Gateway Use Case Architecture for Network Applications

P4 Developer Days

Agenda

- The main goal of the presentation is to introduce P4 flow for FPGA chips

1. Gateway Example
2. Introduction into FPGA & RTL development flow
3. FPGA Platform & Shell
4. Conclusion
5. Discussion

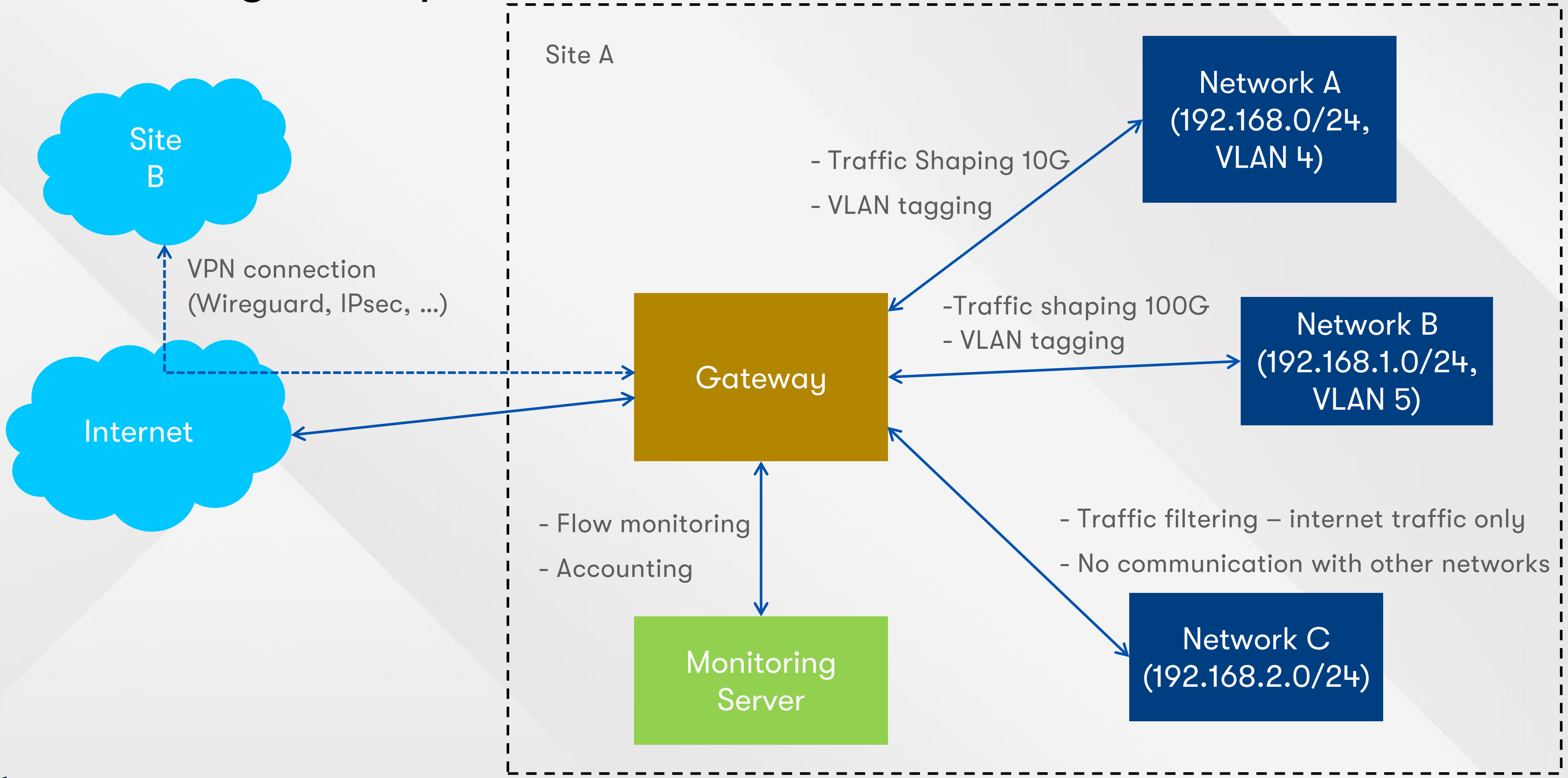
Problem definition

Gateway Example

Gateway Example

- Gateway is a central point for traffic processing – in terms of security, traffic engineering and many other aspects
- Examples:
  - Load balancing – based on various policies (source/destination IP address)
  - Tunnelling – MPLS, VLAN, GRE, IPv6 over IPv4, …
  - Network Address Translation (NAT) – source/destination NAT, masquerade, …
  - Security – Firewall, traffic filtering, lawful inspection, encryption, …
  - Traffic-based accounting
- The gateway can range from simple to very complex high-speed device
- Required functionality may vary through the time as network evolves
  - Flexibility of underlaying HW platform is important – FPGA allows this approach
  - Adding of new features might be a huge benefit – P4 allows the datapath definition
  - FPGA + P4 = a very flexible networking solution capable to hit high-speed packet processing that can evolve in time

