M.Tech Thesis Presentation (EE-798)

Towards an SoC architecture for Software Defined Networking (SDN)

Presenter : Siddhant Singh Tomar (213079010)
Guide : Prof. Madhav P. Desai
Department of Electrical Engineering, Indian Institute of Technology Bombay

OUTLINE

Introduction

Objectives

Custom NIC design.

Baseline Architecture

Performance characterization

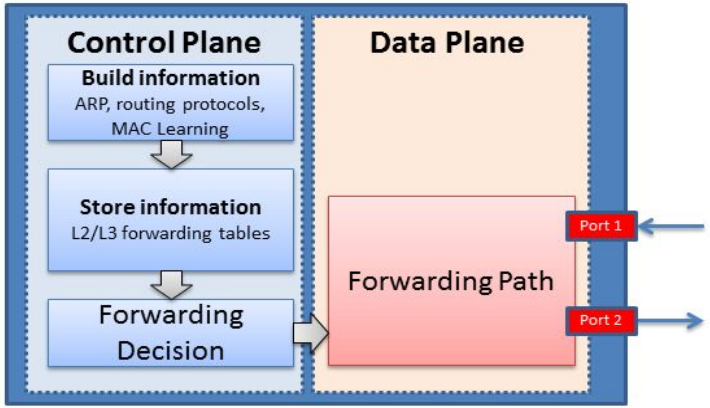
Optimization in Baseline Architecture

Comparative Analysis

Conclusion & Future work

Introduction: Networking Trends

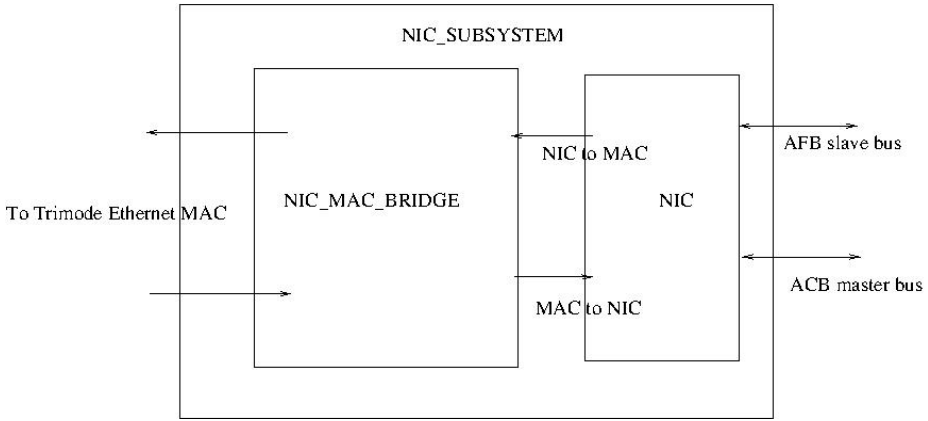
- Software Defined Networking (SDN), modern approach for managing and controlling networks.
- Control and Data traffic are separated for easier management.
- **Need System-on-Chip (SoC) tailored for SDN applications!**



