Silicom Denmark AS

FPGA SOLUTIONS

Tailor Made Solutions | Off-the-shelf Products | Innovative Solutions

P4 on Silicom's ThunderFjord SmartNIC

Lars Munch – Senior Software Engineer | Eleftherios Kyriakakis – FPGA Developer

Agenda

- Who we are
- FPGA Framework
- Gateway Reference Application
- Demo-time
- Questions

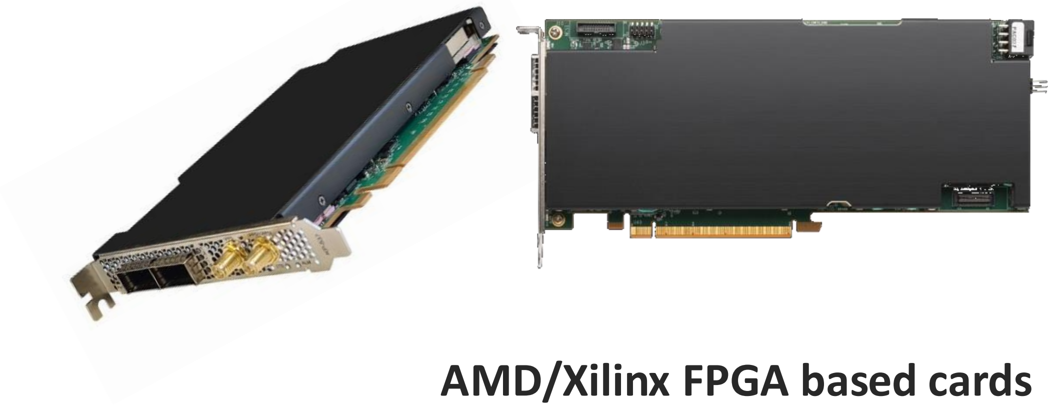
Silicom Denmark A/S

Silicom Ltd. delivers high-performance networking and data infrastructure solutions that enhance server throughput, reduce latency, and support scalable Cloud, NFV, SD-WAN, and cybersecurity systems—built on over 30 years of industry experience

Product ranges include multi-speed server adapters, FPGA-based cards, edge networking devices, x86 open appliances, and intelligent bypass/TAP solutions for both data centers and edge environments

Silicom Denmark A/S is the FPGA competency center focused on developing and producing high-performance, high-quality products for network acceleration, offload, and packet capture

