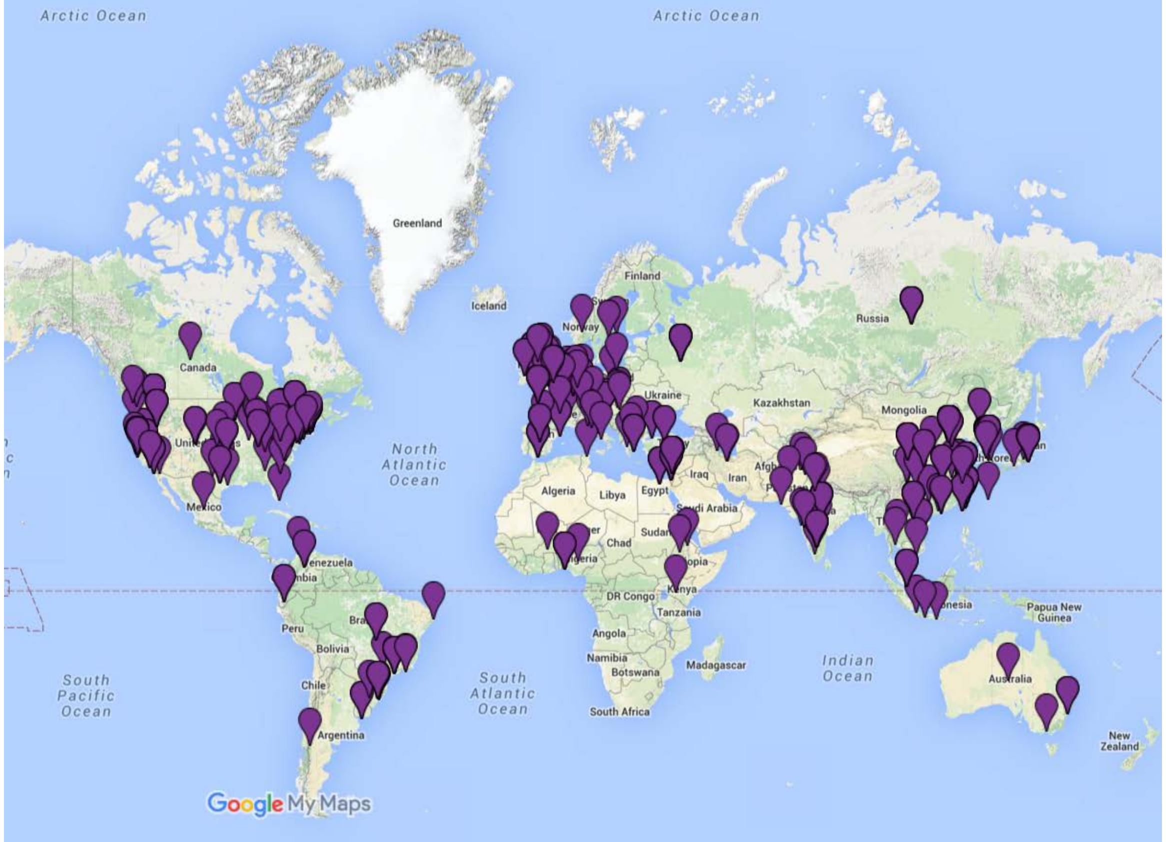
Over 1,200 users, using over 3500 cards at

200 universities in over 47 countries



NetFPGA SUME Community (*since Feb 2015*)

Over 600 users, 300 universities
in 60 countries and 6 continents



NetFPGA's Defining Characteristics

- Line-Rate
  - Processes back-to-back packets
    - Without dropping packets
    - At full rate
  - Operating on packet headers
    - For switching, routing, and firewall rules
  - And packet payloads
    - For content processing and intrusion prevention
- Open-source Hardware
  - Similar to open-source software
    - Full source code available
    - BSD-Style License for SUME, LGPL 2.1 for 10G
  - But harder, because
    - Hardware modules must meet timing
    - Verilog & VHDL Components have more complex interfaces
    - Hardware designers need high confidence in specification of modules

Test-Driven Design

- Regression tests
  - Have repeatable results
  - Define the supported features
  - Provide clear expectation on functionality

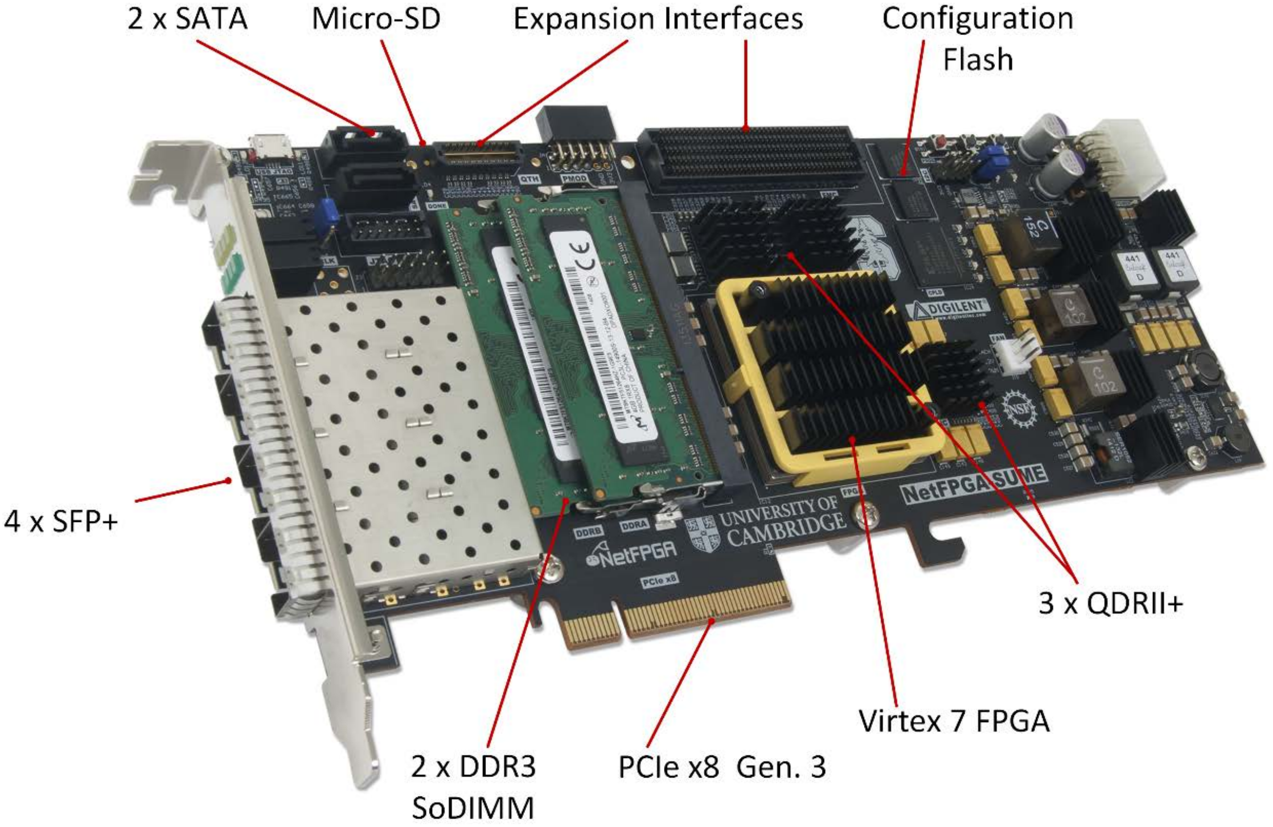
- *Example:* Internet Router
  - Drops packets with bad IP checksum
  - Performs Longest Prefix Matching on destination address
  - Forwards IPv4 packets of length 64-1500 bytes
  - Generates ICMP message for packets with TTL <= 1
  - Defines how to handle packets with IP options or non IPv4