Objective

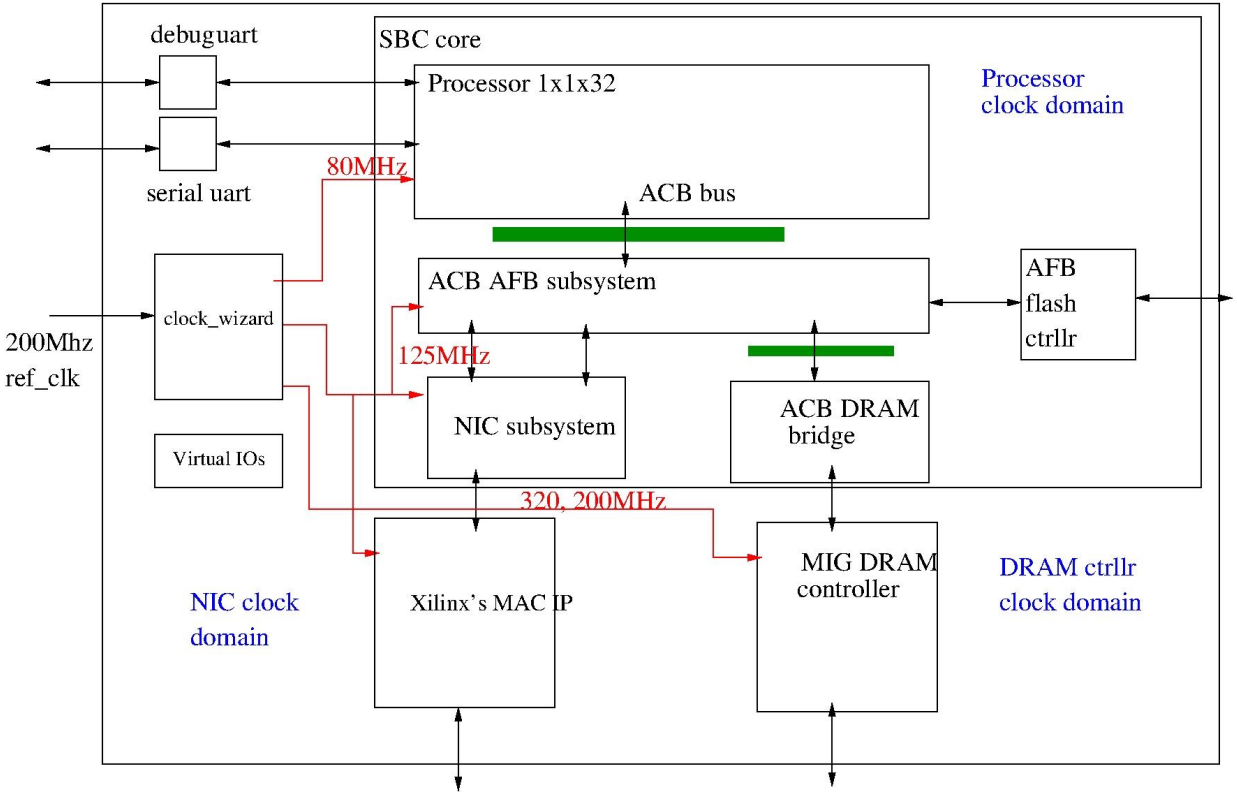
- A network appliances capable of delivering computing capacity, networking capabilities, and storage solutions
- Our objective is to design a System on Chip (SoC) that can encompass all three essential aspects.
- Towards this end, Custom Network interface controller (NIC) was designed by Harshad Ugale.
- Building on his work, the NIC was modified and integrated with a single-threaded 32-bit AJIT processor core and memory subsystem to realize such an SoC.

Custom Network Interface Controller (NIC)



- The NIC operates with Xilinx's tri-mode Ethernet MAC IP to capture Ethernet frames.
- Exposes two interfaces: Slave register interface, Master memory interface.
- NIC makes use of physical addresses to directly access memory for receiving and transmitting the packets.

Baseline architecture : The Single Board Computer (SBC)



Baseline architecture : packet processing

- Four queues are maintained in the main memory: the Receive Free Queue, Transmit Free Queue, Receive Queue, and Transmit Queue.
- Reception: The NIC (HW side) retrieves a free packet buffer pointer from the Receive Free Queue, writes the received packet contents, and pushes the pointer to the Receive Queue, which is polled by the processor (SW side).
- Transmission: The processor (SW side) retrieves a free packet buffer pointer from the Transmit Free Queue, writes the packet contents, and pushes the pointer to the Transmit Queue, which is polled by the NIC (HW side) for transmission.