Gateway Example


Site A
Site B
VPN connection (Wireguard, IPsec, ...)
Internet
Gateway
- Traffic Shaping 10G
- VLAN tagging
Network A (192.168.0/24, VLAN 4)
-Traffic shaping 100G
- VLAN tagging
Network B (192.168.1.0/24, VLAN 5)
- Traffic filtering – internet traffic only
- No communication with other networks
Network C (192.168.2.0/24)
- Flow monitoring
- Accounting
Monitoring Server

Gateway Example – What is Needed

- Shopping list for the Gateway Example on FPGA
  - An FPGA based platform – NIC equipped with FPGA
    - Like ThunderFjord SmartNIC by Silicom
    - Presented on P4 Developer Days (link)
  - Shell – initial design that allows to use surrounding FPGA infrastructure + allows us to implement the HW accelerated application
    - It is typically bounded to FPGA/delivered with the platform as an example
    - Includes blocks for PCIe, Ethernet, memory controllers for HBM/DDR, and others
  - Tools
    - EDA tool for FPGA (like Quartus) - takes care of translation from RTL to bitstream (~FPGA configuration file)
    - P4 compiler – translates the P4 to language accepted by EDA tool
- We will go through all parts and introduce all aspects so user will be able to understand what development for FPGA is about

Intermezzo

A Short Introduction into FPGA & RTL Development Flow

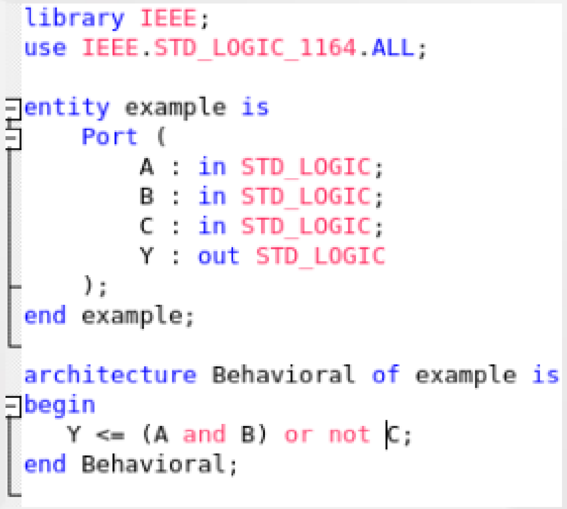
Digital Design

- Chips around us are typically synchronous circuits which consists of several kinds of logic (a very simplified view):
  - Combinational logic – time independent logic
    - Output depends just on input values
    - $Y = f(i_0, i_1, \dots, i_n)$ where Y are outputs and i_x are inputs
    - Example: adders, comparators, multiplexer, demultiplexer, …
  - Sequential logic – history dependent logic (~combinational logic with memory)
    - Output depends input values and history of passed outputs
    - (S, I, O, f, z) where S is set of states, I are inputs, O are outputs, f is function defining new state of the system and z is output function (i.e., generating the output value based on current state and input values).
    - State changes are (typically) happening with rising edge of *Clock signal*
    - Memory element is typically a register/D-type Flip Flop
    - Example: Finite-State Machines, Controllers, Edge detectors, …
  - Clock Domain Crossing circuits (out of scope of the presentation)
    - Serves for data/control signal transfer between multiple clock domains
    - Details out of scope of this presentation

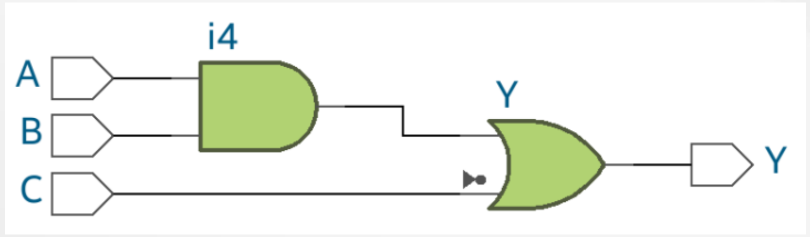
altera

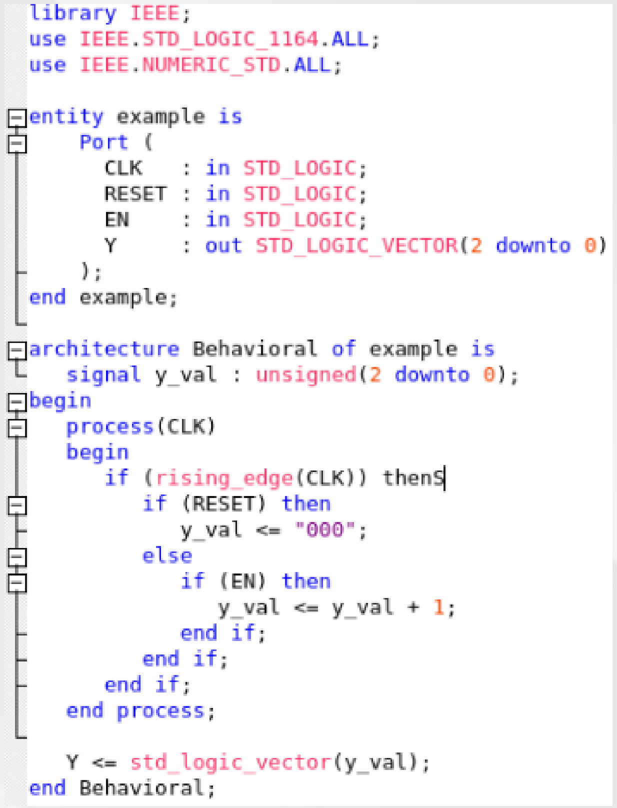
Digital Design

- Sequential and combinational circuits can be described using HDL languages (Hardware Description Language)
  - Examples: VHDL, Verilog/System Verilog
  - Highly parallel programming languages because everything in HW is happening in parallel