  … and dozens more …

  Every feature is defined by a regression test

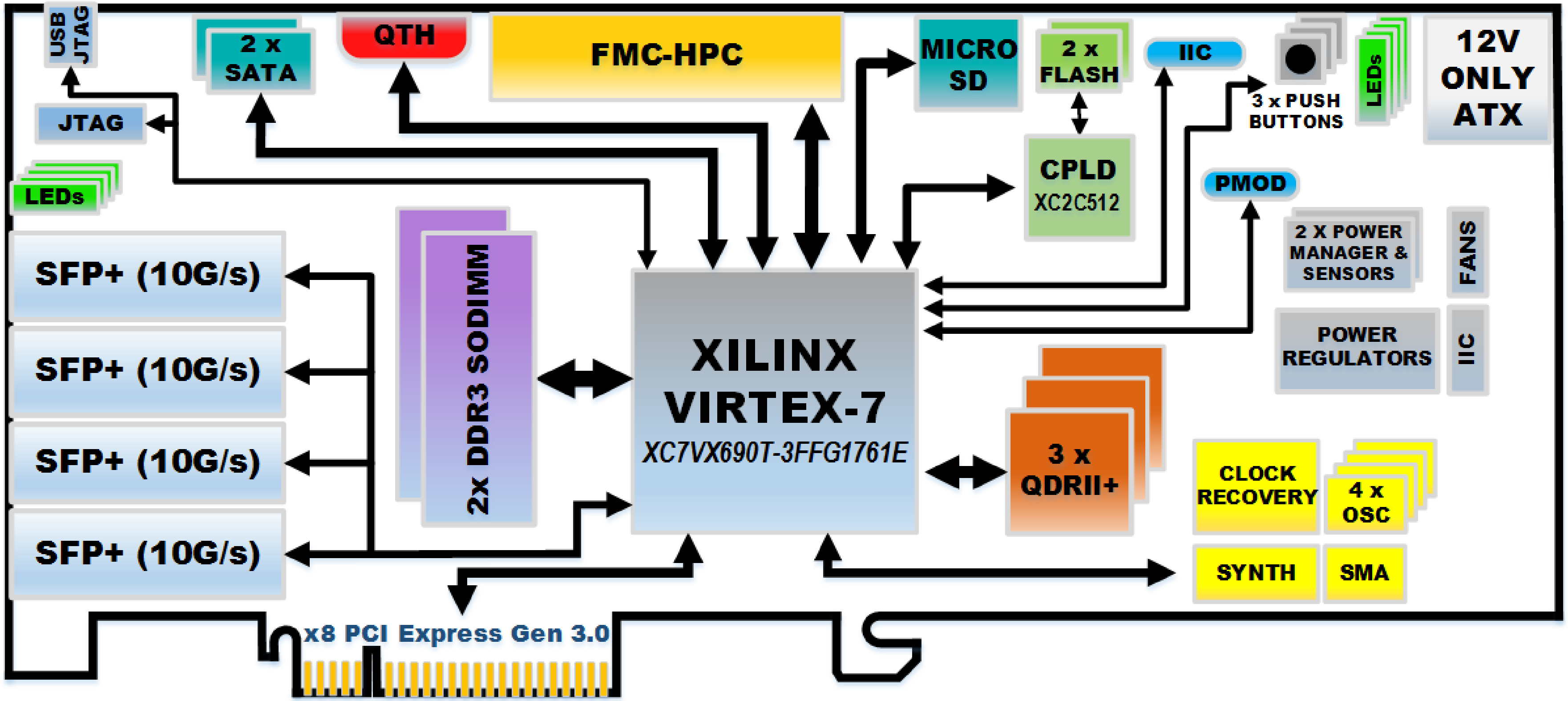
Section II: Hardware Overview

NetFPGA-SUME



NetFPGA-SUME

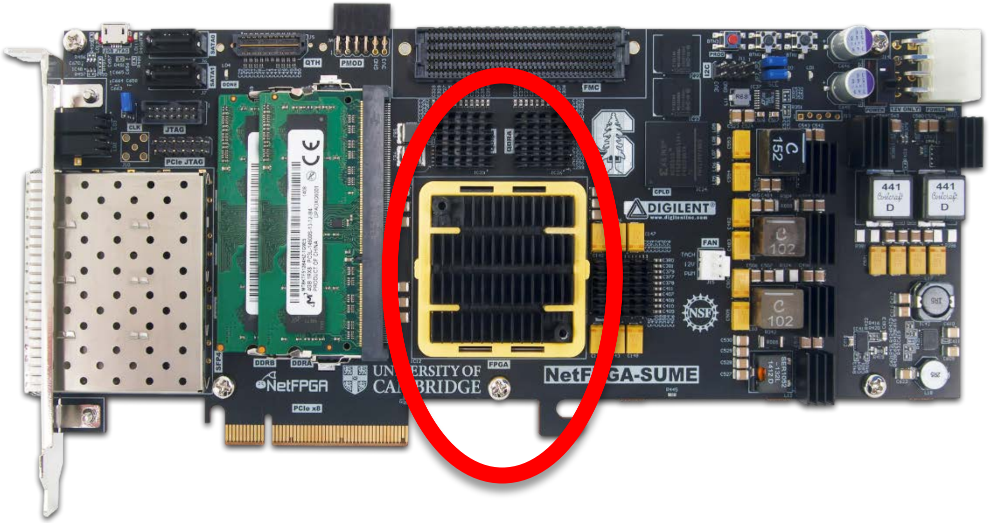
High Level Block Diagram



UNIVERSITY OF CAMBRIDGE

Xilinx Virtex 7 690T

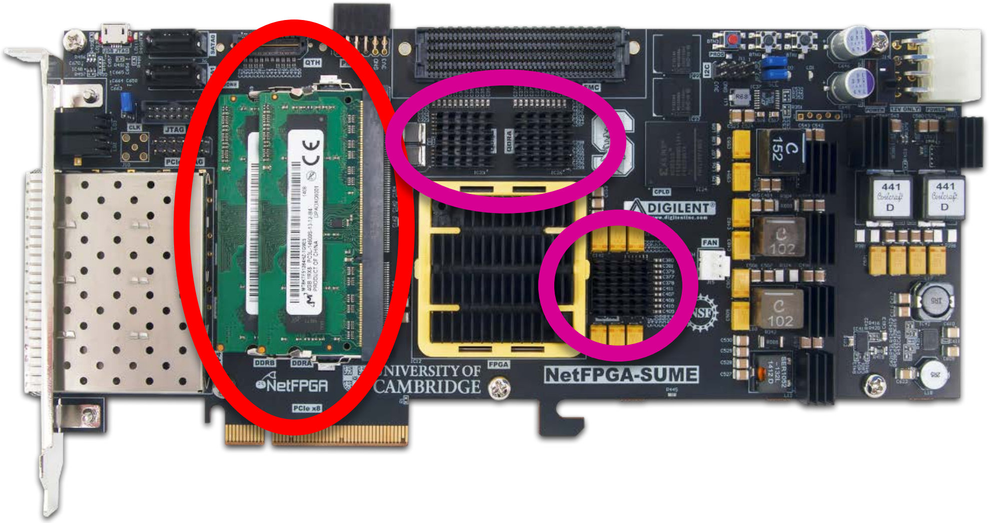
- Optimized for high-performance applications
- 690K Logic Cells
- 52Mb RAM
- 3 PCIe Gen. 3 Hard cores



Memory Interfaces

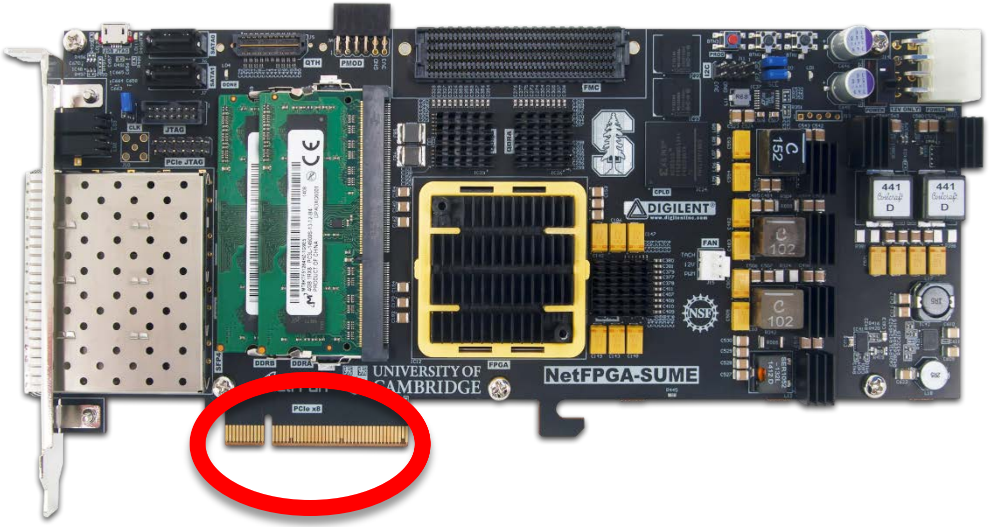
- DRAM:
  2 x DDR3 SoDIMM
  1866MT/s, 4GB