Baseline architecture : Performance Characterization

- The performance of the SoC was characterized using two applications, namely the standard Ping application and the Network content cache application.
- Both the applications were executed in a bare-metal environment using an open-source TCP/IP stack, lwIP(lightweight IP), which was ported for the AJIT processor and custom-designed NIC.
- **Measurement Setup**:

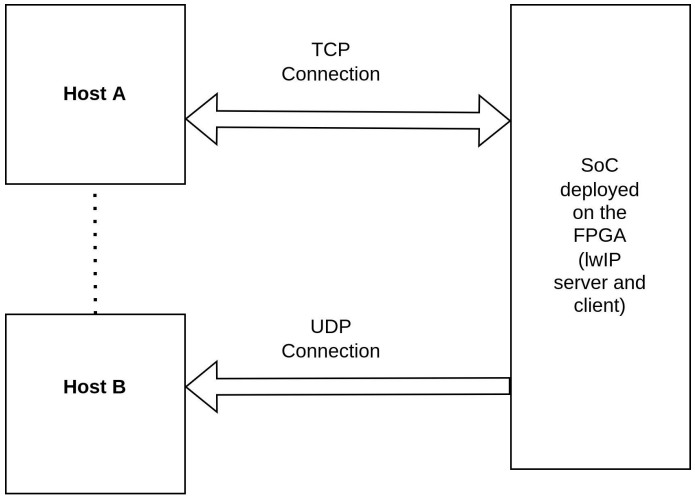
  Processor operating frequency : 80 Mhz
  NIC subsystem operating frequency : 125 MHz
  Ethernet link : 1 Gbps

Baseline architecture : Results (ping application)

The table below presents the average round trip time (RTT) in microseconds (µs) for various packet sizes.

Packet Size (bytes)	Average Round trip time (us)
64	290
128	354
256	388
512	429
1024	449

Baseline architecture : Results (Network Cache)



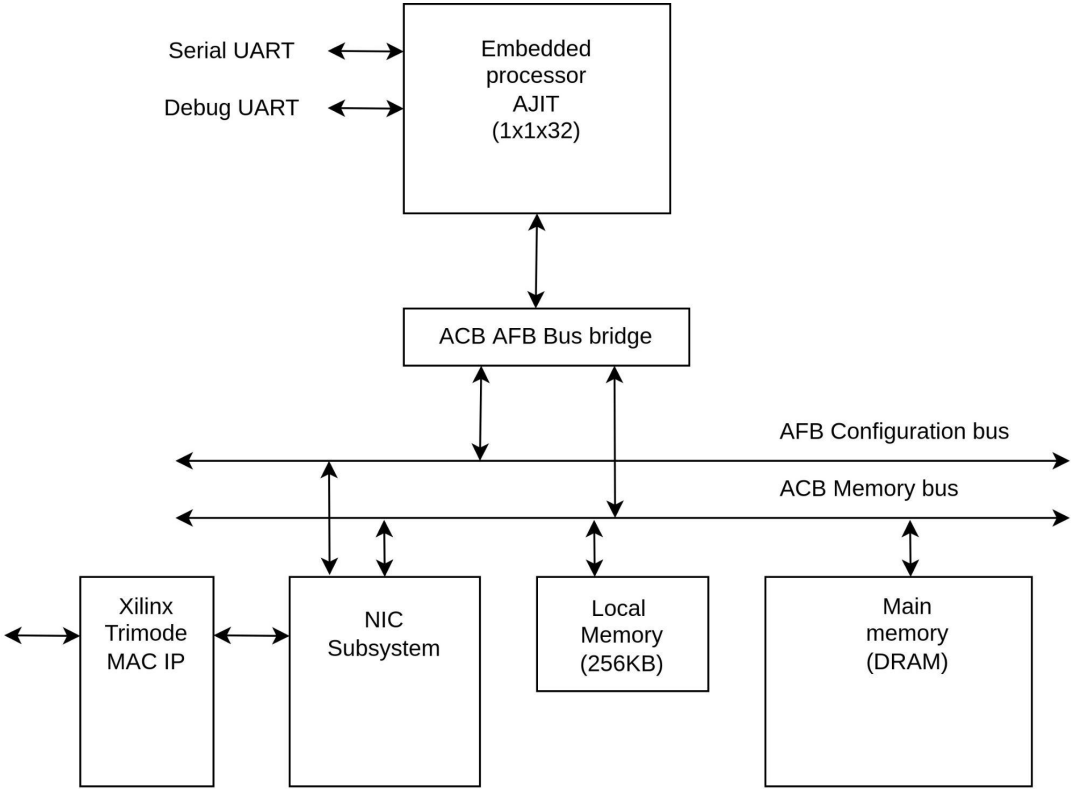
For Network Caching application :