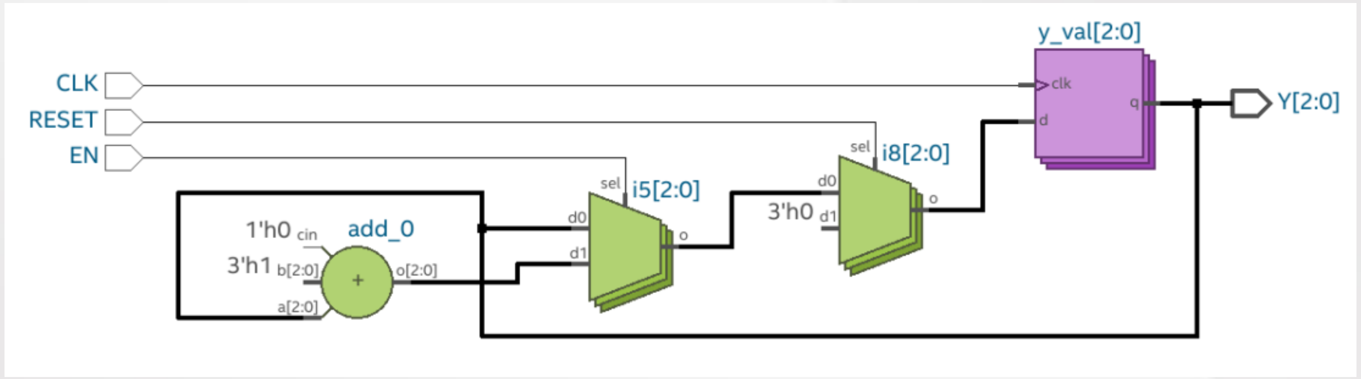


```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity example is
    Port (
        A : in STD_LOGIC;
        B : in STD_LOGIC;
        C : in STD_LOGIC;
        Y : out STD_LOGIC
    );
end example;

architecture Behavioral of example is
begin
   Y <= (A and B) or not C;
end Behavioral;
```



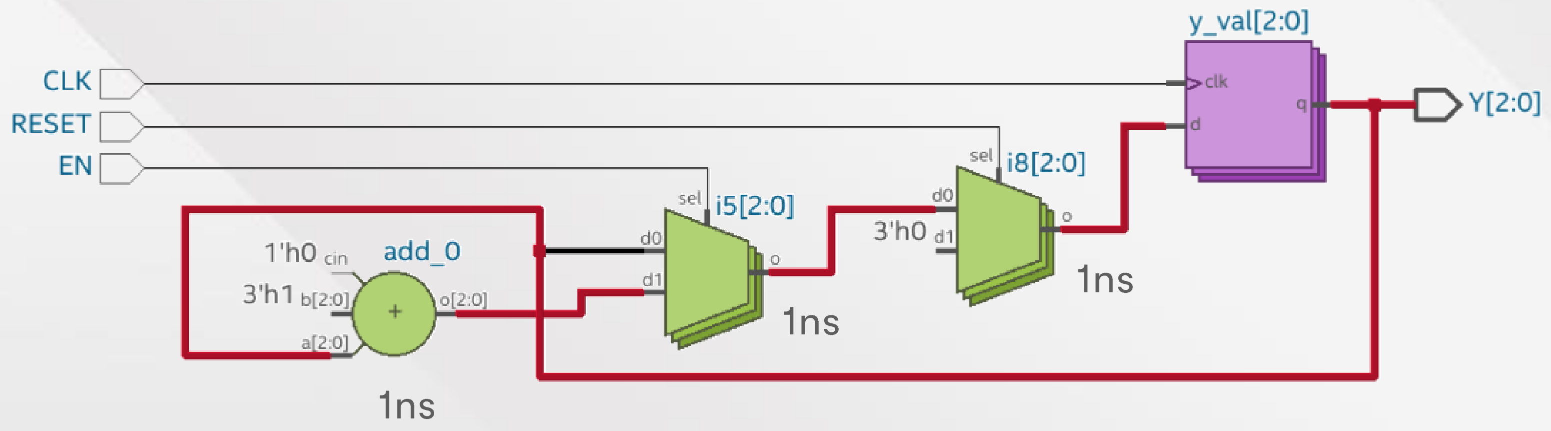


```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity example is
    Port (
      CLK   : in STD_LOGIC;
      RESET : in STD_LOGIC;
      EN    : in STD_LOGIC;
      Y     : out STD_LOGIC_VECTOR(2 downto 0)
    );
end example;

architecture Behavioral of example is
   signal y_val : unsigned(2 downto 0);
begin
   process(CLK)
   begin
      if (rising_edge(CLK)) then$
         if (RESET) then
            y_val <= "000";
         else
            if (EN) then
               y_val <= y_val + 1;
            end if;
         end if;
      end if;
   end process;

   Y <= std_logic_vector(y_val);
end Behavioral;
```