- SRAM:
  3 x 9MB QDRII+, 500MHz



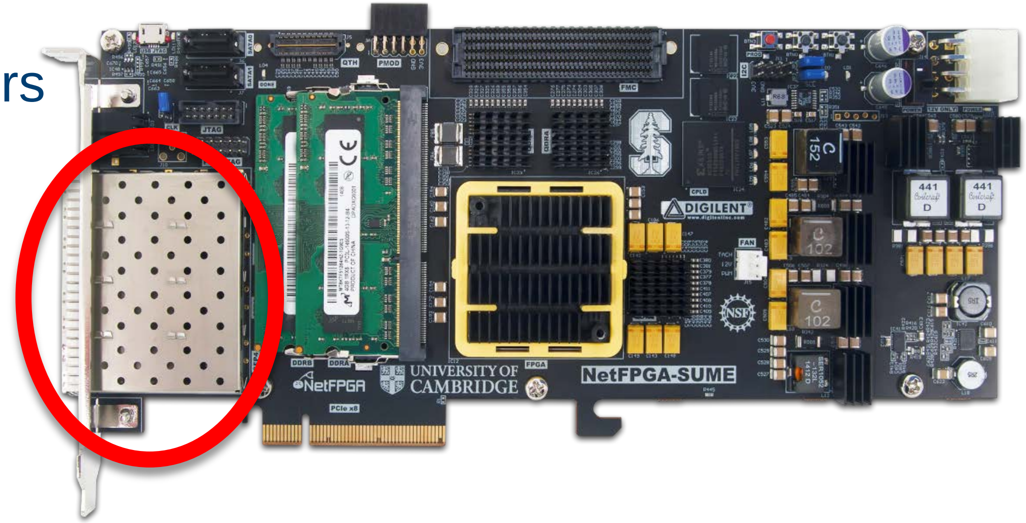
Host Interface

- PCIe Gen. 3
- x8 (only)
- Hardcore IP



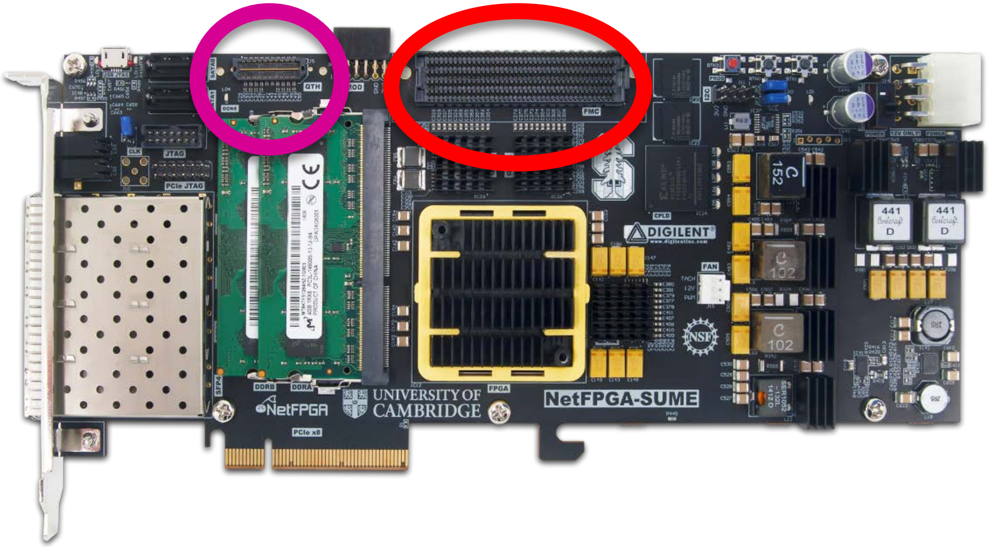
Front Panel Ports

- 4 SFP+ Cages
- Directly connected to the FPGA
- Supports 10GBase-R transceivers (default)
- Also Supports 1000Base-X transceivers and direct attach cables



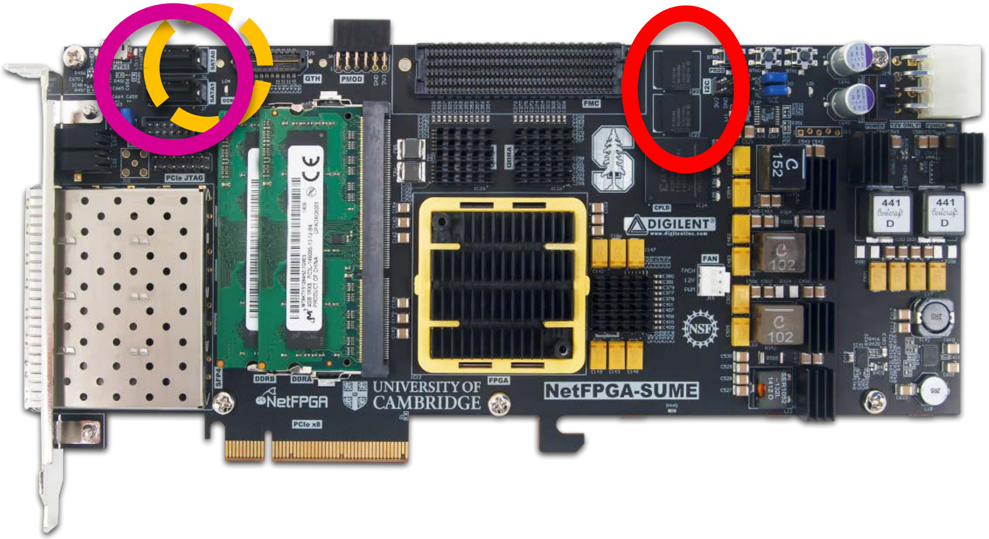
Expansion Interfaces

- FMC HPC connector
  - VITA-57 Standard
  - Supports Fabric Mezzanine Cards (FMC)
  - 10 x 12.5Gbps serial links
- QTH-DP
  - 8 x 12.5Gbps serial links

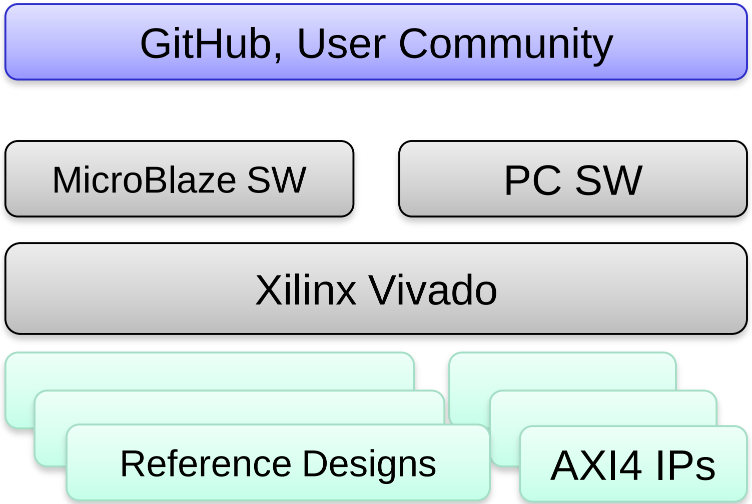


Storage

- 128MB FLASH
- 2 x SATA connectors
- Micro-SD slot
- Enable standalone operation



