UNIVERSITY OF WATERLOO

CS 856: Programmable Networks

Lecture 4: Smart Network Interface Cards

Mina Tahmasbi Arashloo

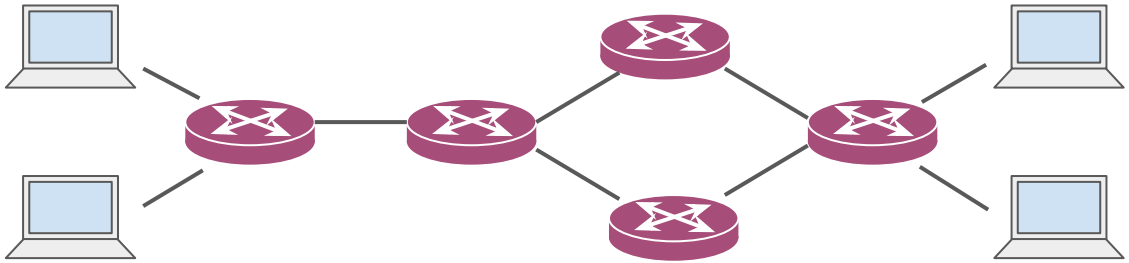
Winter 2024

Logistics

- Reviews are due **Monday, Feb 5, at 5pm.**
- Assignment 1 is due **Monday, Feb 26th.**

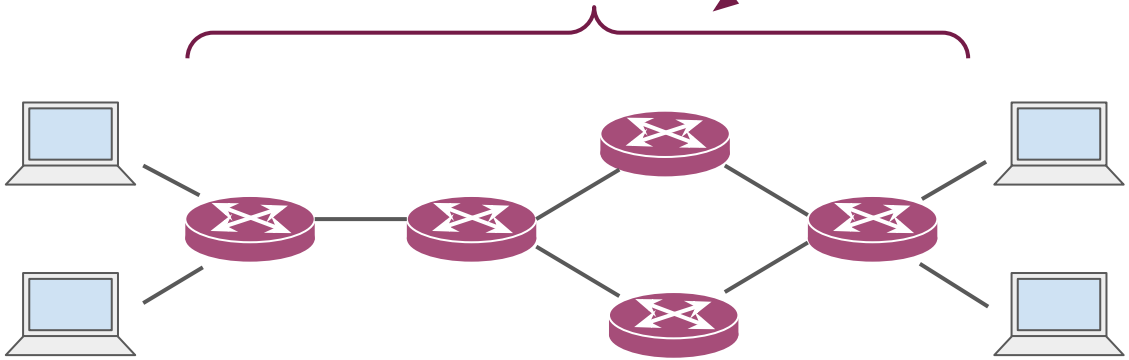
Network programmability at the end points

- So far, we have mostly talked about programmability for devices and functionality inside the network
  - e.g., switches and routers, forwarding, etc.



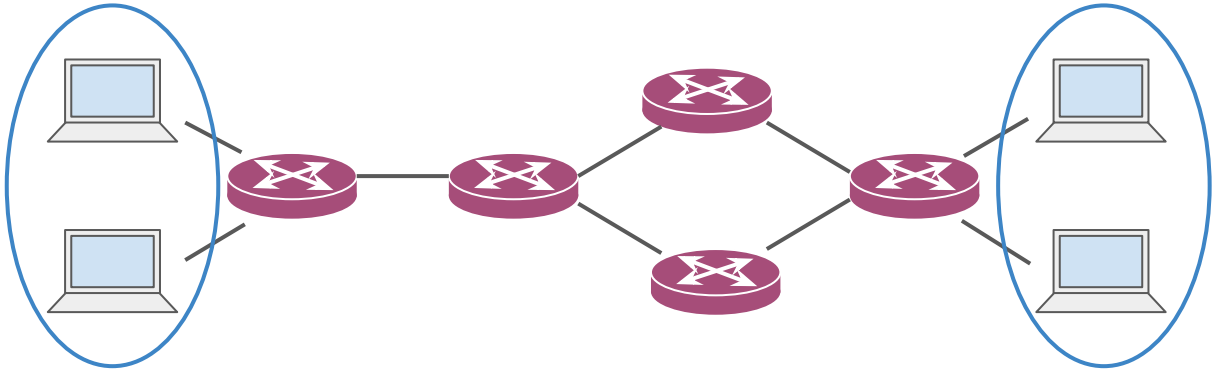
Network programmability at the end points

- So far, we have mostly talked about programmability for devices and functionality inside the network
  - e.g., switches and routers, forwarding, etc.

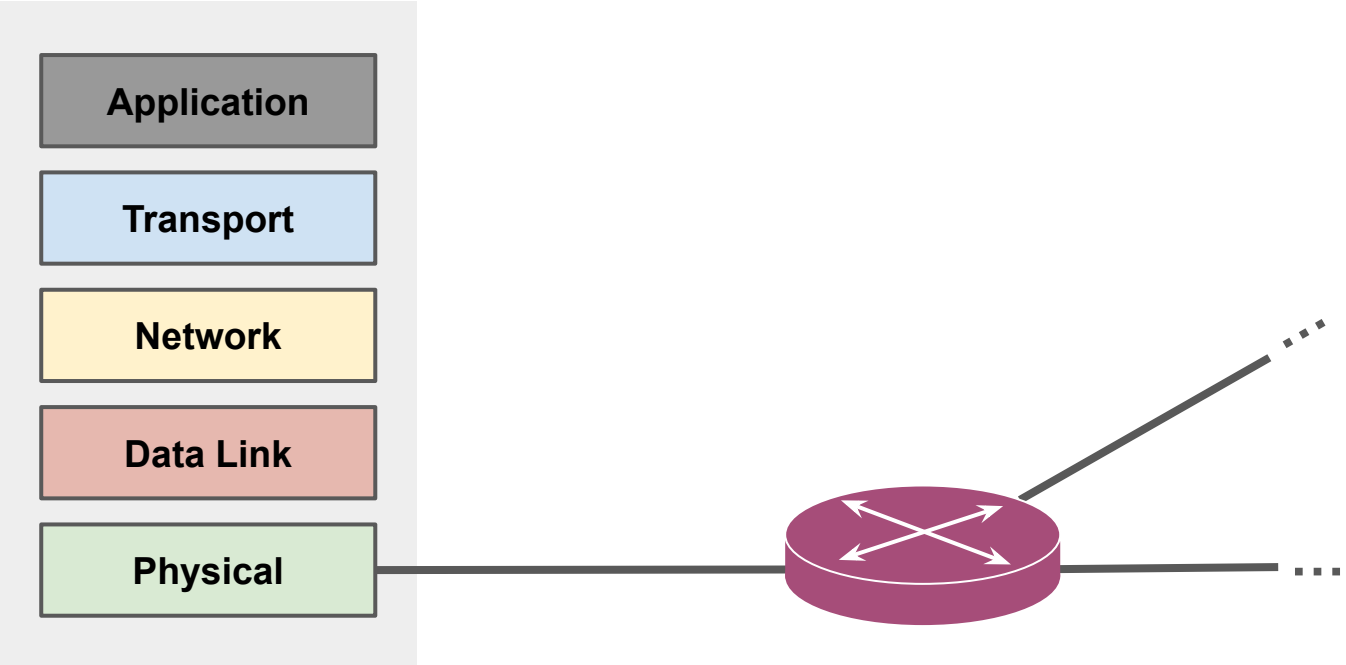


Network programmability at the end points

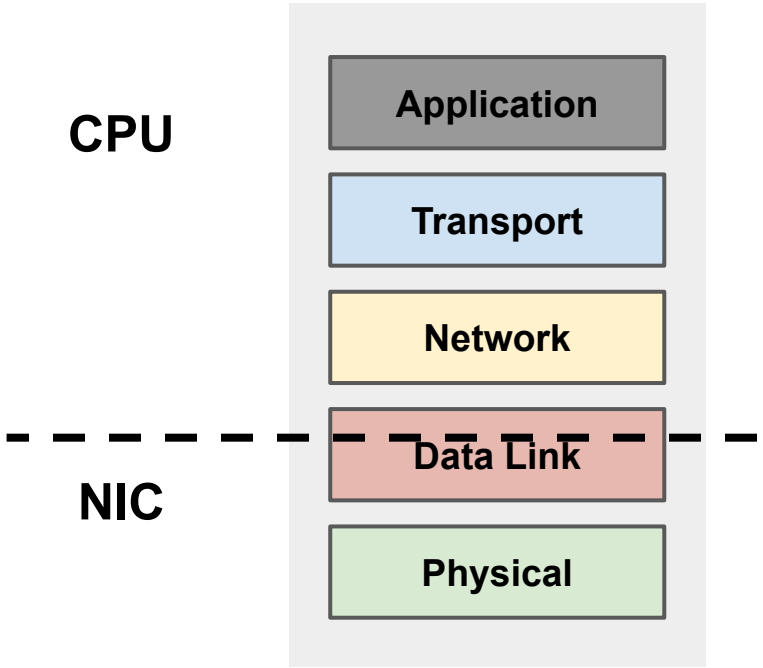
- but that's not the only place where network processing happens.
- There is a whole network stack for preparing and processing packets at the end points



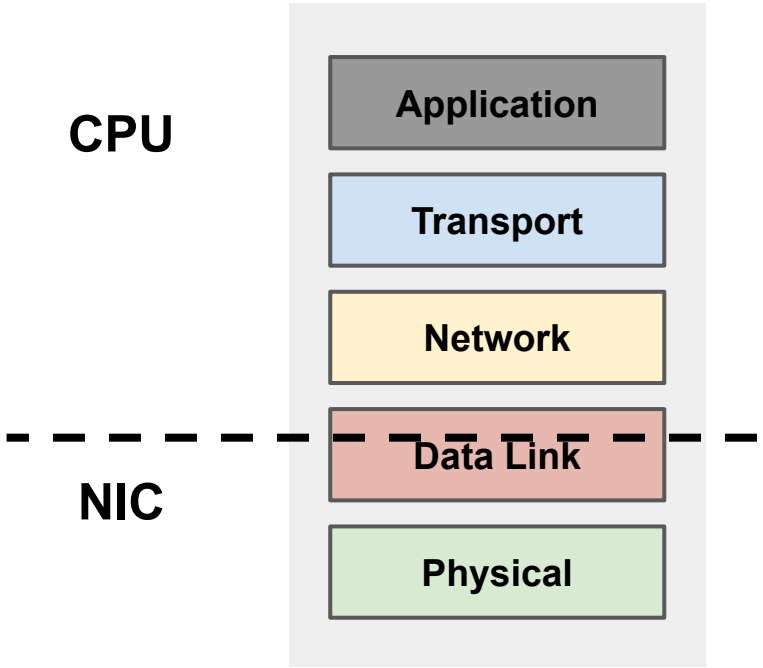
End-Point network stack



Network interface cards (NICs)