- Host(on PC) to Server (On FPGA) : 23.7 Mbps
- Server (On FPGA) to Host(on PC) : 16.2 Mbps

Optimizations in the baseline architecture

- In our efforts to optimize the baseline architecture, we have focused on enhancing the memory subsystem architecture.
- The first architecture employs a fast local packet memory shared by both the processor and the NIC.
- The second architecture uses an L2 cache in the processor and NIC's access path to the main memory.

Optimization A : Fast local packet memory

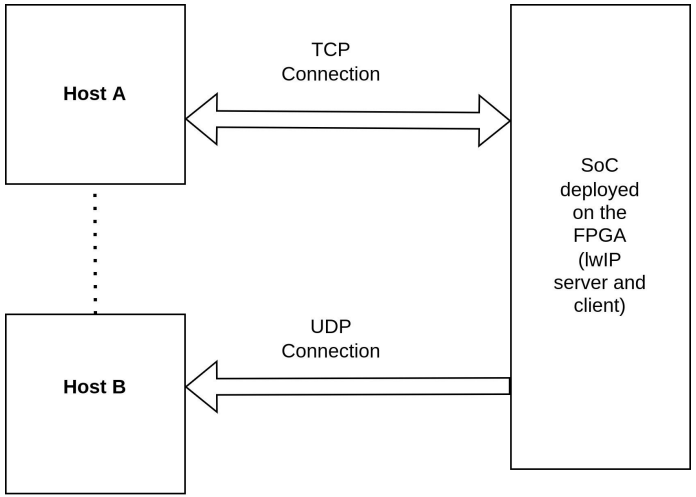


Optimization A : Ping application performance

For Ping application, architecture with fast local memory performs 34.5 % better than the baseline architecture

Packet Size (bytes)	Average Round trip time (us)
64	216
128	261
256	288
512	313
1024	337

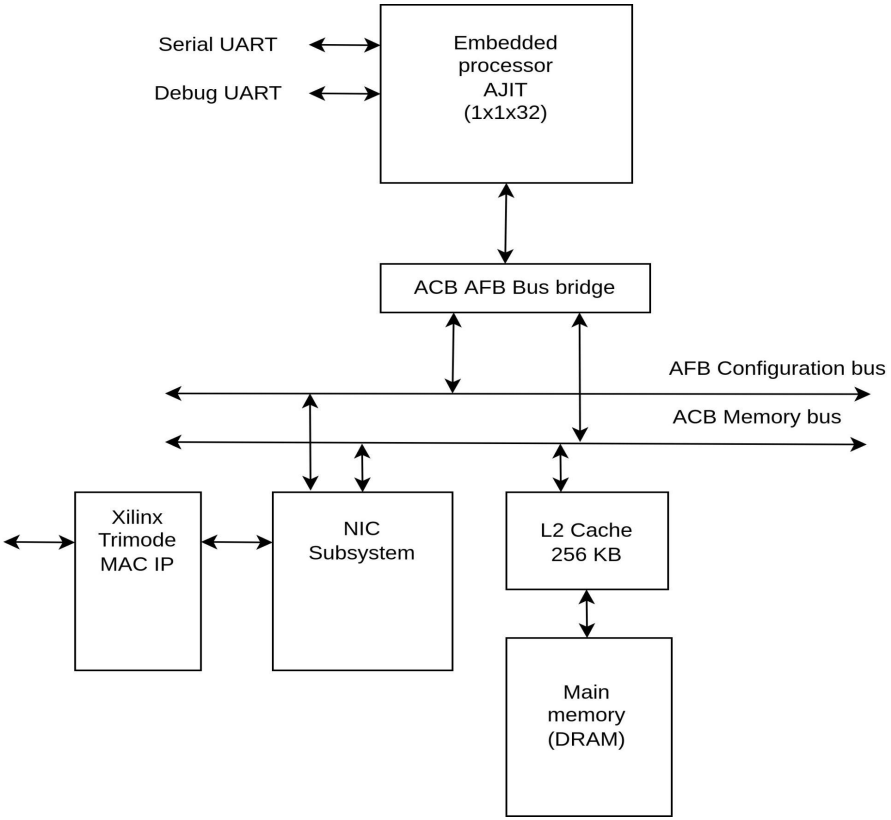
Optimization A: Networking caching application