Altera FPGA based cards

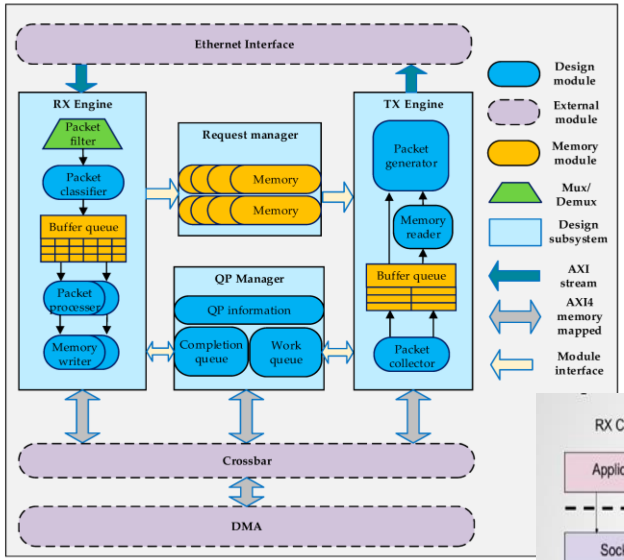


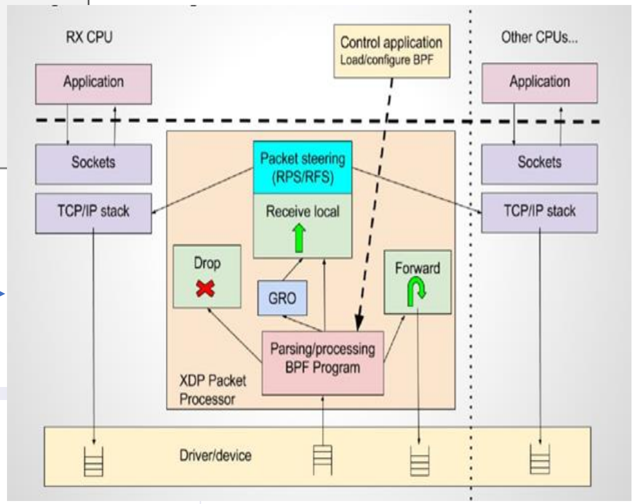
AMD/Xilinx FPGA based cards

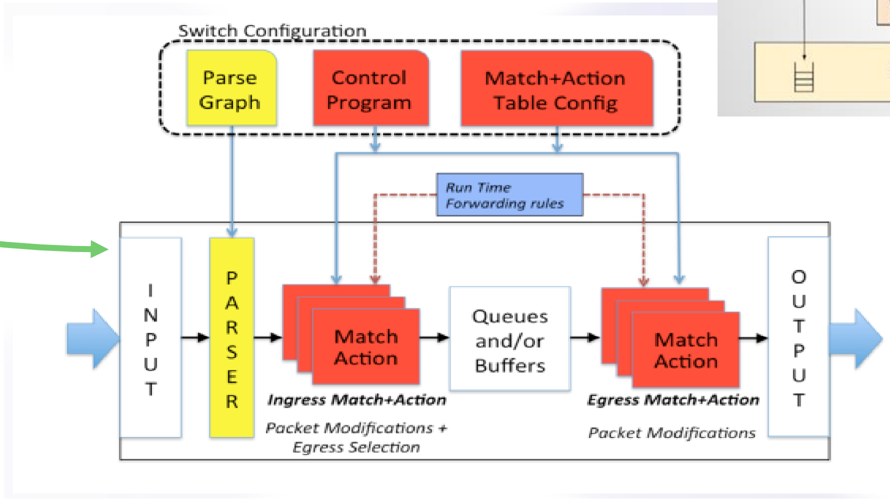
Collaboration with Altera (ex Intel PSG)

Why P4 for NIC Packet Filtering

- In hardware
  - High throughput can be achieved
  - Custom drivers/API
  - Limited by hardware capabilities
- In software
  - Standardized framework (DPDK, XDP/eBPF)
  - Flexible reconfiguration
  - Difficult to achieve high throughput
- P4 bridges the gap
  - Especially when combined w/ FPGA
  - Easily reconfigurable data & control planes
  - Standard API Table Driven Interface (TDI)
  - Fantastic for inline network applications on SmartNICs