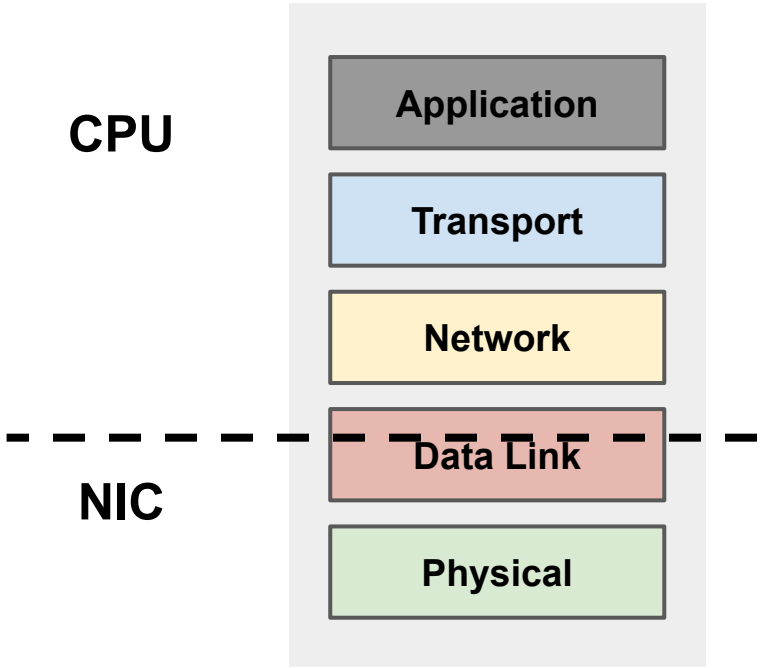
Network interface cards (NICs)



On transmit (egress):

- The host CPU generates packets on application request
- Packets are sent to the NIC over PCIe
- The NIC transforms packets to bits and sends them over the link

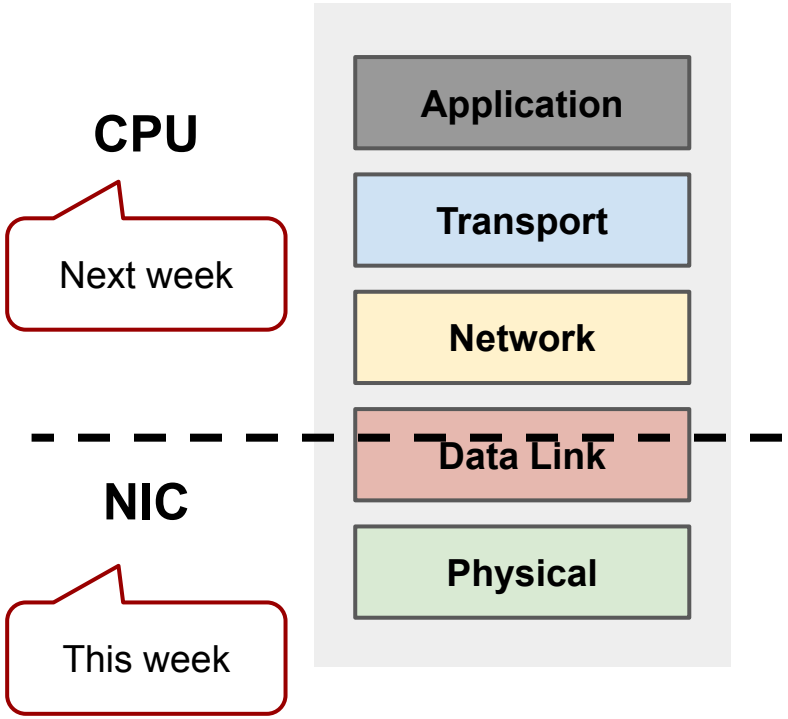
Network interface cards (NICs)



On receive (ingress)

- The NIC turns bits into packets
- Packets are sent to the host over PCIe
- The host CPU processes packets and delivers them to applications

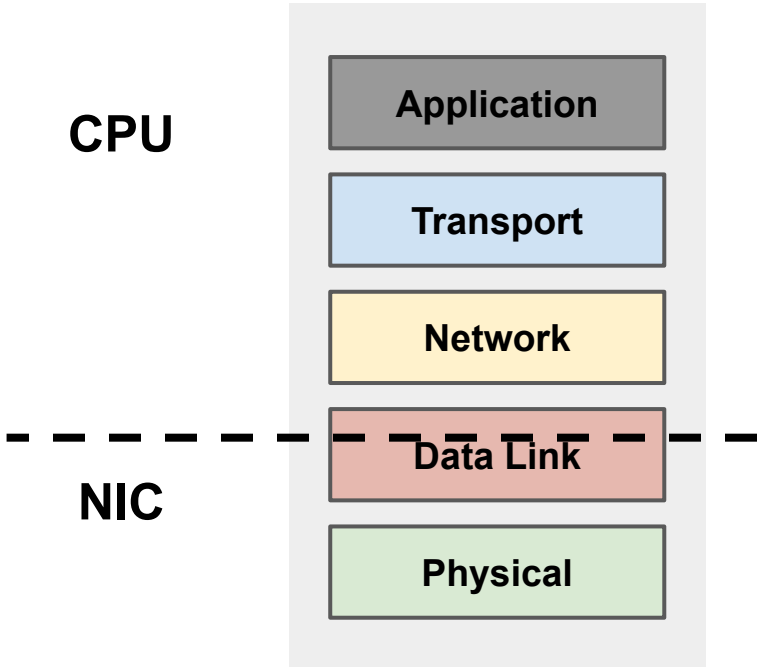
Network interface cards (NICs)



On receive (ingress)

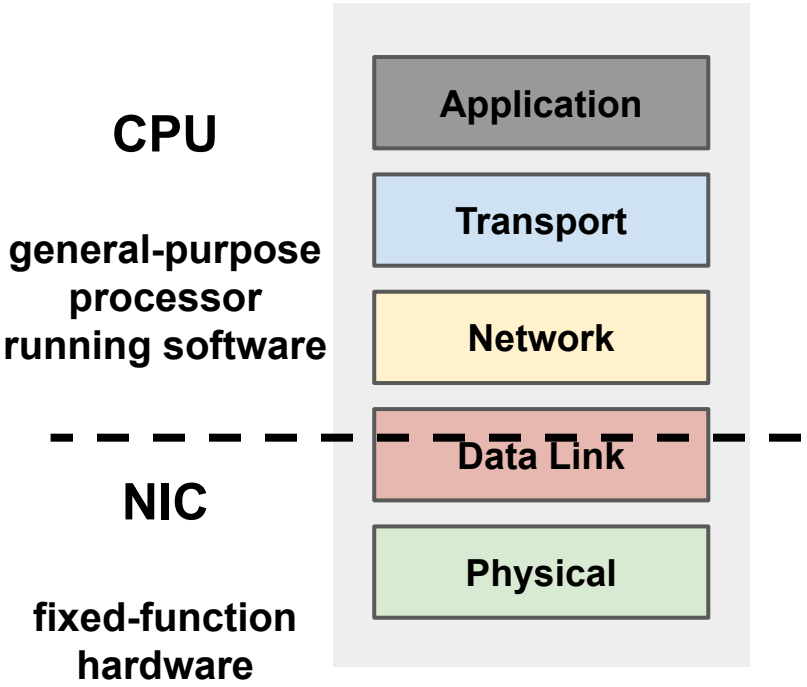
- The NIC turns bits into packets
- Packets are sent to the host over PCIe
- The host CPU processes packets and delivers them to applications

Network interface cards (NICs)



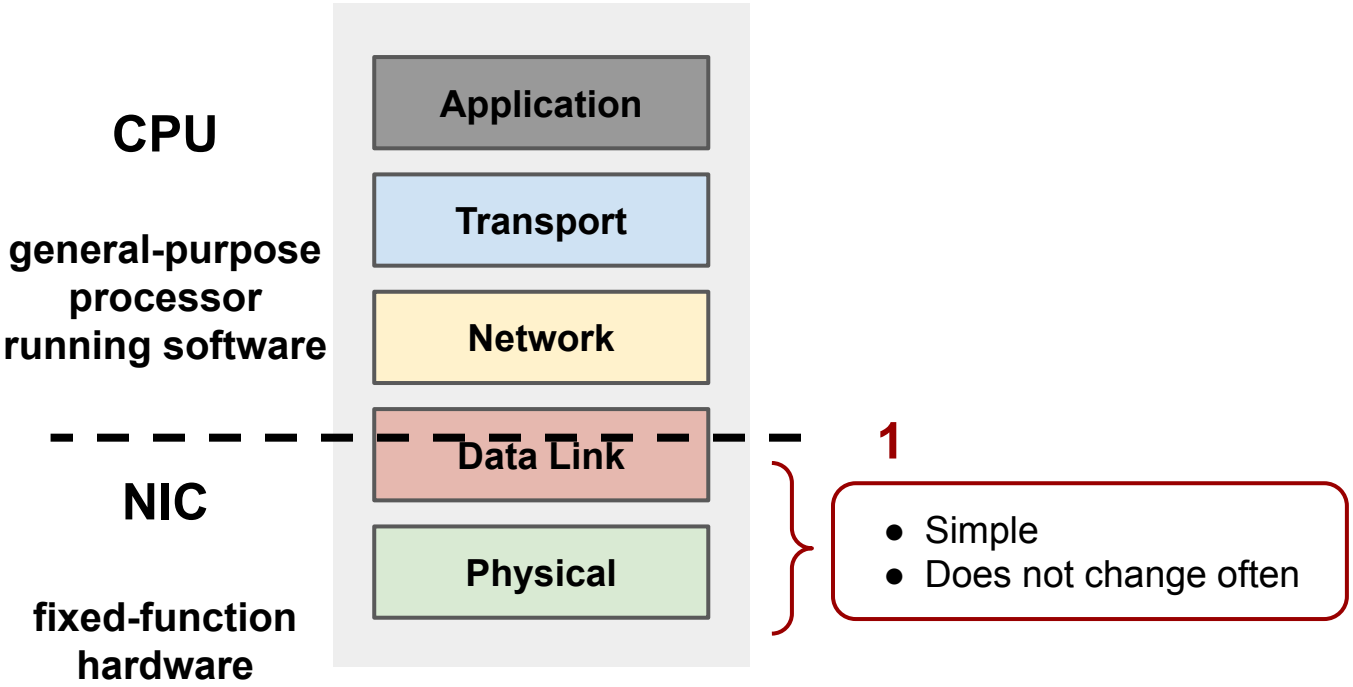
Great division of labor!