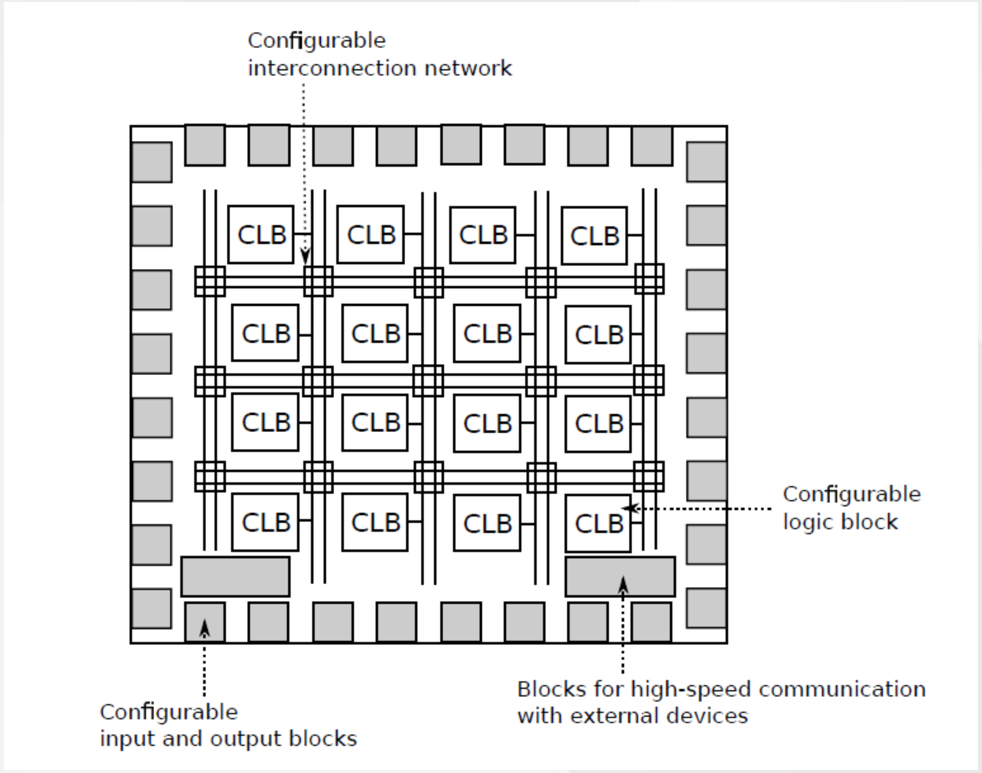
Digital Design

- Clock driven circuits are typically characterized by work frequency
  - Clock frequency depends on critical path in the circuit (i.e., the longest path between two registers)
  - Each element on the path is adding a delay and the delay is cumulating through the path
- If the frequency is higher than the circuit is not working properly
  - Metastability – situation when a digital circuit (typically a register) enters undefined state (e.g., it can be 0 or 1)
- Safe working frequency can be figured out using the Static Timing Analysis (STA)
  - Example (very simplified; some delays on the path ignored) – delay is 3 ns so clock period should be bigger or equal to the value ~ frequency is around 333MHz.



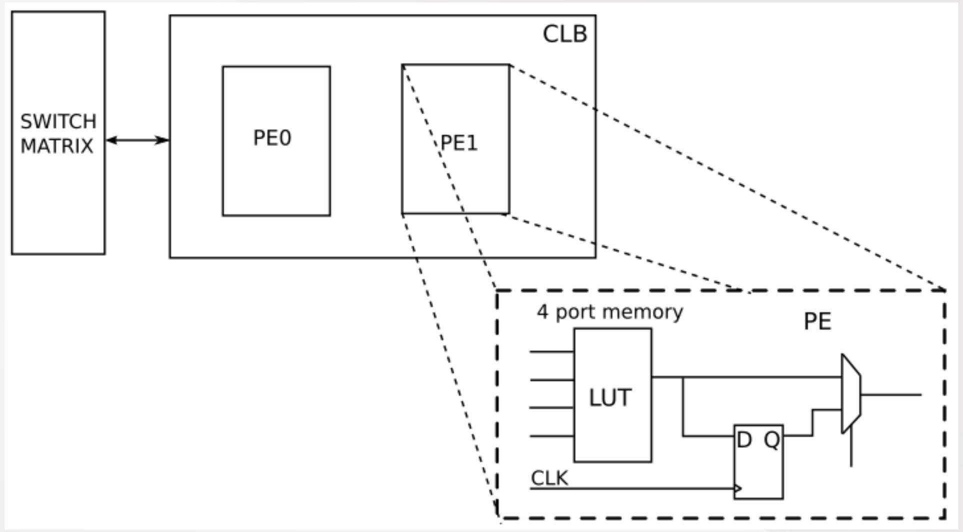
Field Programmable Gate Array (FPGA)