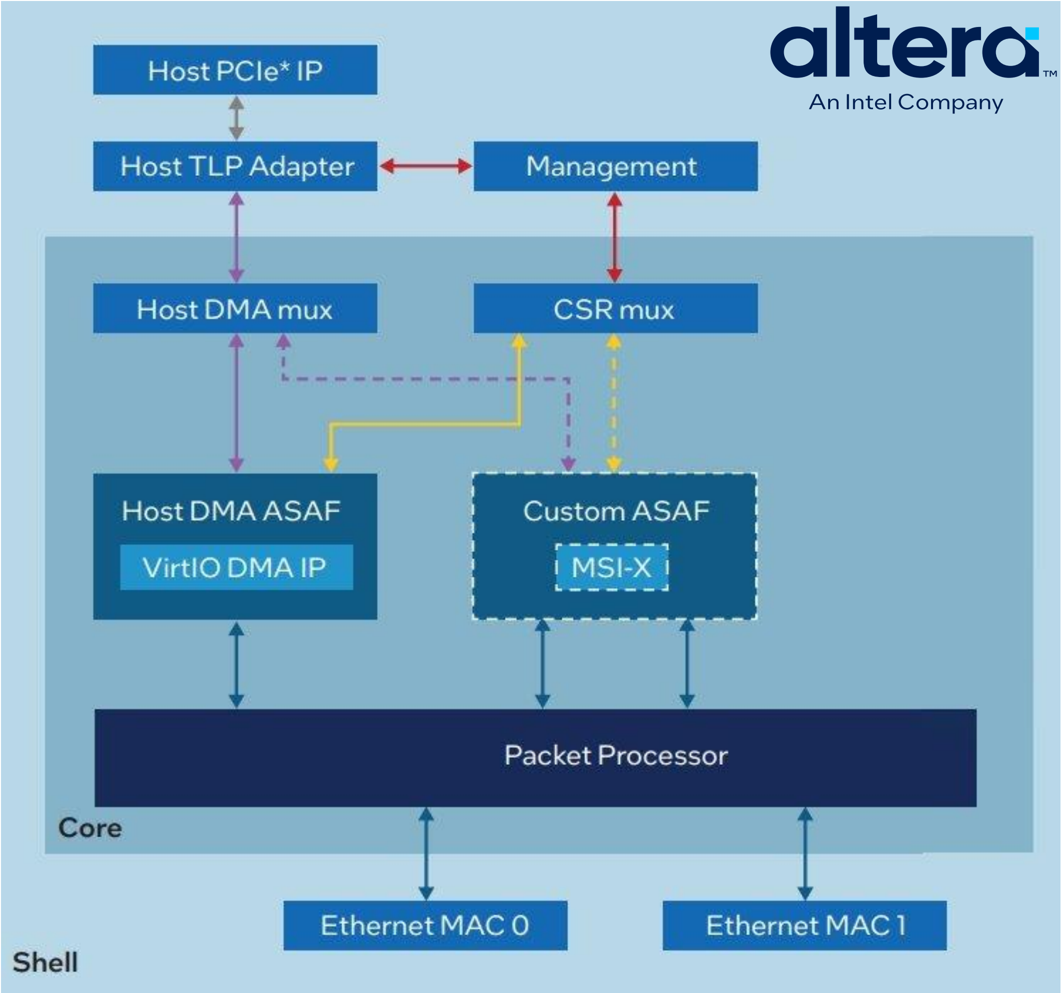
FPGA Framework

ThunderFjord SmartNIC by Silicom

- Altera® Agilex™ M-Series FPGA
- 2 x 400GE QSFPDD56
- PCIe Gen5 x16 and CXL 2.0 Support
- Flexible ARF6 Connectors with 16x28Gbps
- SKUs with various memory configurations
  - HBM2E 32GB SKU
  - DDR5 2x16GB SKU

ASAF – Application Stack Acceleration Framework