Network interface cards (NICs)

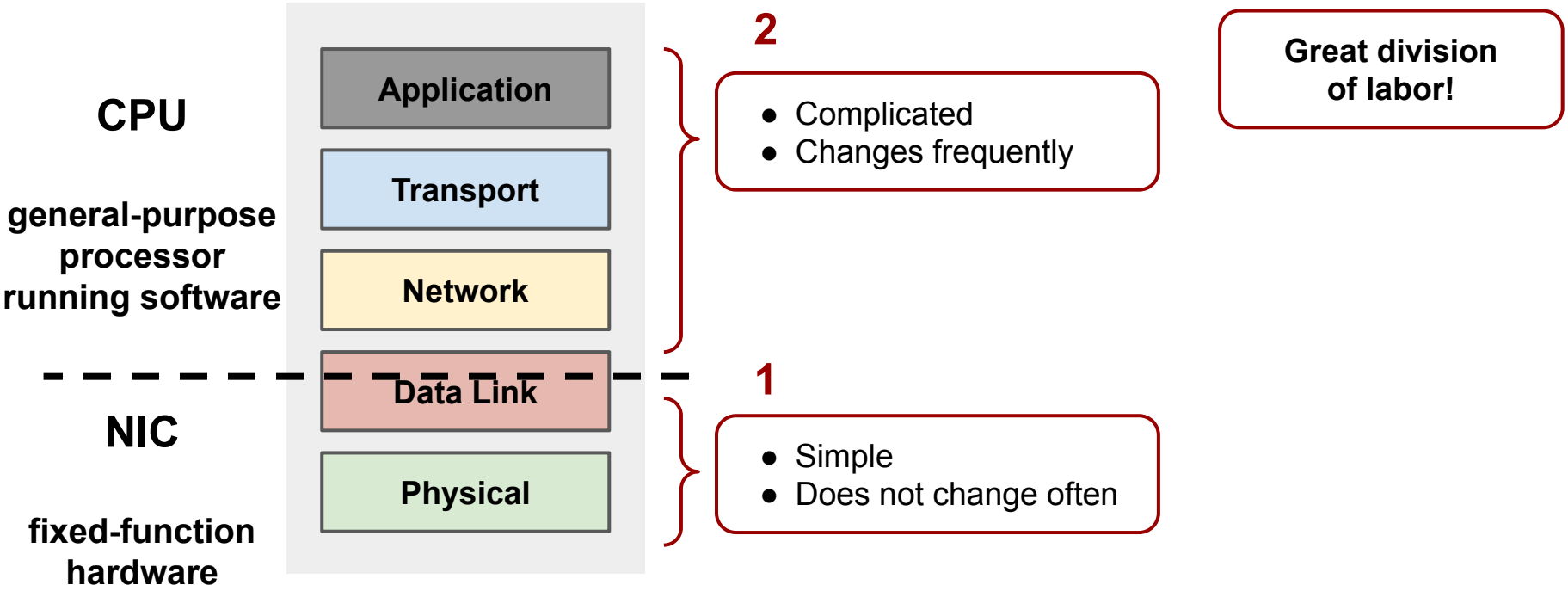


Great division of labor!

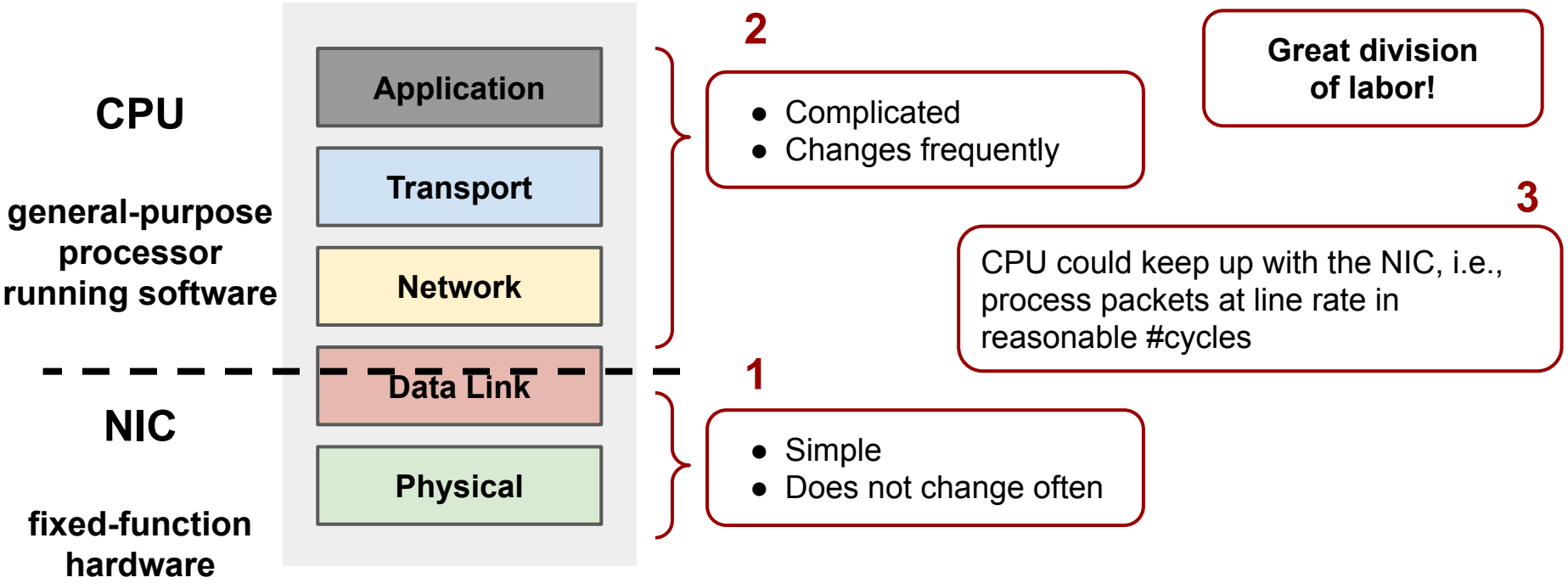
Network interface cards (NICs)



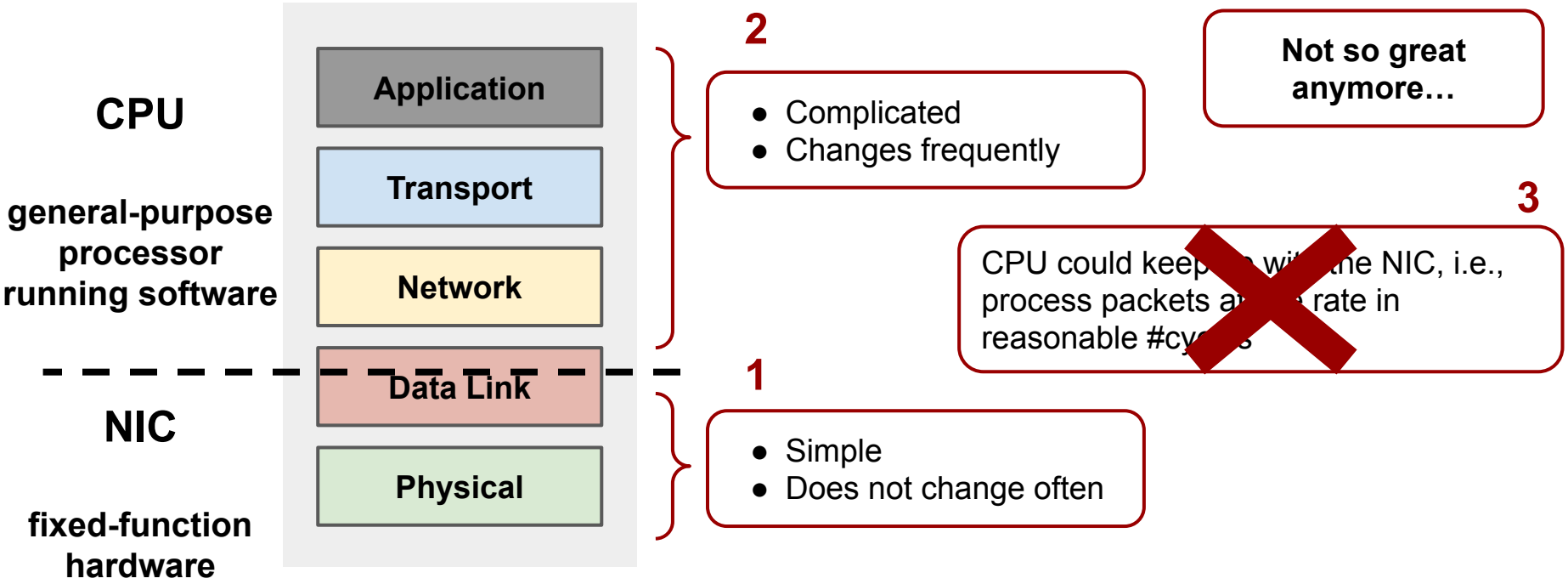
Network interface cards (NICs)



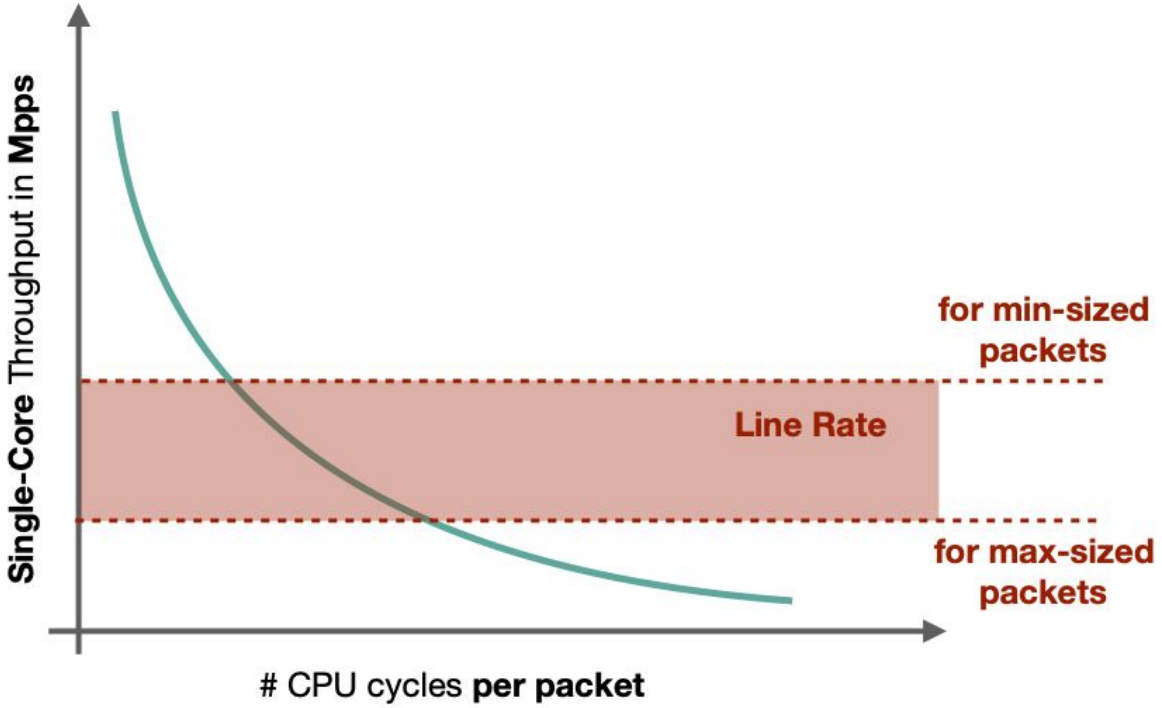
Network interface cards (NICs)