- Integrated circuit that can be programmed after manufacturing
- It is the type of the circuit that can implement sequential and combinatorial logic
- FPGAs are very complex so let's demonstrate on a simplified model
  - CLB – configurable logic block (used for implementation of combinatorial and sequential logic)
  - Configurable interconnection network
    - Used for wiring between all FPGA parts
    - Clock network – low latency network to deliver clocks to CLB and other parts
  - Configurable input/output blocks (pins)
- High-End FPGA's can contain various specialized blocks
  - Ethernet interfaces, PCIe
  - ARM cores, Network-on-Chip
  - Memories & controllers – DDR, HBM, Block Memories


Configurable interconnection network
CLB
Configurable logic block
Configurable input and output blocks
Blocks for high-speed communication with external devices

Field Programmable Gate Array (FPGA)

- Simplified CLB can consist of multiple Programmable Elements (PE):
  - LUT (lookup table – logic function encoded using truth table)
  - D Flip-Flop (used as memory element for implementation of sequential circuits)
  - Multiplexer for switching between register and LUT
- Our example:
  - 4 port LUT memory encodes a truth table with 16 lines and produces one bit output
  - Combinatorial logic path – from LUT to PE output
  - Sequential logic path – from LUT, D Flip-Flop to output
- Switch Matrix connects CLB to configurable interconnection network



Field Programmable Gate Array (FPGA)

- EDA tool (Quartus for Altera FPGAs)
  - Understands the architecture of the chip
  - Compiler that takes RTL code and generates configuration for the chip
- Translation flow
  - User input – circuit description in HDL language (VHDL, Verilog/SystemVerilog)
  - Transformation from the HDL description to Netlist
    - Listing of FPGA components (PE, CLB, …) and how they are connected
  - Technology map
    - Clock tree planning
    - Placement of FPGA components to physical locations on FPGA
    - Routing all parts together
  - Bitstream generation (e.g., configuration for FPGA to match the described behavior)
- EDA Tool is solving NP-complete problem so compilation might take several hours (depending on design complexity, chip occupancy, etc.)