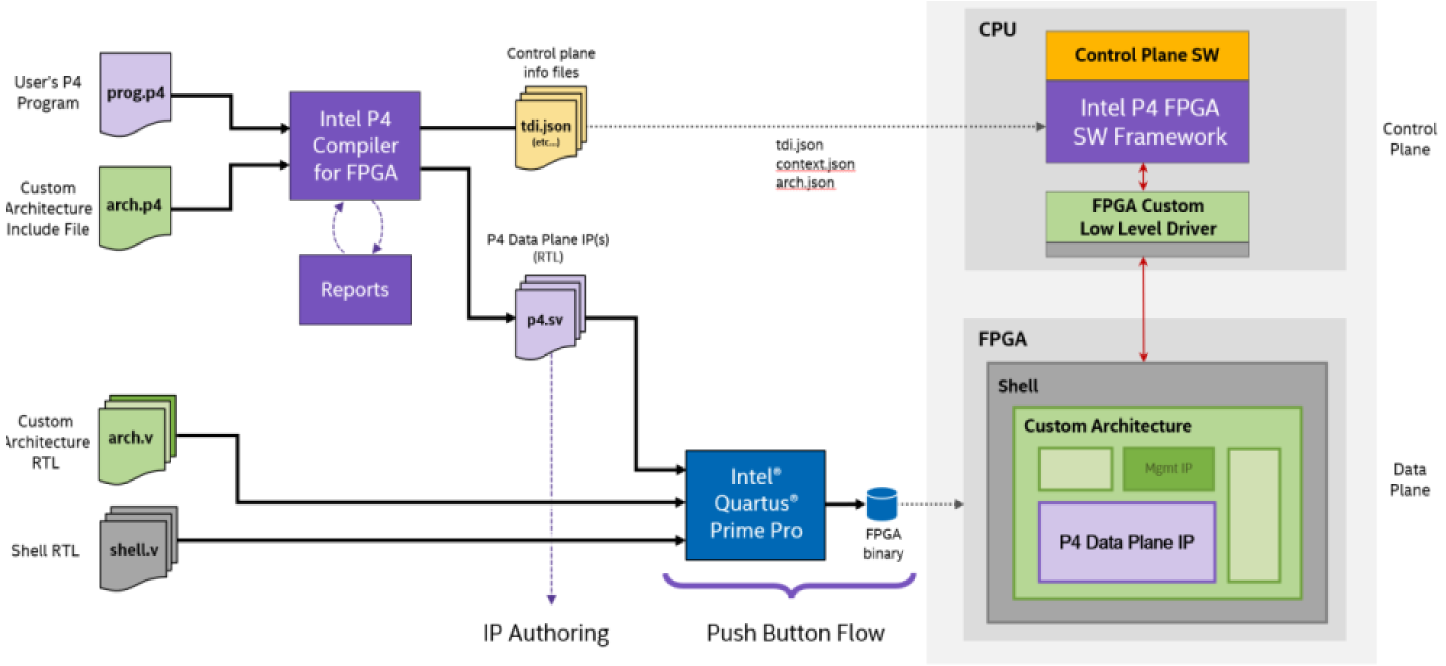
- Concept separating the FPGA design into:
  - Base NIC functionality (PCIe, Ethernet, BMC)
  - Application stack functionality (packet processor/user logic)
- VirtIO DMA provides plug-n-play Linux driver and DPDK support
- Open-source similarities:
  - OFS Intel
  - NDK by CESNET