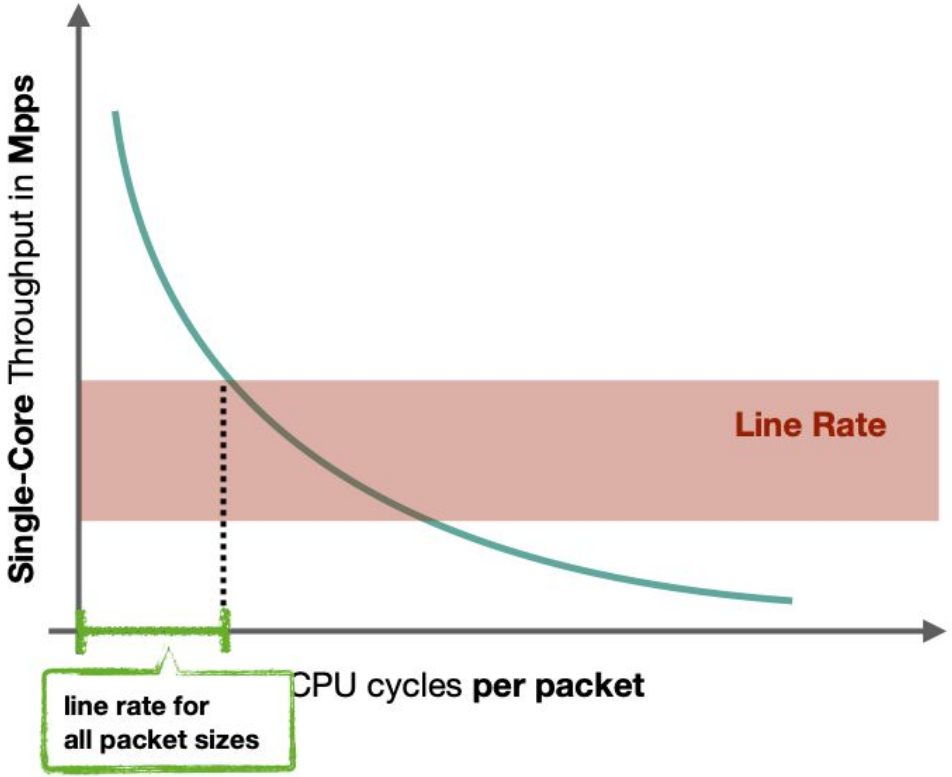
Network interface cards (NICs)