FPGA Platform & Shell

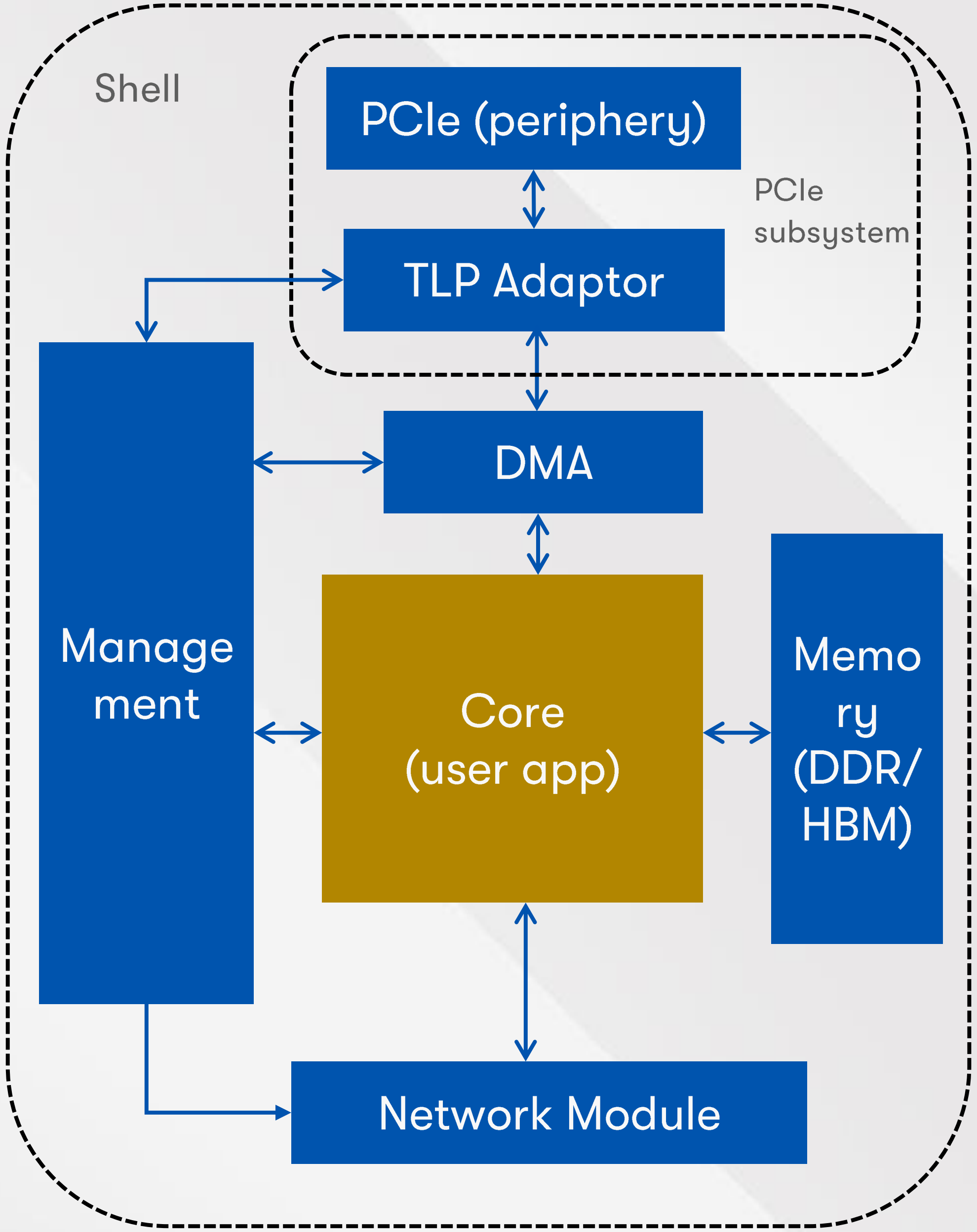
FPGA Platform

- Platform selection depends on many aspects
  - Supported interfaces
  - FPGA chip size
  - Form factor – tabletop/server NIC
  - …
- Example of FPGA platform – Thunderfjord
  - Presented on previous developer day (link)
  - Altera® Agilex M-Series FPGA
  - 2 x 400GE QSFPDD cages
  - PCIe Gen5 x 16
  - SKUs with various memory configurations
    - HBM2E 32GB SKU
    - DDR5 2x16GB SKU

Silicom ThunderFjord - FPGA SmartNIC
FB2CDG1@AGM39D-2 (Image from link)

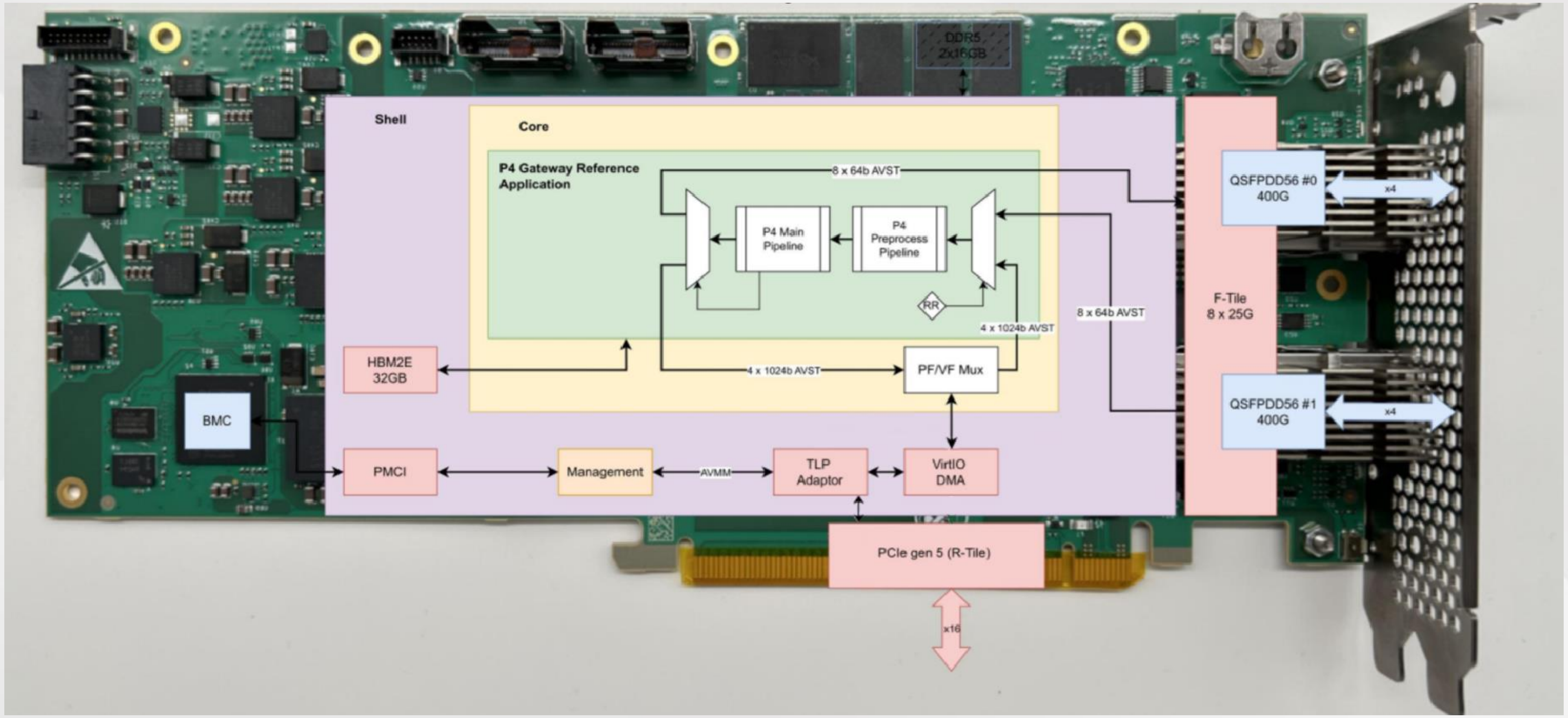
Shell

- A perimeter FPGA design which provides access to peripheries (e.g., Ethernet, PCIe, Memory controllers)
- A typical shell design for SmartNIC:
  - PCIe subsystem – packet transfer from/to SW
  - Management – address decoder, design diagnostic from SW
  - Network module – interface to Ethernet periphery to user application
  - Memory – memory controller that enables external memories to user application
  - DMA - allows high-speed packet transfers from/to user application
  - Core – place for integration of HW accelerated user application (P4 application in this presentation)