**Access only to selected partners*

Intel P4 Build Flow

- P4 Compiler
  - Generate RTL ready for integration within custom architecture
  - Interfaces to architecture owned IPs
- P4 FPGA software framework (FSF)
  - Accessing the TDI using standard TDI C++/C and Python interfaces using: https://github.com/p4lang/tdi
  - Interactive Command Line (CLI) for easy TDI modifications
- End-user needs:
  - JSON describes the register PCIe BAR addresses, and the tables present in the P4 architecture
  - FSF for the driver access to CSR and P4 TDI

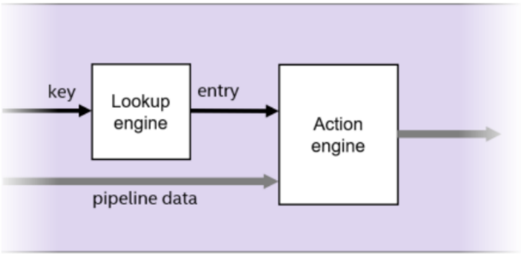


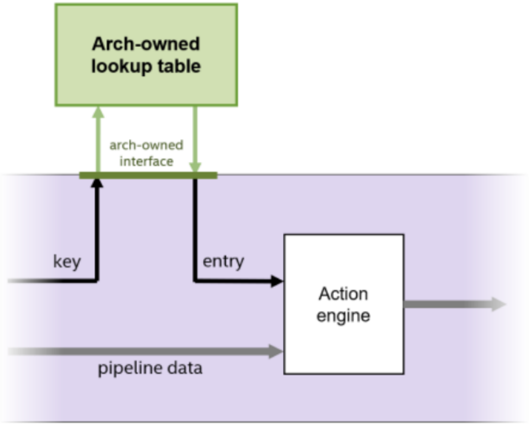
P4 Programmer	altera (An Intel Company) or Silicom Denmark AS FPGA SOLUTIONS or customer
P4 Custom Architecture	altera (An Intel Company) & Silicom Denmark AS FPGA SOLUTIONS
Card & Shell Provider	Silicom Denmark AS FPGA SOLUTIONS
P4 Control Plane Rules by customer	

P4 Compiler Extensions

Architecture-owned lookup tables (AOLUT)

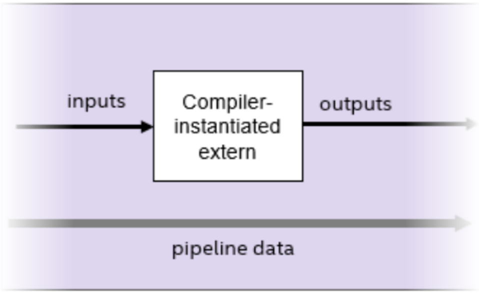
```
@Intel_BindTo[arch_owned_iface="aot_if"]
table big_acl {
    key = {
        ipv6_src_addr: exact;
    }
    actions = {
        Drop;
        Allow;
    }
    size = 300000;
    const default_action = Drop;
}
```

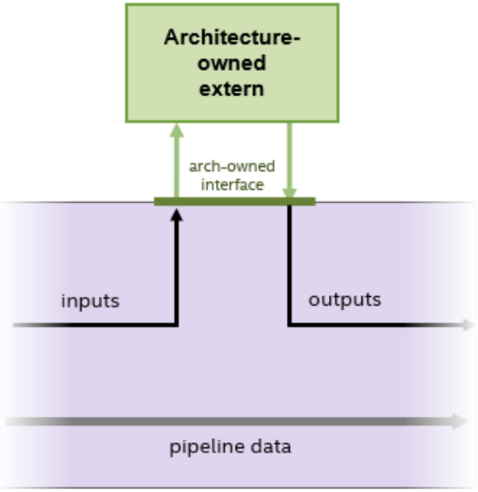




Architecture-owned externals (AOE)

```
// (within the body of an action or match-action pipeline)

@Intel_BindTo[arch_owned_iface="aoe_if"]
ArchOwnedExtern_t() aoe_inst;

// ...
bit<32> z = aoe_inst.invoke(x, y);
// ...
```