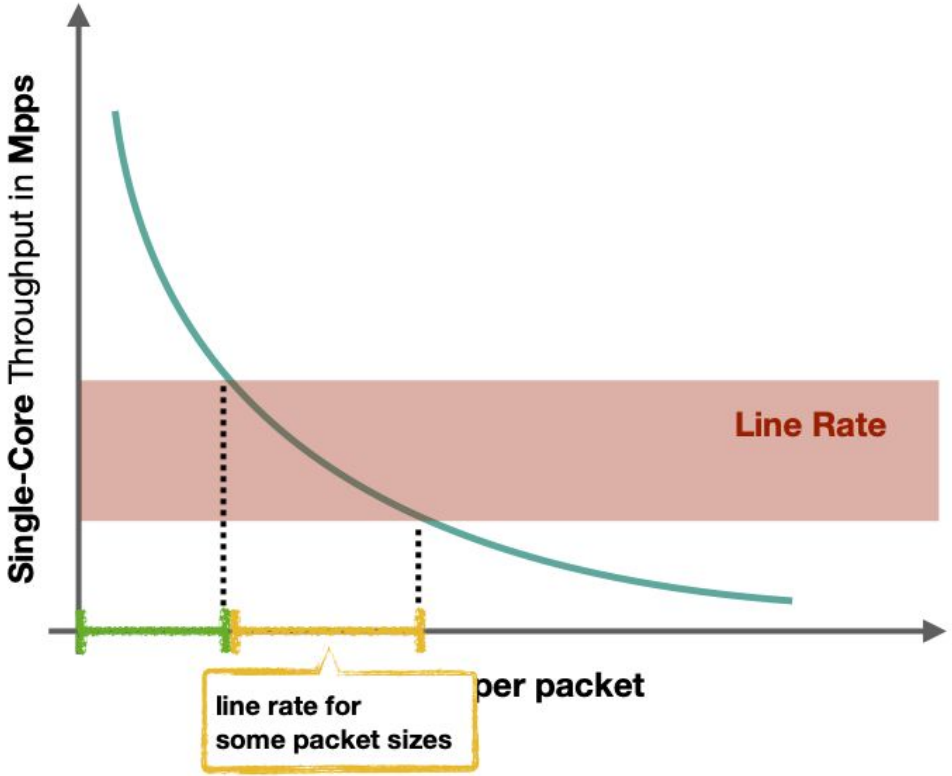
Limits of Software Packet Processing



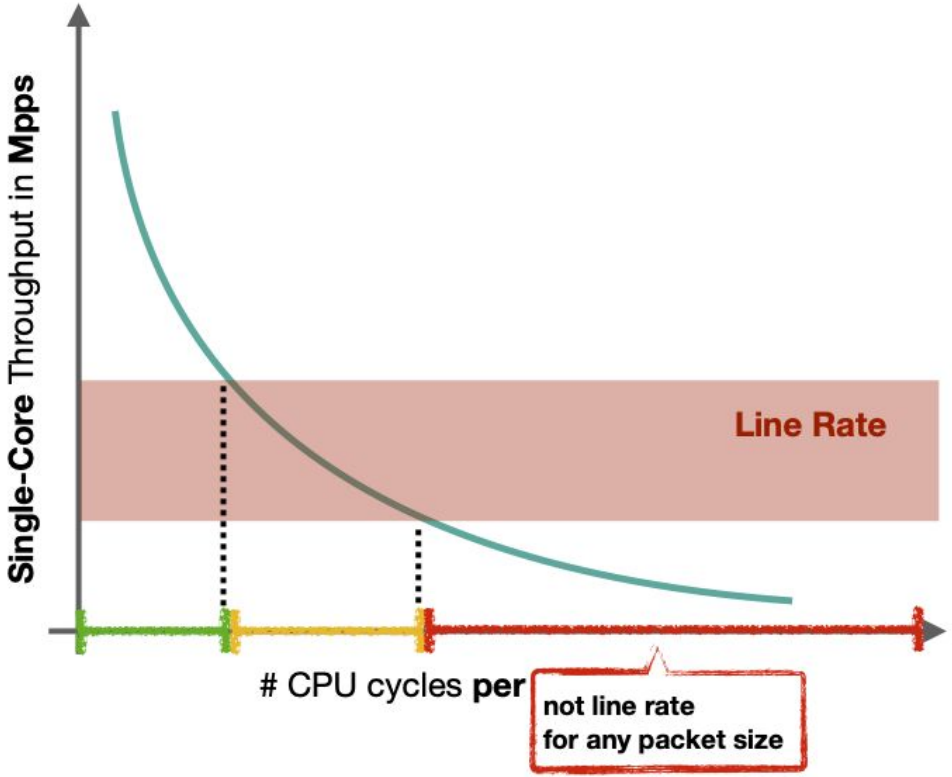
Limits of Software Packet Processing



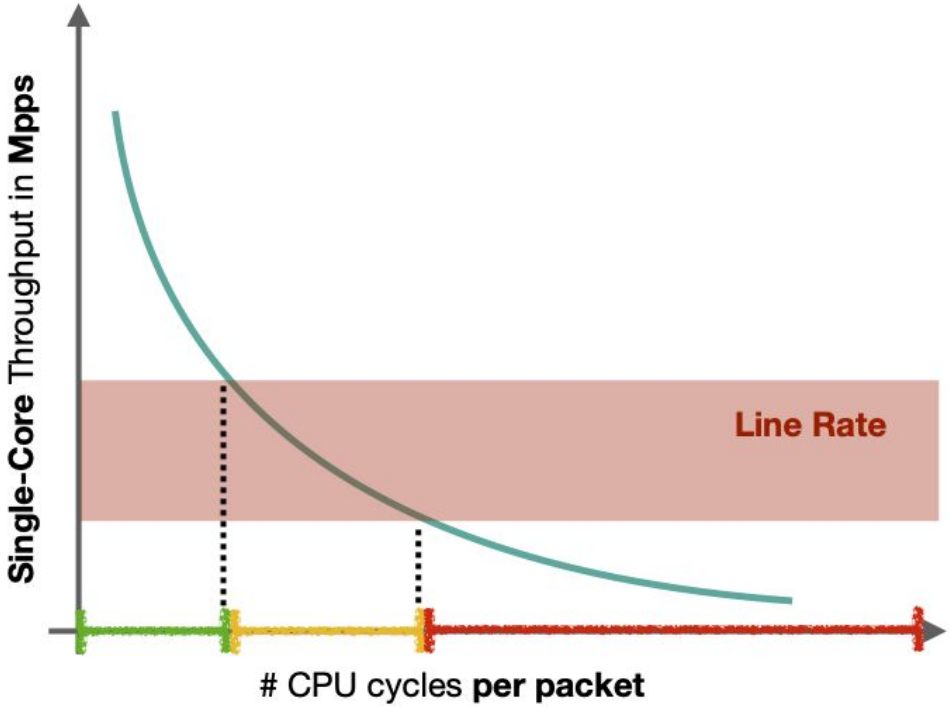
Limits of Software Packet Processing



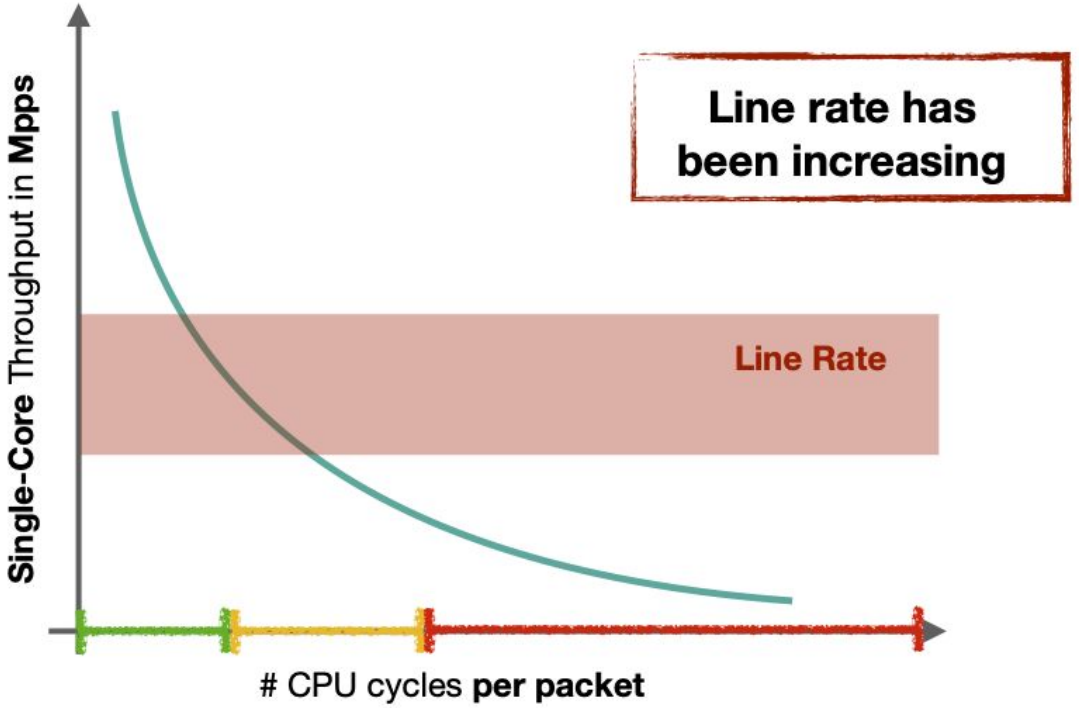
Limits of Software Packet Processing



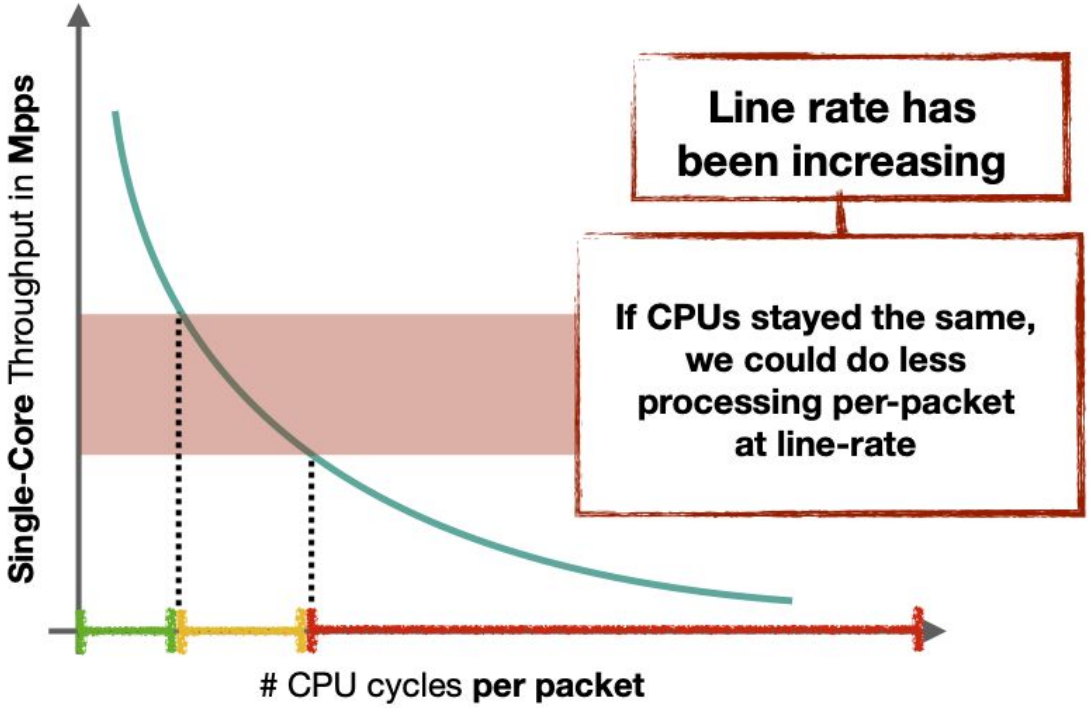
Limits of Software Packet Processing



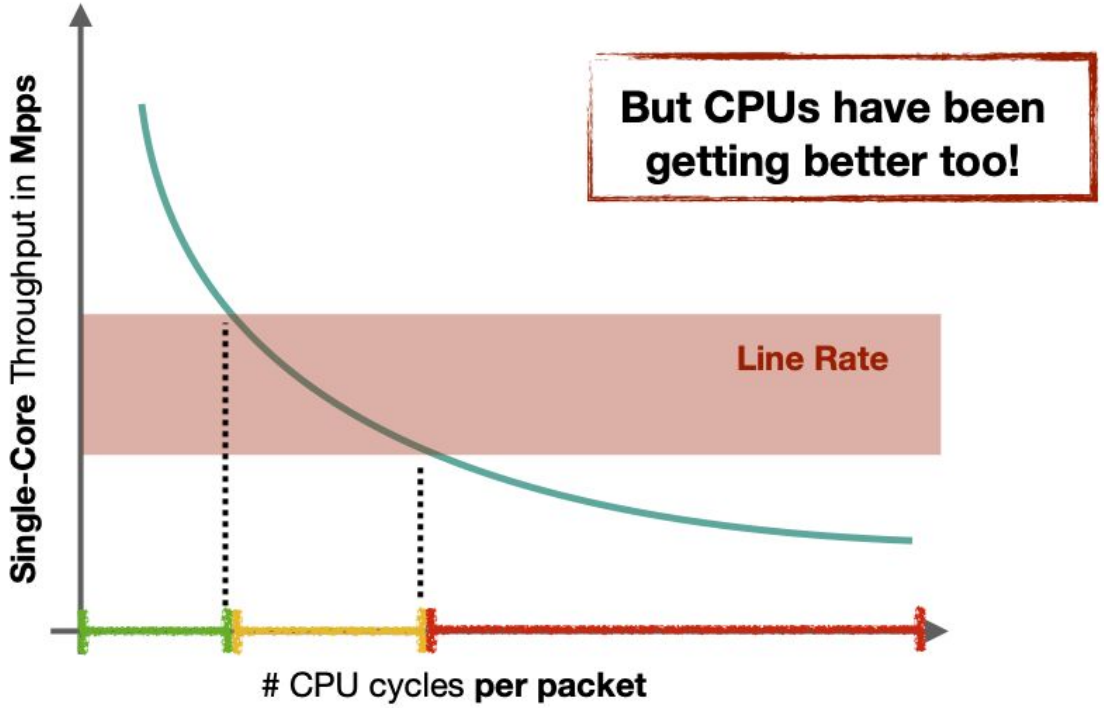
Limits of Software Packet Processing