Beyond Hardware


GitHub, User Community

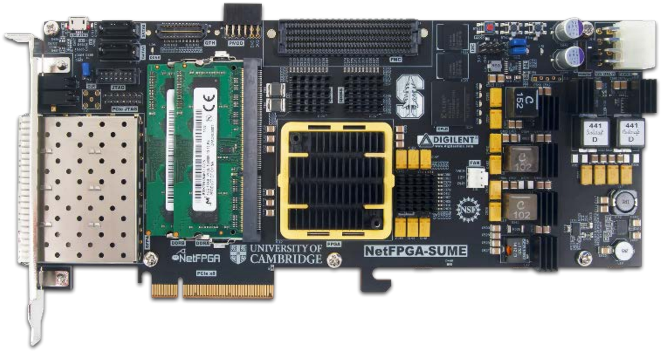
MicroBlaze SW

PC SW

Xilinx Vivado

Reference Designs

AXI4 IPs



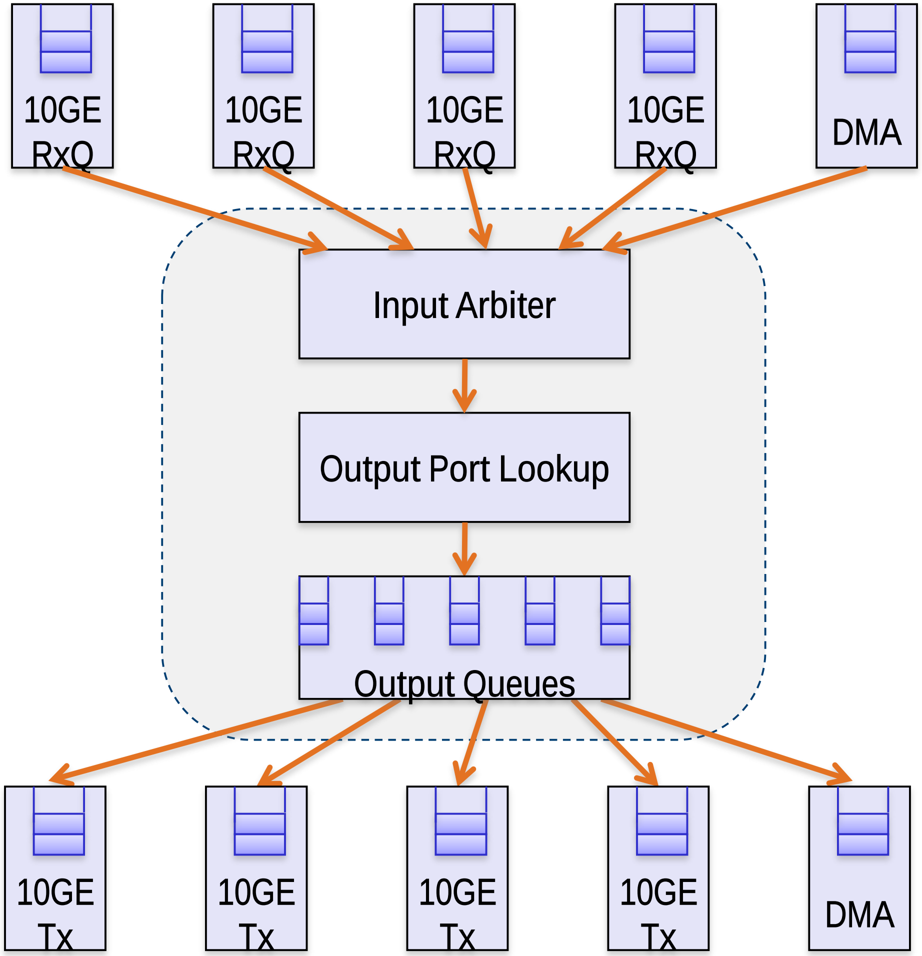
- NetFPGA Board
- Xilinx Vivado based IDE
- Reference designs using AXI4
- Software (embedded and PC)
- Public Repository
- Public Wiki

Section III: Life of a Packet

Reference Switch Pipeline

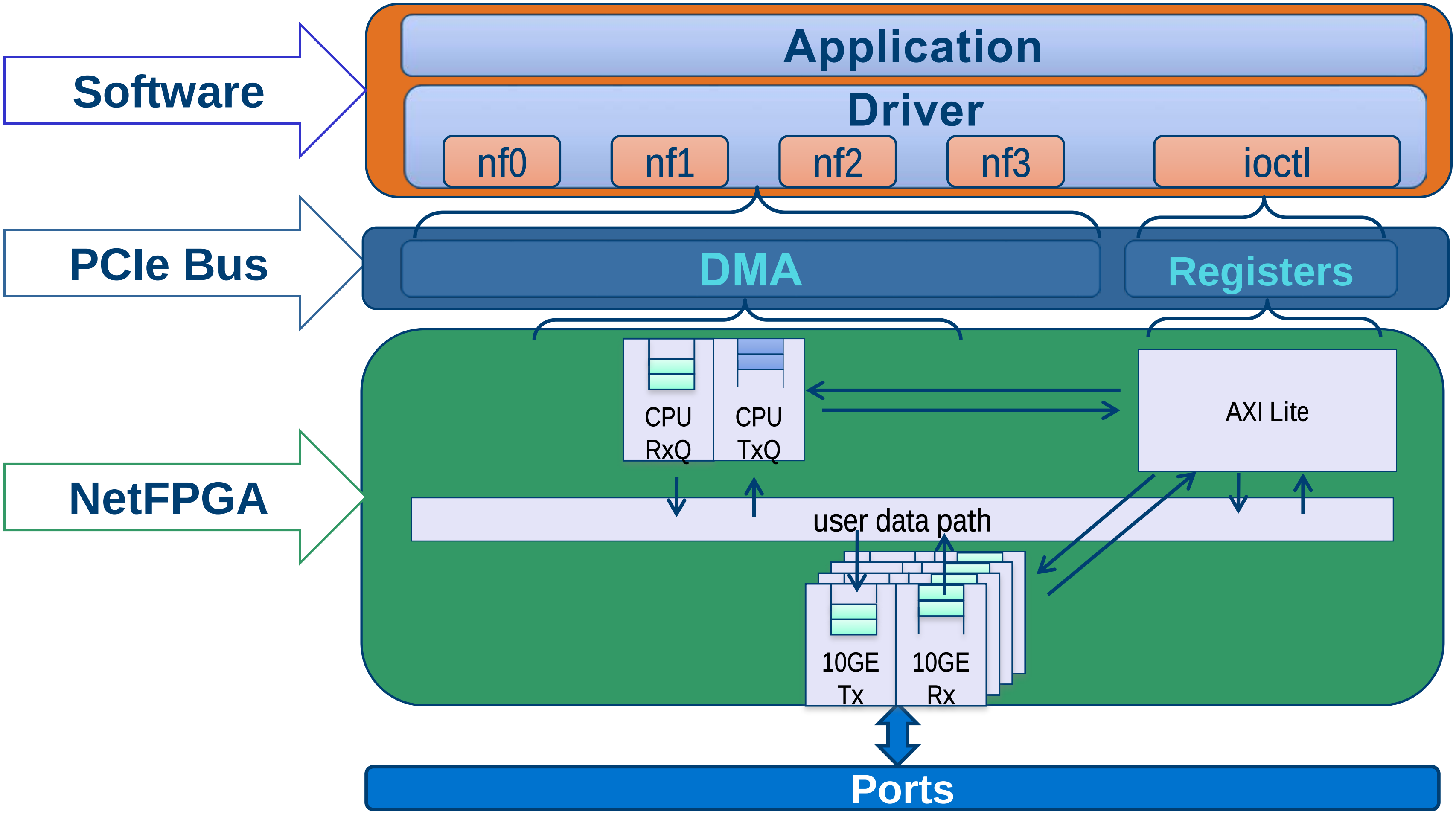
- Five stages:
  - Input port
  - Input arbitration
  - Forwarding decision and packet modification
  - Output queuing
  - Output port