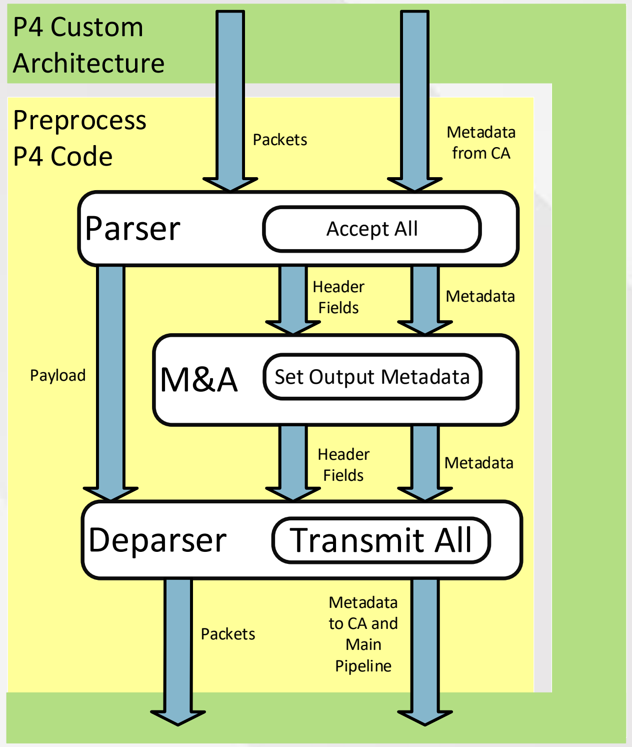
P4 Application

- Common use-case that we did together with Silicom (link)
- 8x25G Ethernet, 4xVirtio DMA
- Worst-case throughput: 29.86Gbps



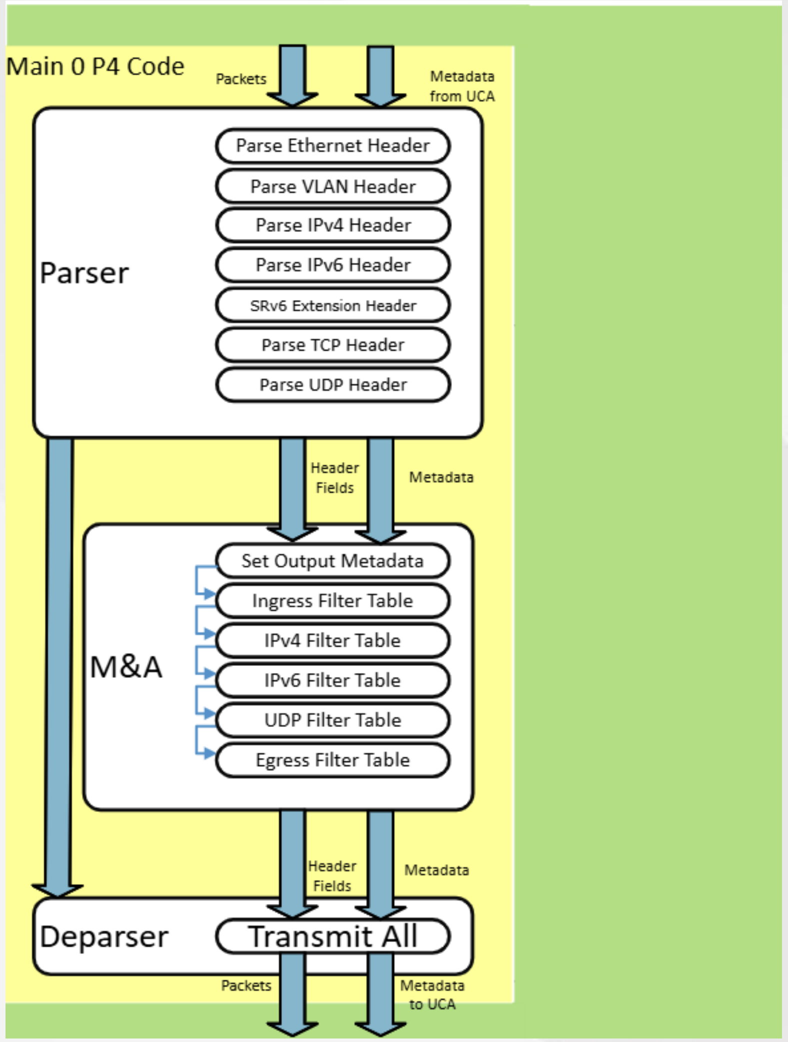
P4 Application – Preprocess Pipeline

- No tables or filtering is applied
- Performs metadata conversion which are required for processing in main pipeline
  - Main pipeline have different metadata types in comparison to metadata from ASAF
- No special header parsing – just passes everything to next P4 pipeline