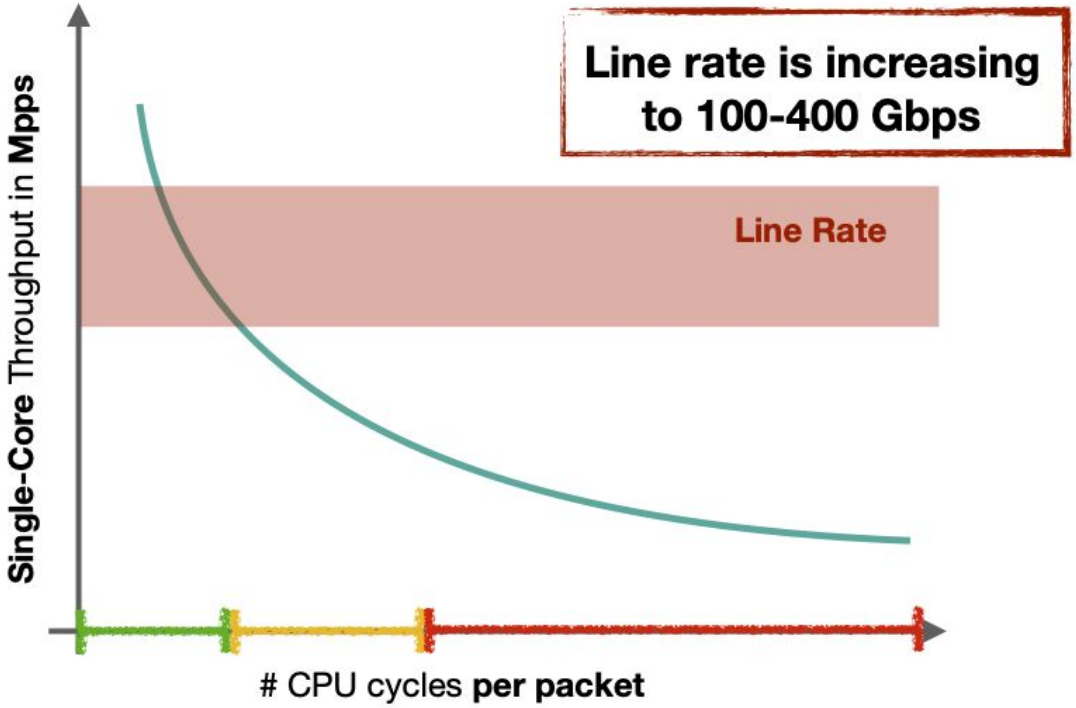
Limits of Software Packet Processing



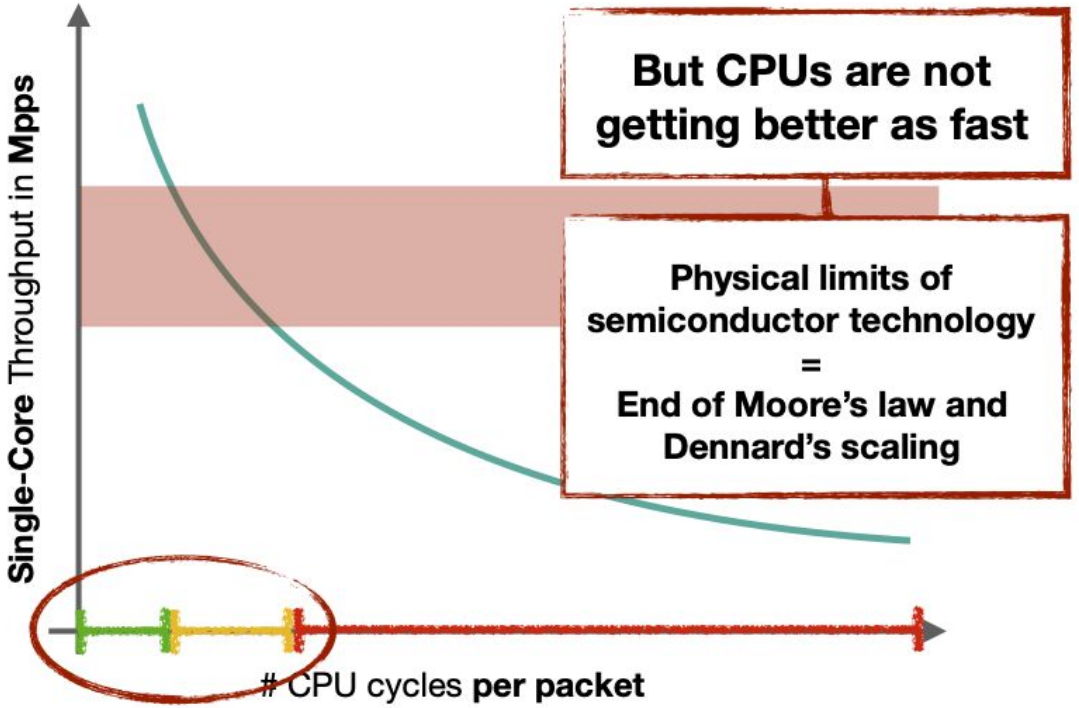
Limits of Software Packet Processing



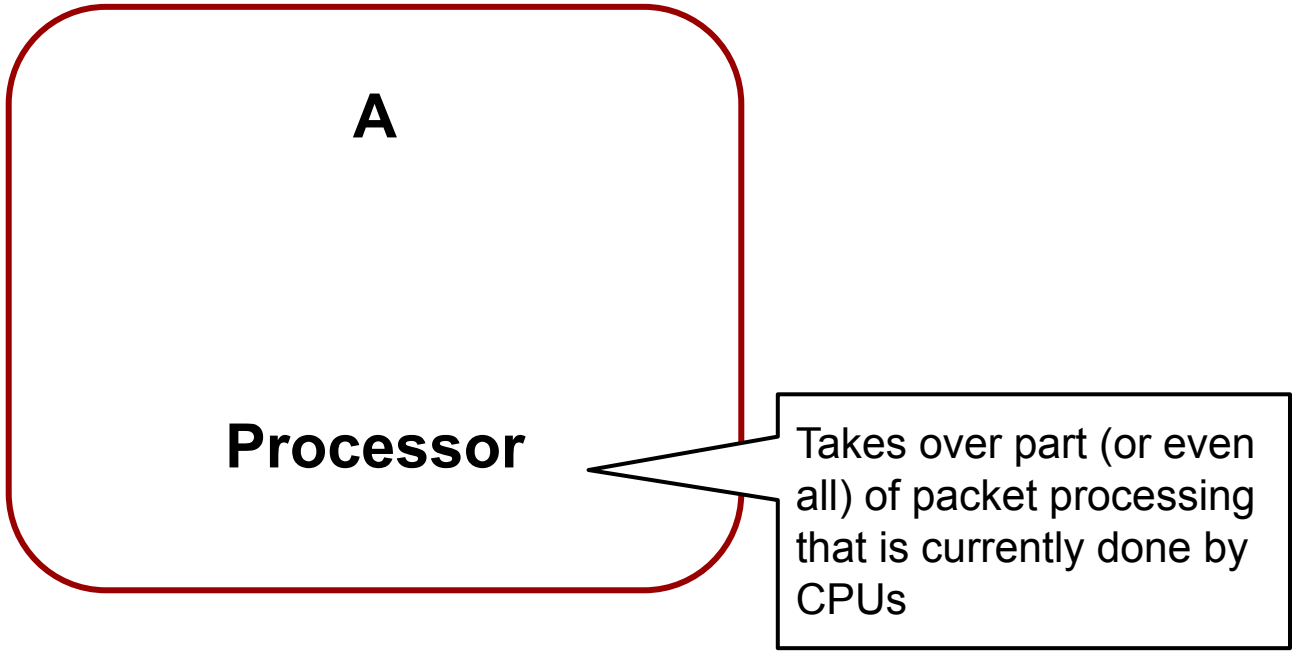
Limits of Software Packet Processing



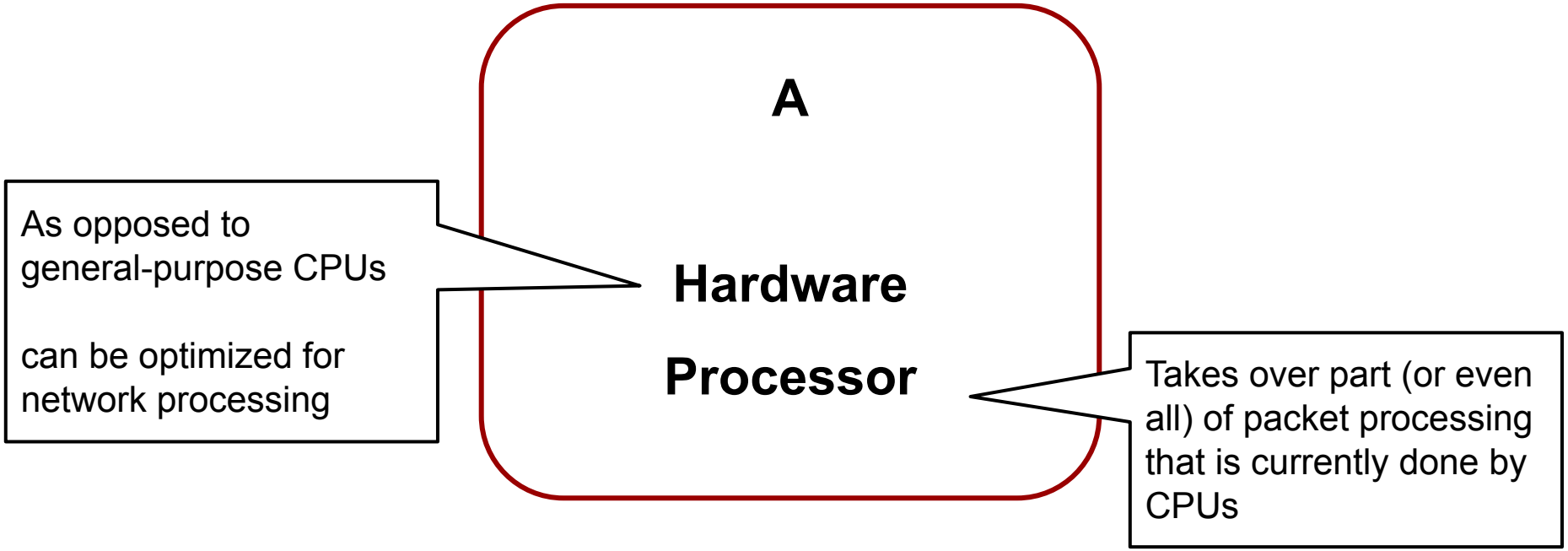
Limits of Software Packet Processing



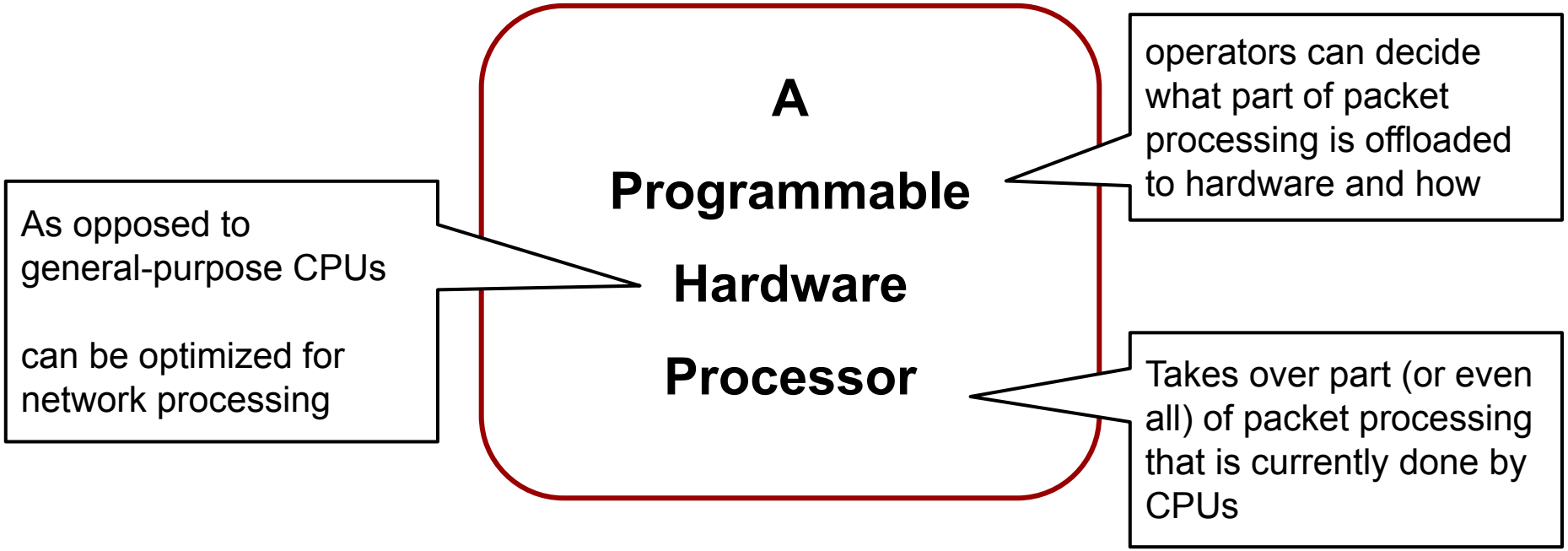
Solution?


A
Processor
Takes over part (or even all) of packet processing that is currently done by CPUs

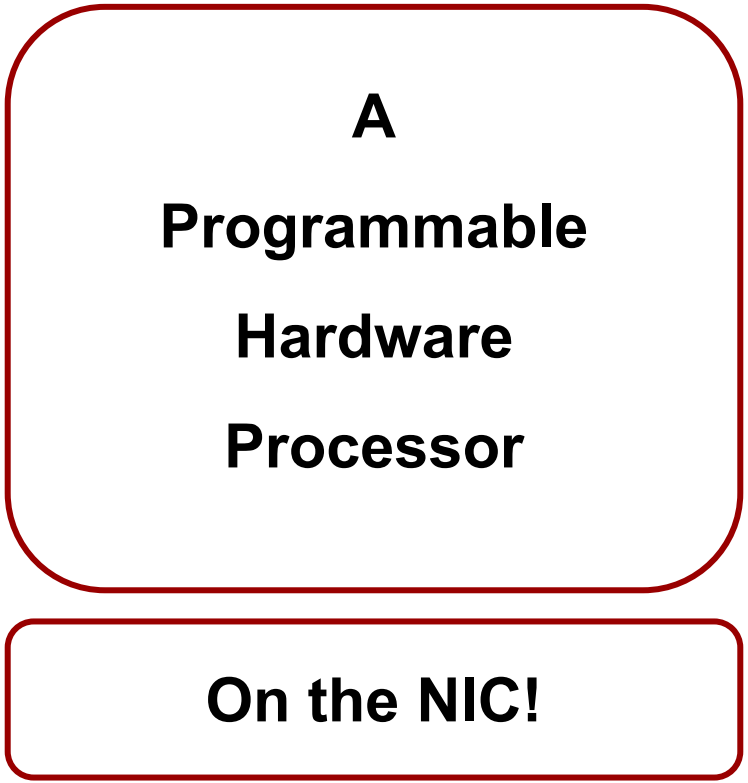
Solution?