- Packet-based module interface
- Pluggable design

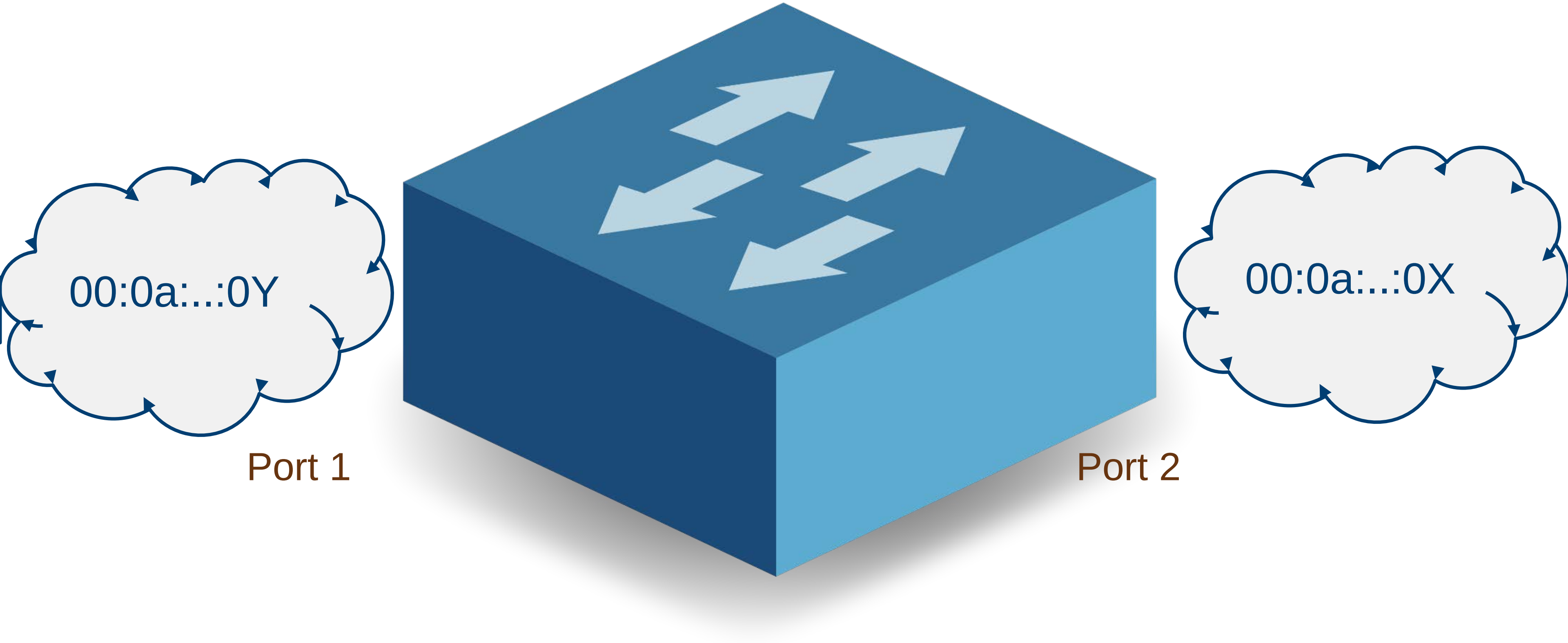


UNIVERSITY OF CAMBRIDGE

Full System Components



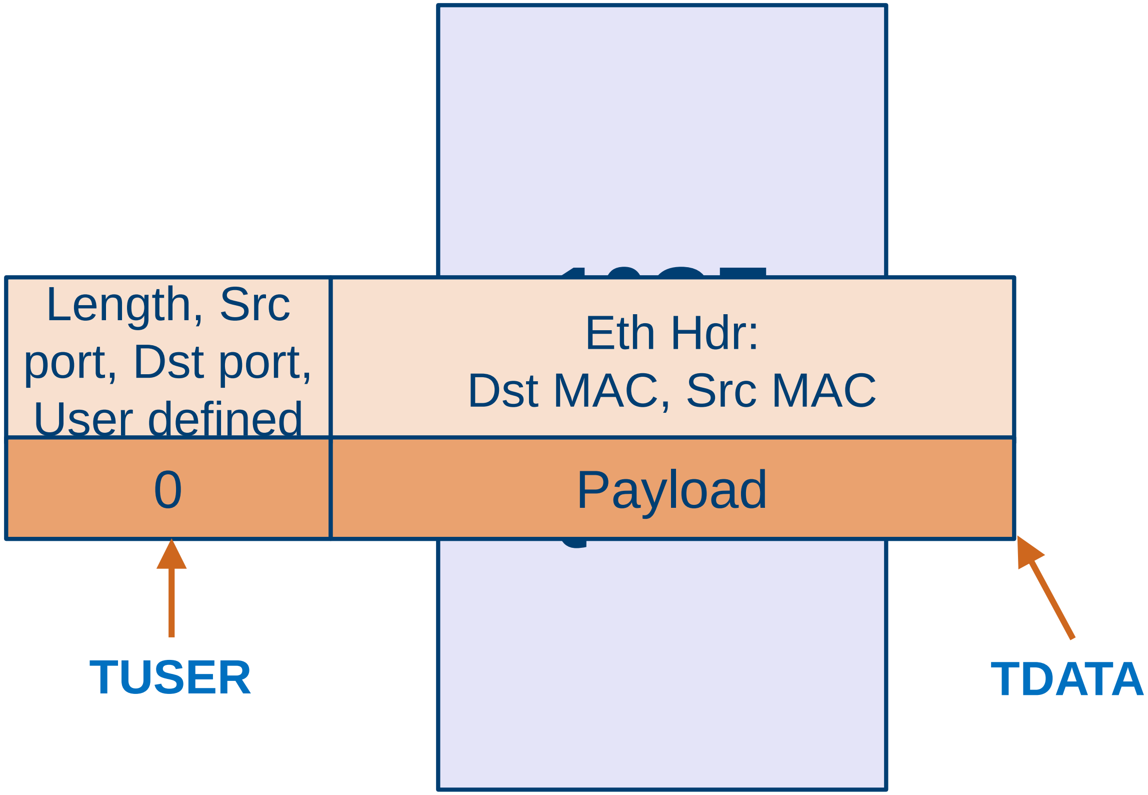
Life of a Packet through the Hardware



10GE Rx Queue

10GE
Rx
Queue

10GE Rx Queue



Length, Src port, Dst port, User defined

Eth Hdr:
Dst MAC, Src MAC

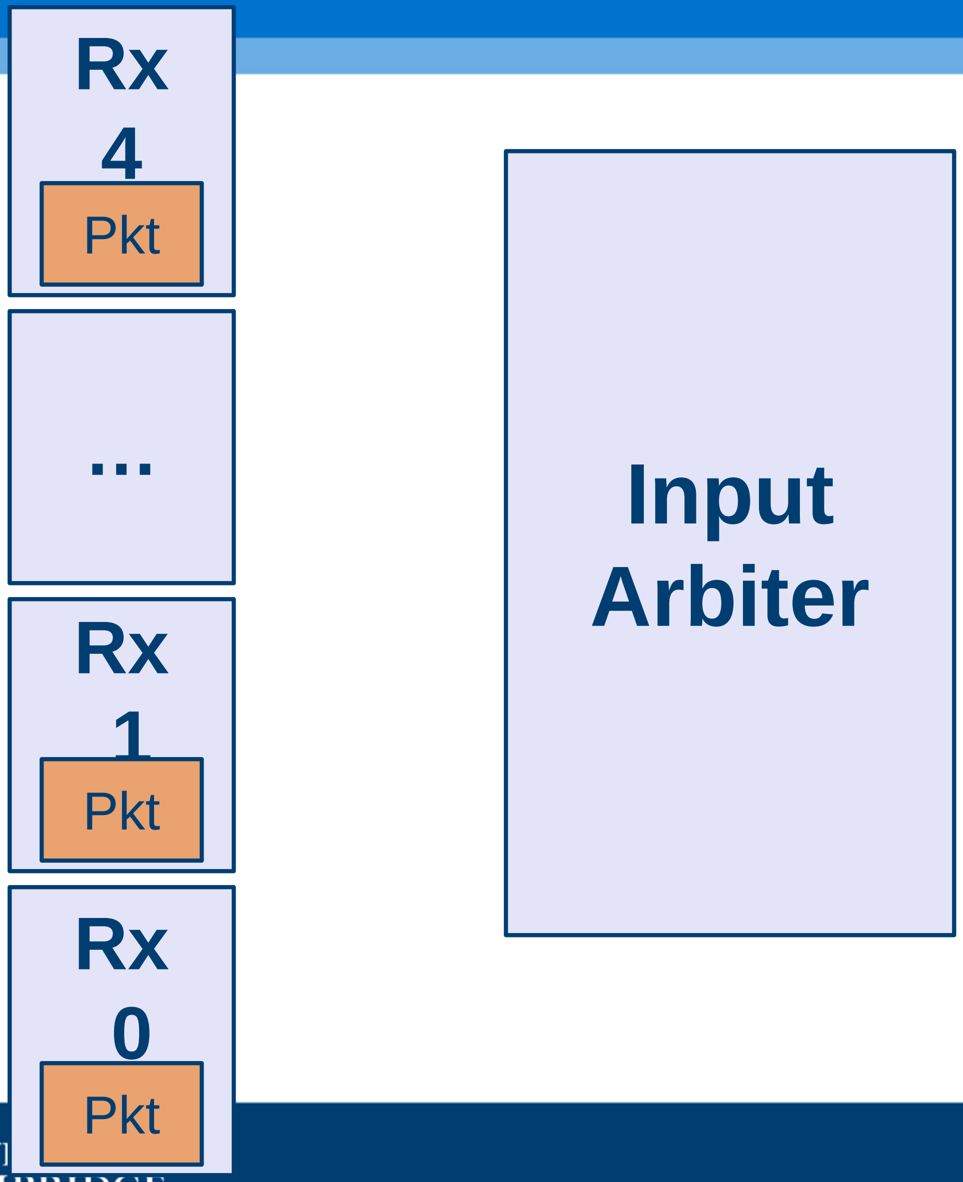
0

Payload

TUSER

TDATA

Input Arbiter



Rx 4

Pkt

…

Rx 1

Pkt

Rx 0

Pkt

Input Arbiter

Output Port Lookup

Output Port Lookup