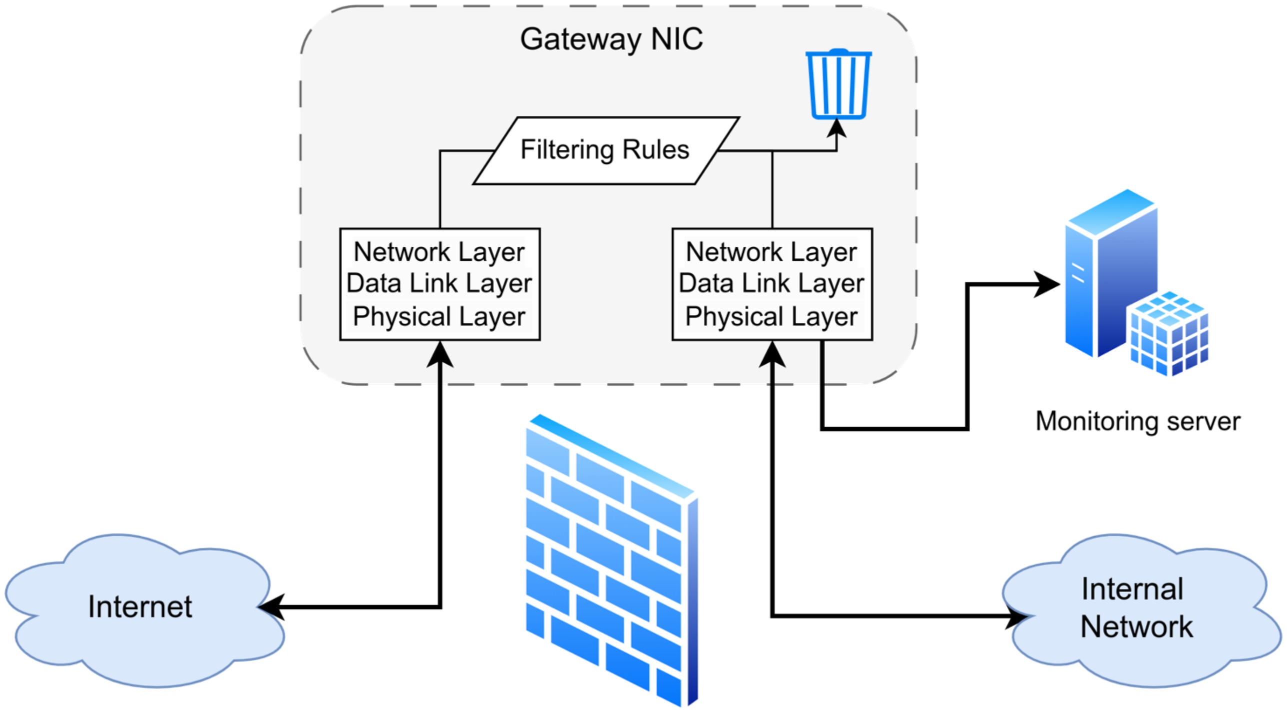




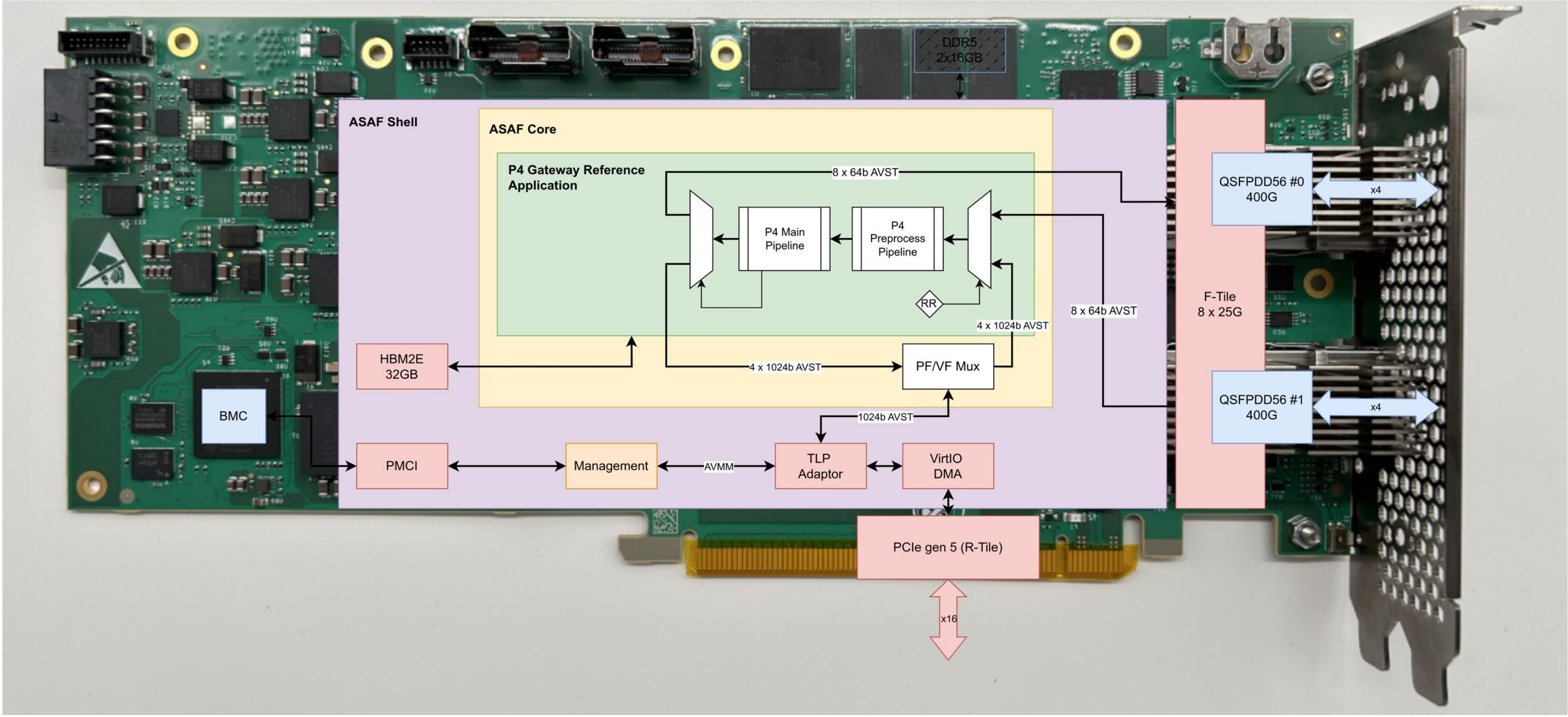
Gateway Reference Application

Gateway NIC Example

- Central point for controlling network traffic access
  - Security, monitoring, lawful interception
  - Traffic allowed / blocked / re-directed
- P4
  - Re-deployment of Gateway NIC
  - Runtime traffic re-shaping
- ThunderFjord SmartNIC
  - High-speed throughput of 400G
  - Expandable connections: 8x25G + 16x25G