A
Hardware
Processor
As opposed to general-purpose CPUs
can be optimized for network processing
Takes over part (or even all) of packet processing that is currently done by CPUs

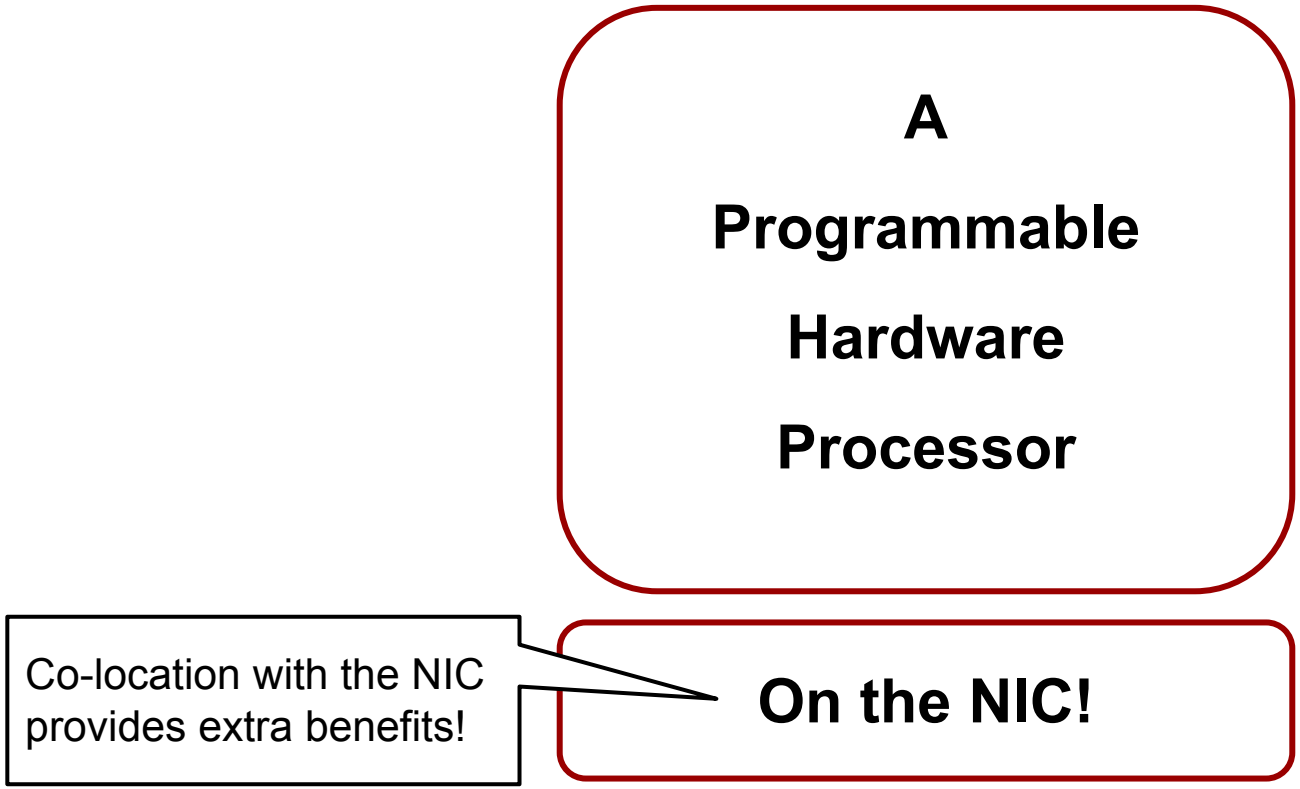
Solution?


A
Programmable
Hardware
Processor
As opposed to general-purpose CPUs
can be optimized for network processing
operators can decide what part of packet processing is offloaded to hardware and how
Takes over part (or even all) of packet processing that is currently done by CPUs

Solution?



Solution?


A
Programmable
Hardware
Processor
On the NIC!
Co-location with the NIC provides extra benefits!

Smart NICs!

A regular NIC

+

A programmable domain-specific hardware

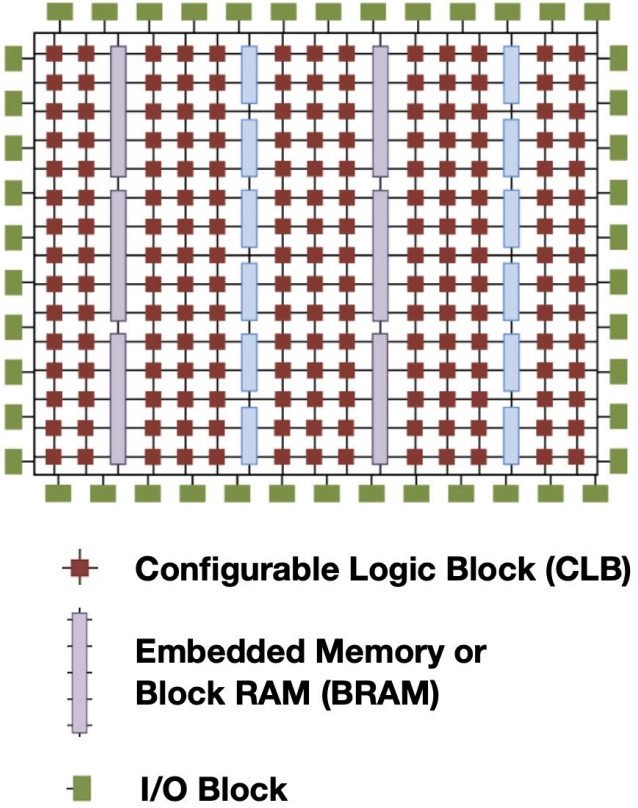
A closer look at the hardware

- Field Programmable Gate Arrays (FPGAs)
- Multi-Core Systems on Chip (SoCs)
- P4-Programmable pipelines
- Or combinations of the above …

FPGAs

- An FPGA is a collection of small configurable logic and memory blocks

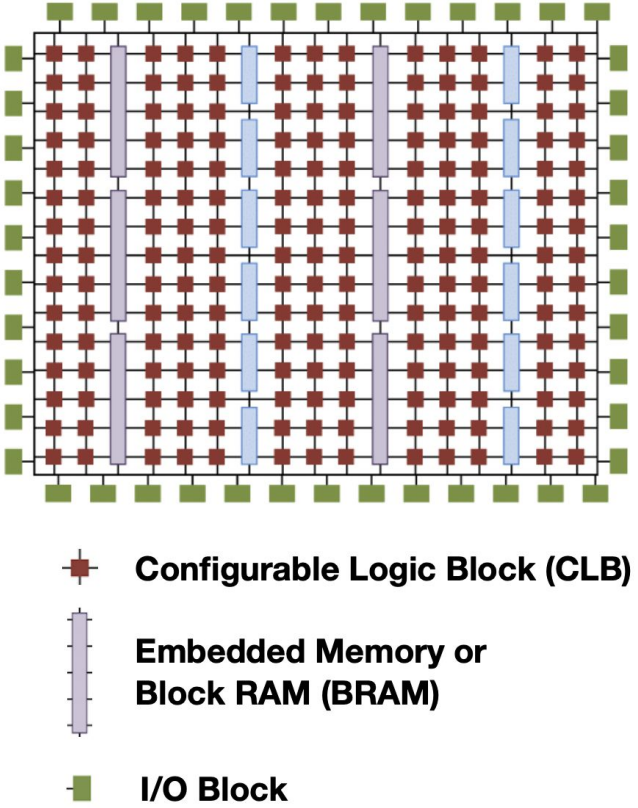
- Programmers can write code to assemble these blocks to perform their desired processing



FPGAs

Why is an FPGA a popular hardware choice for smart NICs?

- FPGA hardware resources (logic and memory) can be highly customized for the intended computation
- Great fit for highly-parallelizable computation



Multi-Core Systems on Chip

- A “small” computer on a single chip
- Includes (light-weight) processing cores and a memory hierarchy
- Why is it a popular hardware choice for smart NICs?
  - Programming model is close to software
  - Cores (and the architecture) can be specialized for network processing

FPGAs vs SoCs for network processing

	FPGAs	SoCs
Hardware Architecture	**Reconfigurable hardware** and therefore can be highly customized for the intended packet processing	The cores' instruction set and memory architecture is fixed and is therefore less customizable
Programming Model	Hardware description languages (e.g., Verilog) ↓ Harder to program	C-like languages ↓ Easier to program
Performance	Higher throughput lower latency *	Lower throughput higher latency *

* For most kinds of network processing

Side note #1

- Smart NICs can (and do) have fixed-function blocks
- These blocks are optimized hardware implementations of common packet processing functionality.
  - e.g., encryption, hashing, certain common protocols
- A fully ASIC-based NIC can still be considered a "Smart NIC"
  - as long as it supports more complex functionality than a traditional NIC

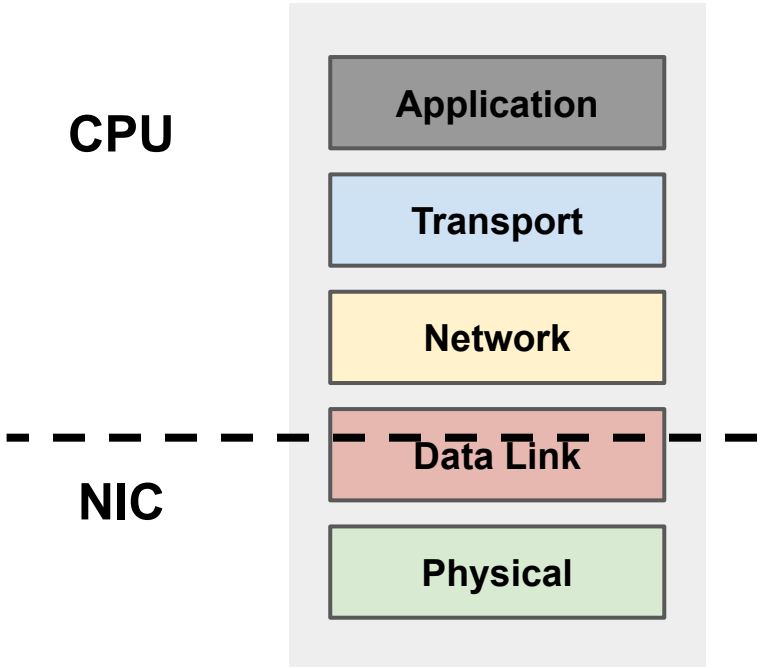
Side note #2

- In industry, Smart NICs have various names
  - Data Processing Unit (DPU)
  - Infrastructure Processing Unit (IPU)
  - …
- They are all conceptually the same.
  - Accelerators of compute and communication at the interface card.

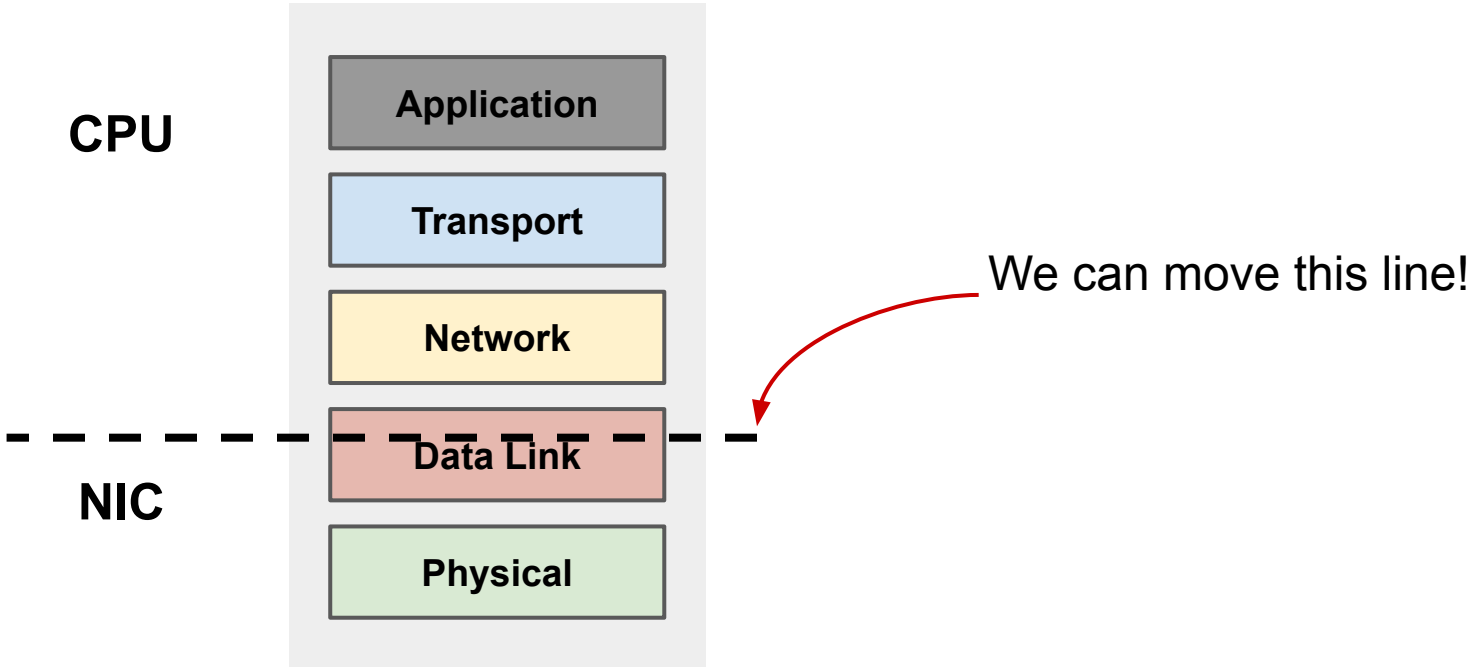
Today's Smart NICs / DPUs/ IPUs /…

- A combination of the following kinds of hardware
  - FPGA
  - SoC
  - P4-programmable pipelines
  - Fixed-function accelerators