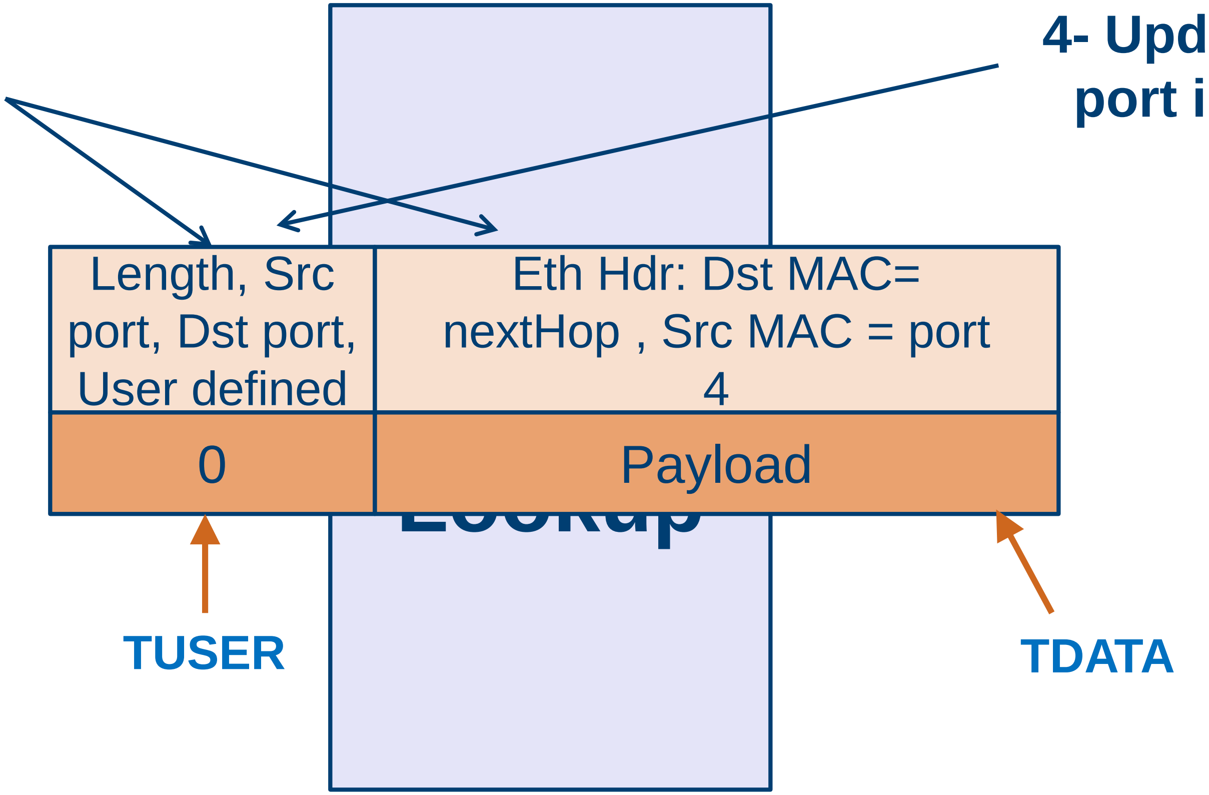
Output Port Lookup

1- Parse header: Src MAC, Dst MAC, Src port

2 - Lookup next hop MAC& output port

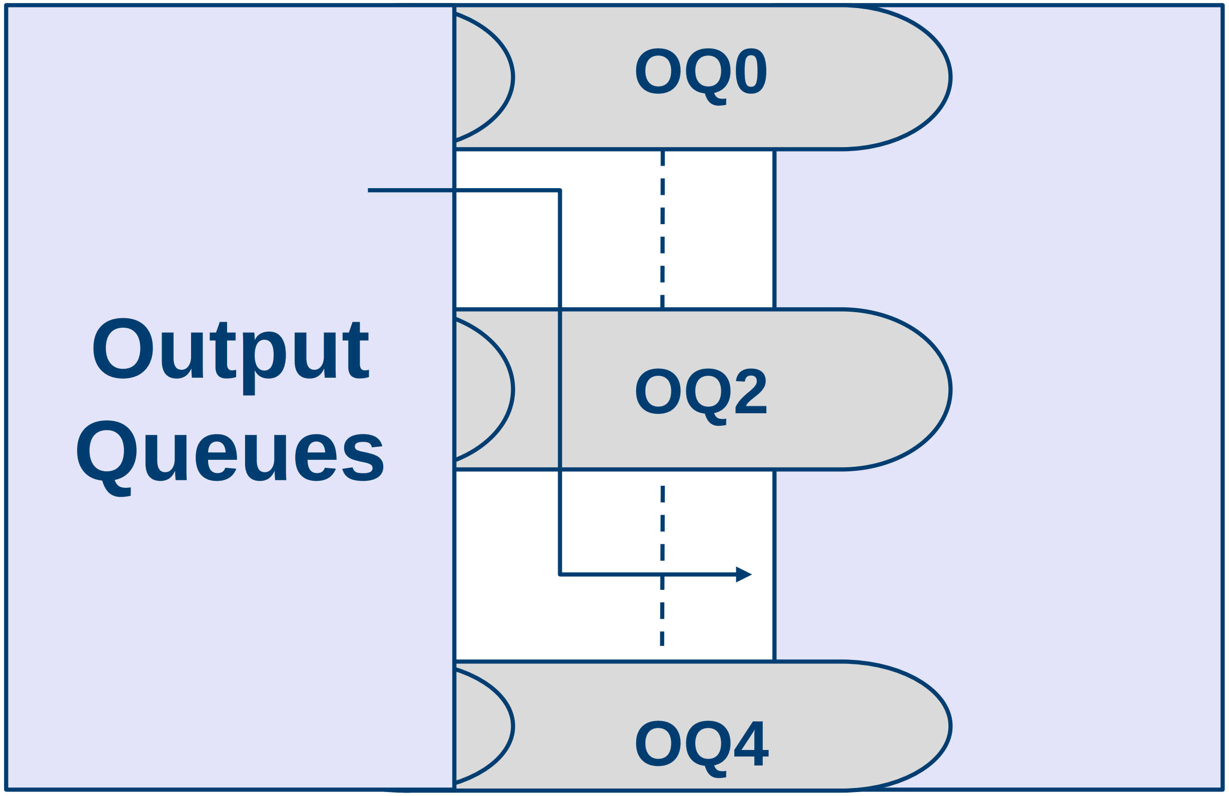
3- Learn Src MAC & Src port

4- Update output port in TUSER



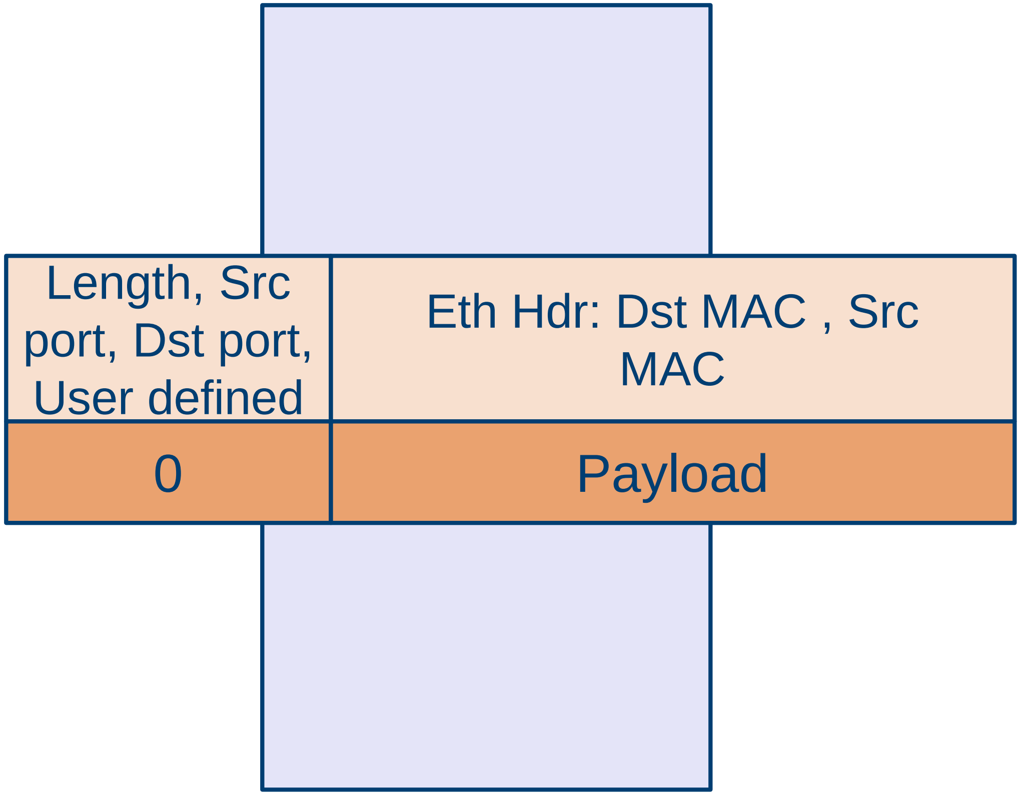
Length, Src port, Dst port, User defined	Eth Hdr: Dst MAC= nextHop , Src MAC = port 4
0	Payload

Output Queues



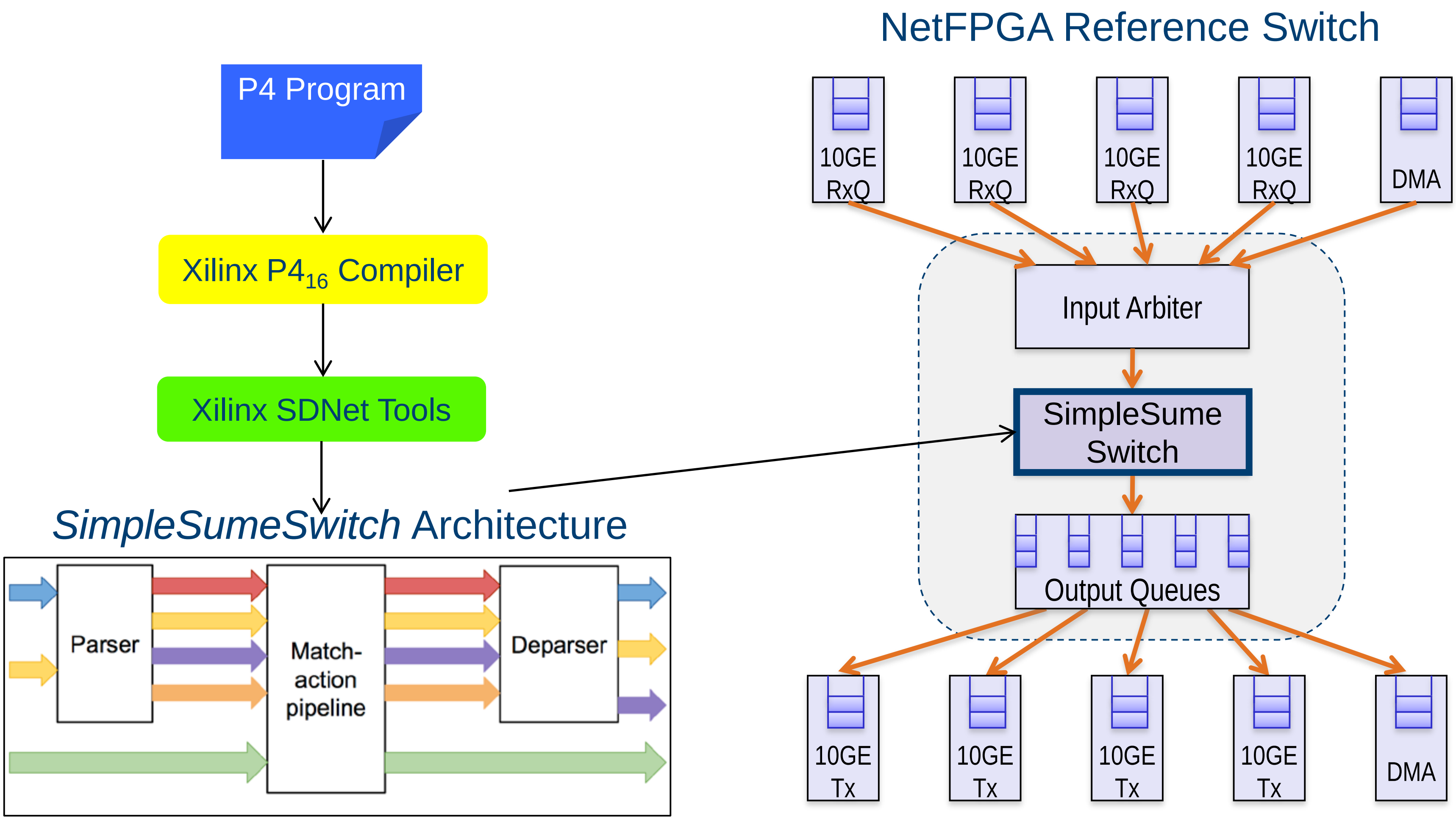
10GE Port Tx

MAC Tx Queue



Length, Src port, Dst port, User defined	Eth Hdr: Dst MAC , Src MAC