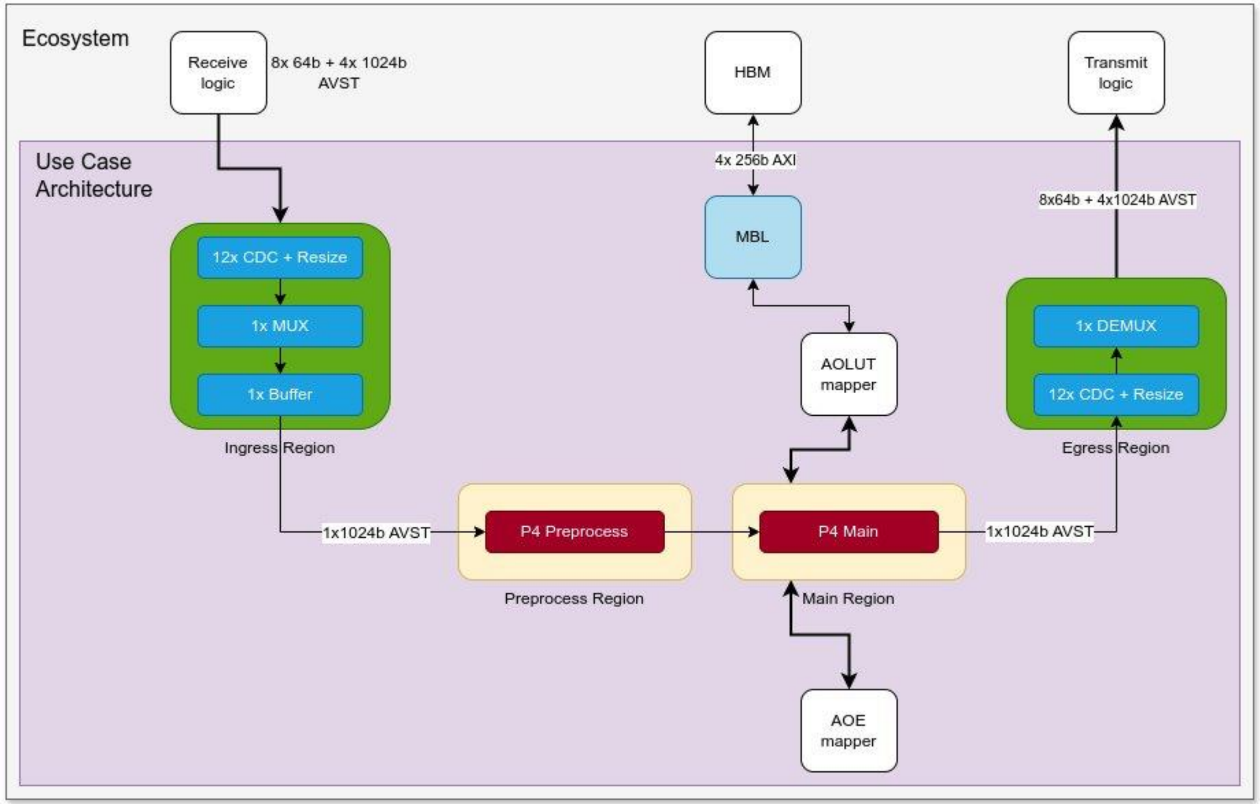


ThunderFjord P4 Use-Case Architecture



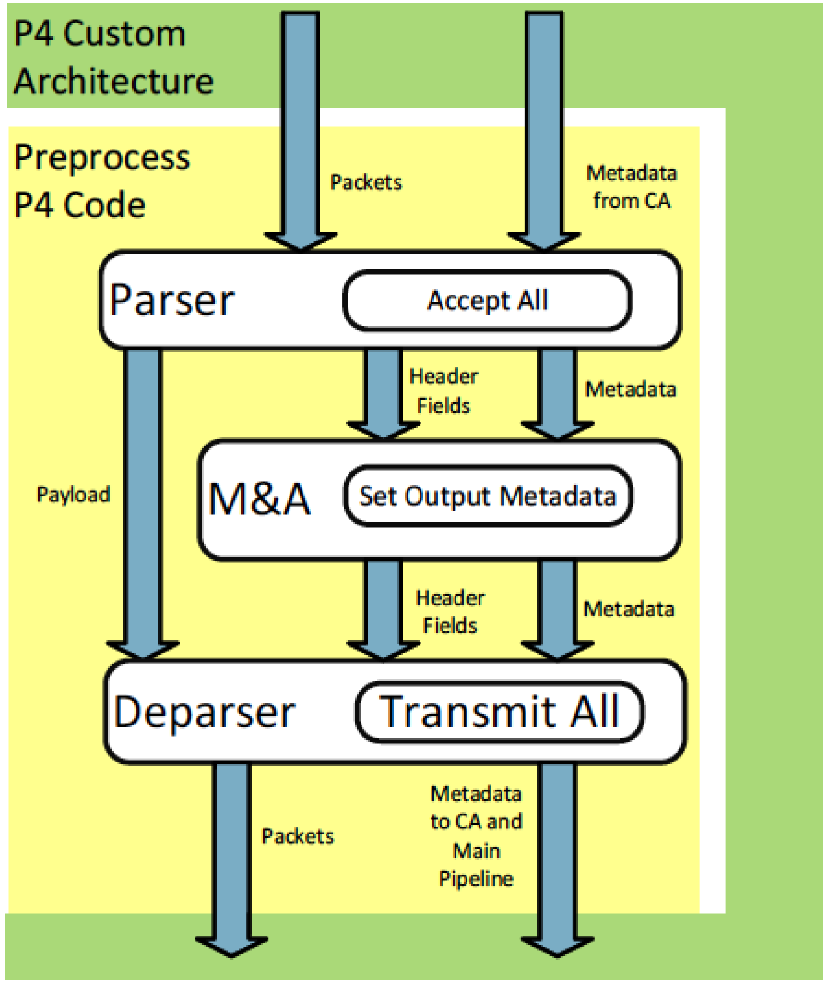
P4 Gateway Reference Application

- Configured to support 12 ports
  - Eight ports mapped to two QSFP
  - Four ports mapped to four VirtIO VFs
- Packets from host are decapsulated from DMA header/metadata in PF/VF Mux
- High-speed content lookup through Multi-Bin Lookup (MBL) IP hosted in HBM memory
- Worst-case throughput: 29.86 Gbps