For Network Caching application, architecture with fast local memory performs 21.1 % better than the baseline architecture

- Host(on PC) to Server (On FPGA) : 28.77 Mbps
- Server (On FPGA) to Host(on PC) : 13.43 Mbps

Optimization B: L-2 Cache in memory access path

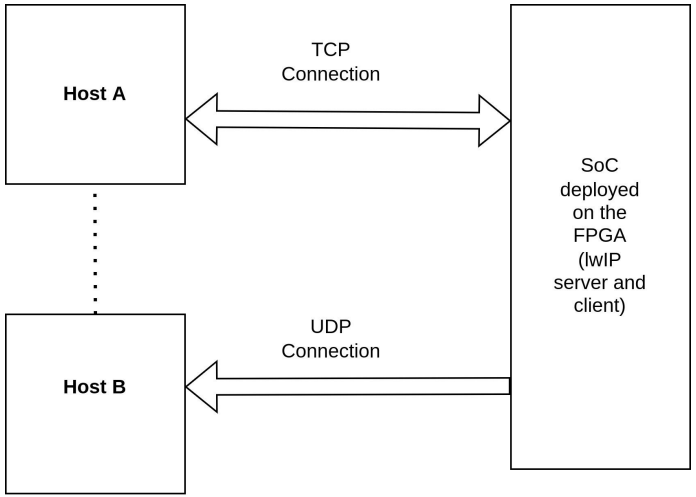


Optimization B: Ping application performance

For Ping application, architecture with L2 cache performs 18.3% better than the baseline architecture

Packet Size (bytes)	Average Round trip time (us)
64	245
128	288
256	322
512	350
1024	371

Optimization B: Networking caching application



For Network Caching application, architecture with L2 cache performs 14.4 % better than the baseline architecture

- Host(on PC) to Server (On FPGA) : 27.12 Mbps
- Server (On FPGA) to Host(on PC) : 14.22 Mbps

Comparative Analysis (ping application)

The table below presents the average round trip time (RTT) in microseconds (µs) for various packet sizes and architectures.

Packets (bytes)	Baseline arch.	**Fast local memory arch.**	L2 Cache arch.
64	290	216	245
128	354	261	288
256	388	288	322
512	429	313	350
1024	449	337	371

Comparative Analysis (Network caching application)

The table below displays the uplink and downlink speeds for network caching applications across three different architectures.

	Baseline arch.	Fast local memory arch.	L2 Cache arch.
Host to FPGA	23.7	28.77	27.12
FPGA to Host	16.22	13.43	14.22

Conclusion

- Both architectures demonstrated performance improvements. Specifically, the first architecture achieved a 34.5% speedup, while the second architecture realized a 18.3% speedup in the considered applications compared to our baseline architecture.
- Therefore, the first architecture is recommended for designing the memory subsystem to ensure sufficient fast buffering for packets in the data plane.

Future work

- Various computational tasks can be further optimized by offloading them to the NIC, such as TCP offloads and checksum generation.
- Adding IPv4 Forwarding information base (FIB) and ARP tables to NIC.
- Enhancing packet handling by directly transferring packets to main memory using DMA instead of the current "memcpy()" function.

THANK YOU!