0	Payload

P4→NetFPGA Compilation Overview



NetFPGA Reference Switch

P4 Program

Xilinx P4$_{16}$ Compiler

Xilinx SDNet Tools

SimpleSumeSwitch Architecture

NetFPGA-Host Interaction

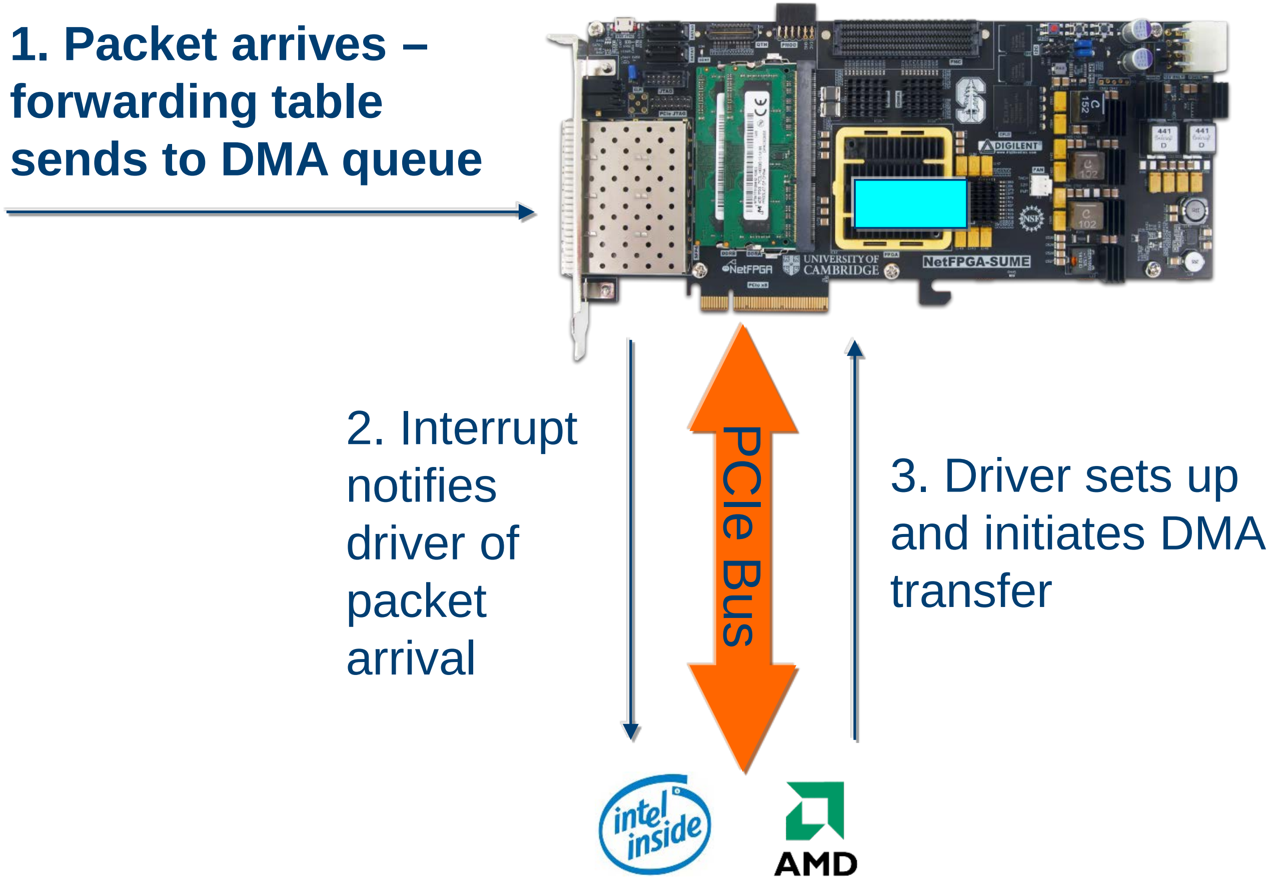
- Linux driver interfaces with hardware
  - Packet interface via standard Linux network stack
  - Register reads/writes via ioctl system call with wrapper functions:
    - rwaxi(int address, unsigned *data);

    eg:

    rwaxi(**0x7d4000000**, &val);