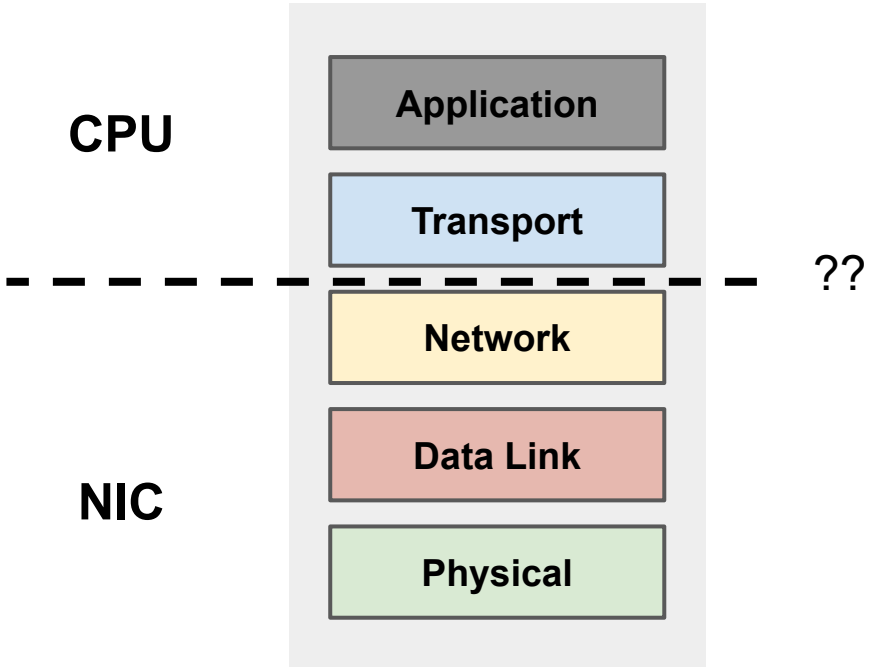
What can we do with our "Smart" NICs?



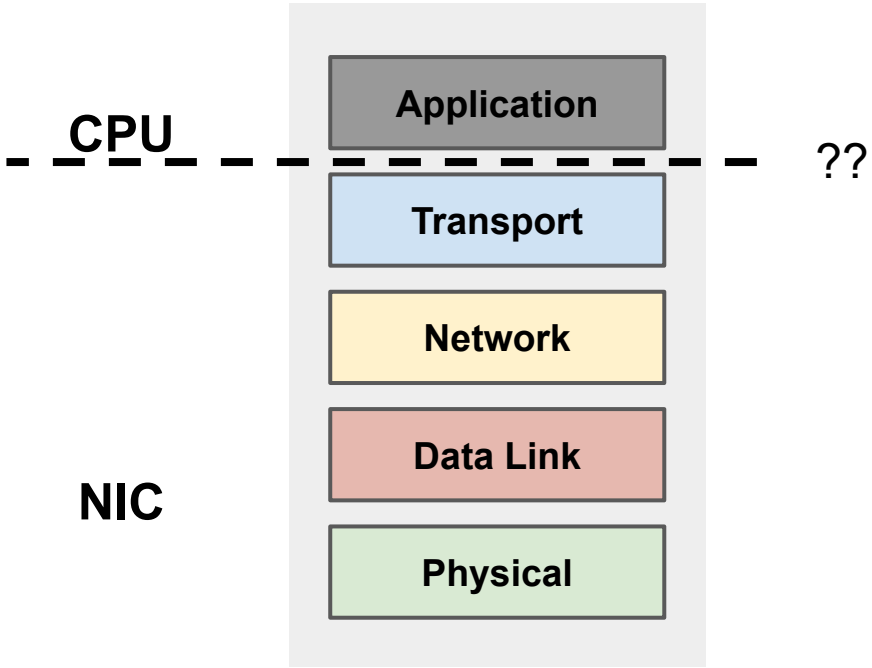
What can we do with our "Smart" NICs?



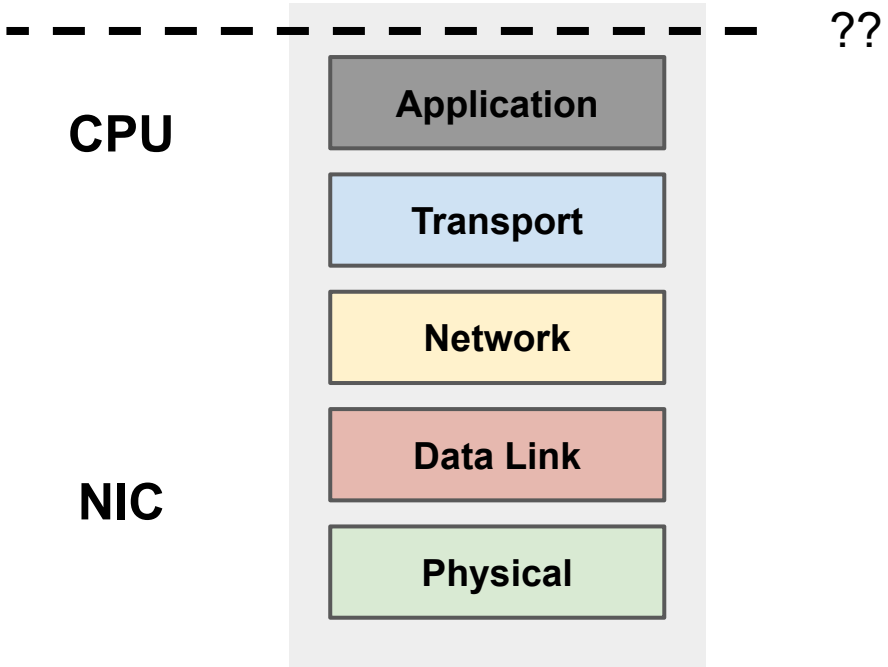
What can we do with our "Smart" NICs?



What can we do with our "Smart" NICs?



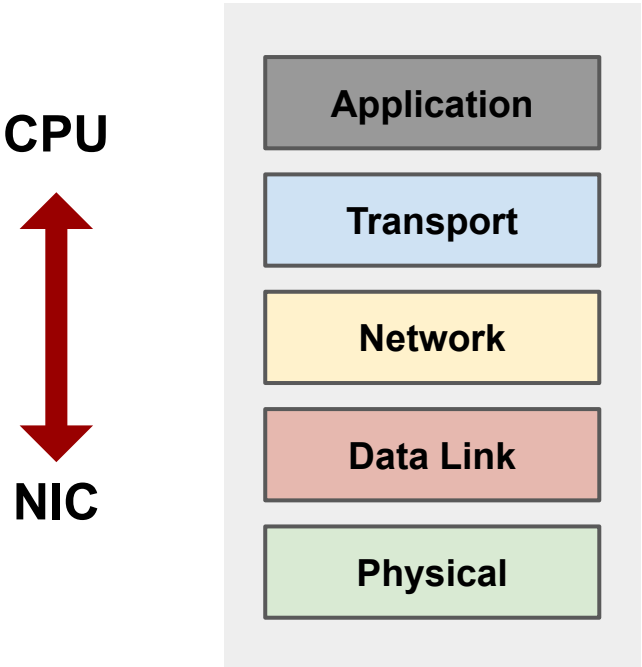
What can we do with our "Smart" NICs?



Offloading the network stack (and beyond!) to the NIC

- Hypervisor vSwitch: AccelNet (NSDI’18)
- Packet scheduling: PIEO (SIGCOMM’19), Loom (NSDI’19)
- Network functions: ClickNP (SIGCOMM’16), FlowBlaze (NSDI’19)
- Transport: Tonic (NSDI’20)
- Even applications: iPipe (SIGCOMM’19), KV-Direct (SOSP’17), Bing web search ranking (ISCA’14)

What can we do with our "Smart" NICs?



We can also optimize the movement of packets between the CPU and the NIC (FlexNIC, ASPLOS'16)

Can we use the same programming model as switches?

- Maybe, but NICs and switches are quite different
- **Speed**: switches have to be faster
  - Switches process traffic for multiple end-points → **Tbps**
  - NICs process traffic for one end point → (10s to 100s of) **Gbps**
- **Functionality:** switches have more limited functionality
  - limited visibility (e.g., don't see both directions of a connection)
  - have to process packets faster.
  - more resource constraints (in contrast, NIC has access to host memory)

Programming abstractions for Smart NICs

- Still an open question!
- There is such a wider range of functionality people can and are interested in implementing on the NICs
- There are many different Smart NIC architectures
    - FPGAs, different kinds of SoCs, P4 pipelines, fixed-function blocks, combinations of these

Programming abstractions for Smart NICs

- Do we keep P4 and rely on architectures and their externs for all the extra functionality?
  - e.g., there are efforts on creating a portable NIC architecture in p4.org
- Or are there more common constructs specific to NIC processing that we can pull out?

Compilation challenges

- Suppose we have a program describing the network processing we want to happen at the end point.
- We can have many different kinds of hardware at our disposal!
  - CPU, all the different hardware on the NIC, even GPUs
- How do we partition/distribute the functionality over these different kinds of hardware? What is the best offloading strategy?

Paper 1: Azure Accelerated Networking: SmartNICs in the Public Cloud

- Published in 2018 by a group in Microsoft Azure
- Microsoft heavily invested in FPGA-based NICs
  - see "A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services" in ISCA 2014.
- This paper describes why they decided to go with FPGAs and how they offload their virtual switch functionality to the NIC.

Paper 2: Offloading Distributed Applications onto SmartNICs using iPipe

- Published in 2019
- Focuses on offloading applications onto SoC-based SmartNICs
- Studies performance characteristics of different multi-core SoC-based Smart NICs for finding good offloading strategies
- Proposes a programming model that could make it easier to program iPipe's target applications onto these NICs.

Additional Resources

- ClickNP (SIGCOMM'16)
  - A programming platform for implementing network functions on FPGAs
- Clara (SOSP'21)
  - Automatically generating offloading insights for SoC-based Smart NICs

Additional Resources