P4 Application – Main Pipeline

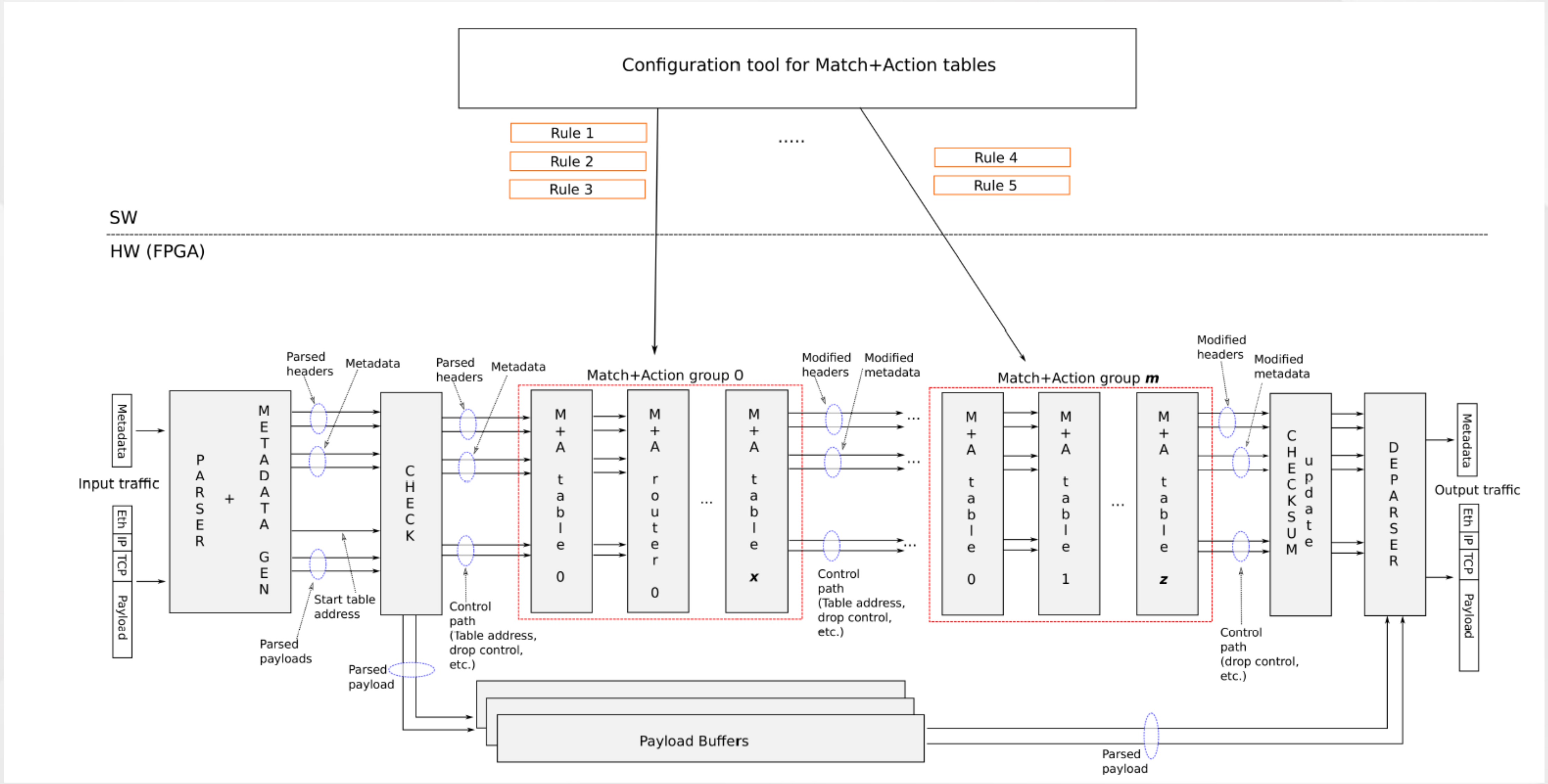
- Performs traffic filtering
  - Ingress filter – input port-based filtering
  - Egress filter – might drop, filter or add/remove VLAN tag
  - IPv4 filter – might change IPs, drop or modify TTL
  - IPv6 filter – same as for IPv4 but for IPv6
  - UDP filter – modify ports or drops traffic
- Actions that can be applied by tables
  - IPv4/IPv6 address changes (source/destination NAT)
  - TLL value update
  - VLAN tagging
  - Packet drop
  - IPv4 checksum update
  - Egress ports selection (Ethernet / DMA)
- Parser supports parsing of Ethernet, VLAN, IPv4, IPv6, TCP/UDP header and up to thee SRv6 headers



P4 Pipeline – What the RTL can look like?

- Altera P4 Suite for FPGA compiler is available under NDA
- I created another compiler proof-of-concept during my Ph.D. so I can use this for demonstration of how the generated pipeline might look like (just for an example)
  - Compiler was implemented for P4 14 using Python HLIR (deprecated)
  - Pipeline throughput up to 100G
  - P4 translated directly to RTL without any High-Level Synthesis layer (like C/C++)
    - Allows finer control over generated code like throughput, frequency
    - Behavior of generated RTL is more predictable
  - Provided with user space driver for rule configuration
  - Text of the thesis available from link and link

P4 Pipeline – What the RTL can look like?



Conclusion

- You should understand what P4 usage on FPGA is about
- P4 language is platform agnostic and FPGA is suitable platform
- Various FPGA platforms – from stand-alone network devices to SmartNIC connected in server
- P4 compiler toolchain available under NDA, contact Altera to get more information

- Benefits:
  - FPGA is very powerful and flexible platform
  - Fast development of new network accelerators using P4
  - Great balance in terms of performance, flexibility and power efficiency

TM