NetFPGA-Host Interaction

NetFPGA to host packet transfer



1. Packet arrives – forwarding table sends to DMA queue

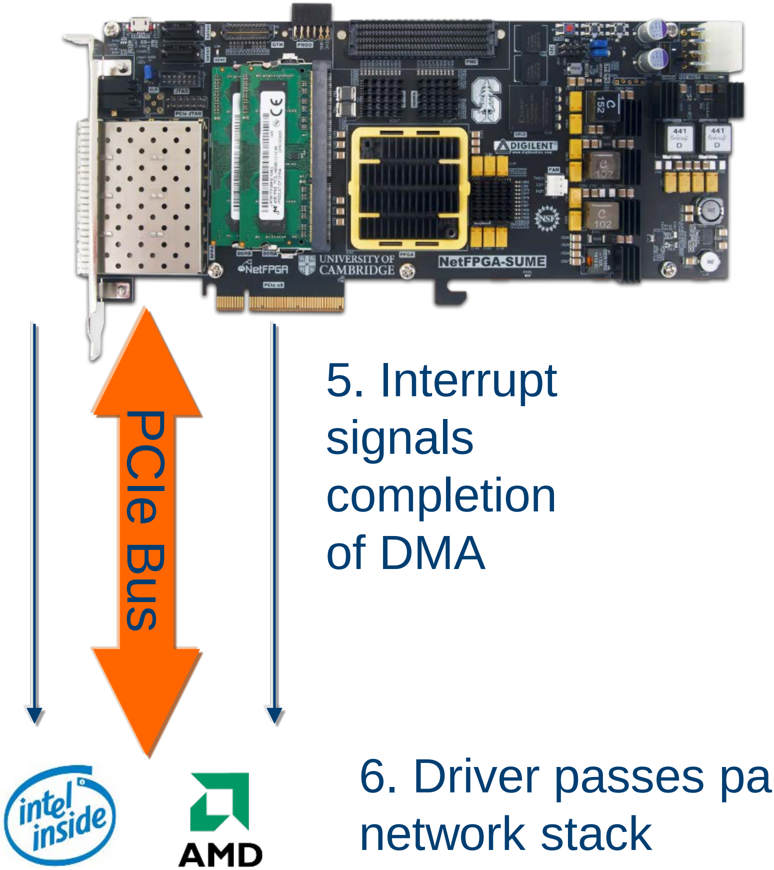
2. Interrupt notifies driver of packet arrival

PCIe Bus

3. Driver sets up and initiates DMA transfer

NetFPGA-Host Interaction

NetFPGA to host packet transfer (cont.)



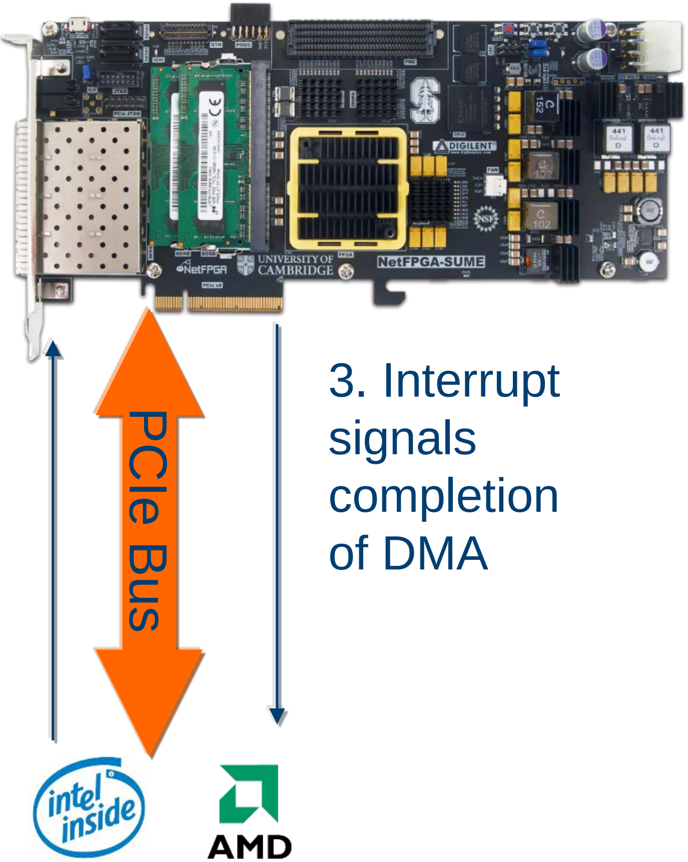
4. NetFPGA transfers packet via DMA

5. Interrupt signals completion of DMA

6. Driver passes packet to network stack

NetFPGA-Host Interaction

Host to NetFPGA packet transfers



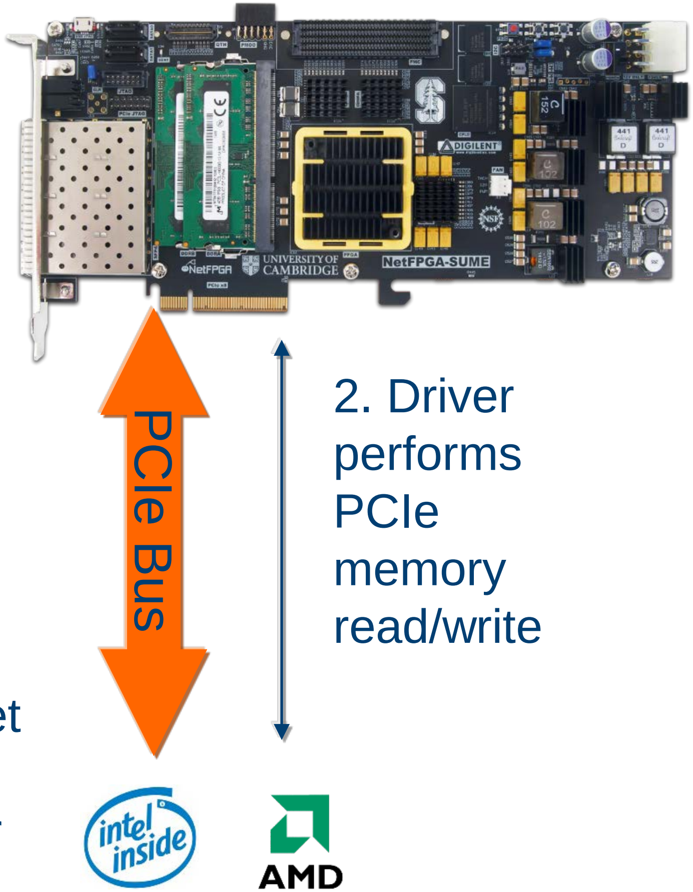
1. Software sends packet via network sockets Packet delivered to driver

2. Driver sets up and initiates DMA transfer

3. Interrupt signals completion of DMA

NetFPGA-Host Interaction

Register access



PCIe Bus

2. Driver performs PCIe memory read/write

1. Software makes ioctl call on network socket

ioctl passed to driver

Section IV: Today’s Lab Session

Today: Getting to know the NetFPGA Platform

- Starting point: experimenting with existing projects
- Then: Learning how to modify projects
- Follow the instructions in the handout
- 1-2 people per machine (2-3 people per pair of machines)