Pre-process pipeline

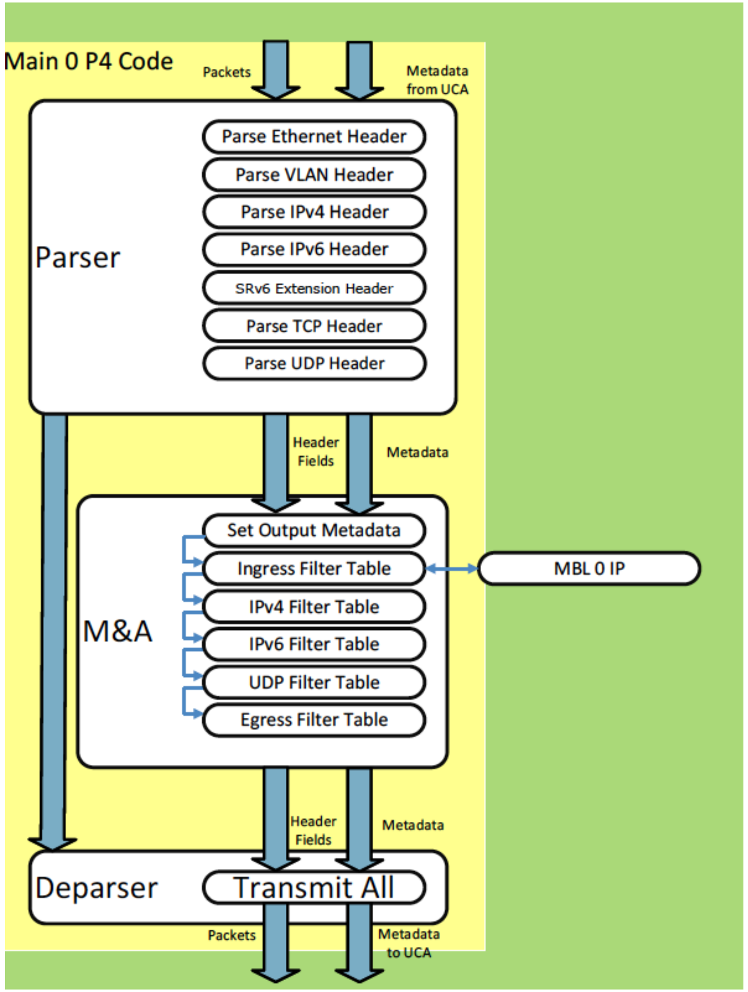
- No tables or filtering
- Connecting the ports to P4
- Responsible for converting the metadata from the ingress to the required of the metadata of the main pipeline
- Computes the packet length
- Accepts all headers



Silicom Denmark AS FPGA SOLUTIONS

Main Pipeline

- Can parse the headers: Ethernet, VLAN, IPv4/6, UDP, TCP, SRv6 (up to three elements of the segment list)
- Network actions:
  - IPv4/IPv6 address modification.
  - TTL update
  - SRv6 Endpoint Node behavior
  - Calculating Checksum values
  - VLAN tagging
  - Changing L4 UDP port values
- The ingress filter table is mapped as AOLUT through MBL to HBM memory



P4 Main - parser

```
// main.p4
Main_Pipeline (
    Main_Parser(),
    Main_MA(),
    Main_Deparser()
) pipe_main;

// headers.p4
header ethernet_h {
    bit<48> dstAddr;
    bit<48> srcAddr;
    bit<16> etherType;
}

header vlan_h {
    bit<3> pcp;
    bit<1> cfi;
    bit<12> vid;
    bit<16> etherType;
}
...
```

```
// parser_main.p4
#include "gateway_arch.p4

parser Main_Parser (
    packet_in                                   pkt,
    out headers_main_t                          hdr,
    out metadata_main_t                         md,
    inout preprocess_main_intrinsic_metadata_t in_md_intr)
{
    state start {
        transition parse_ethernet;
    }

    state parse_ethernet {
        pkt.extract(hdr.ethernet);
        md.etherType = hdr.ethernet.etherType;
        transition select(hdr.ethernet.etherType) {
            L2_PROTO_VLAN  : parse_vlan;
            L2_PROTO_IPV4  : parse_ipv4;
            L2_PROTO_IPV6  : parse_ipv6;
            default         : accept;
        }
    }
    ...
}
```

P4 Main - control block

```
// VLAN tag handling
bit<32> packet_len = get_packet_length();
action set_packet_length() {
    out_md_intr.common.packet_length =
        (bit<16>) packet_len;
}

action update_vlan(bit<12> vid) {
    if (!hdr.vlan.isValid()){
        hdr.vlan.setValid();
        hdr.vlan.etherType = hdr.ethernet.etherType;
        hdr.ethernet.etherType = 0x8100;
        hdr.vlan.pcp = 0;
        hdr.vlan.cfi = 0;
        packet_len = packet_len + 4;
    }
    hdr.vlan.vid = vid;
}

action strip_vlan() {
    hdr.ethernet.etherType = hdr.vlan.etherType;
    hdr.vlan.setInvalid();
    packet_len = packet_len - 4;
}
```

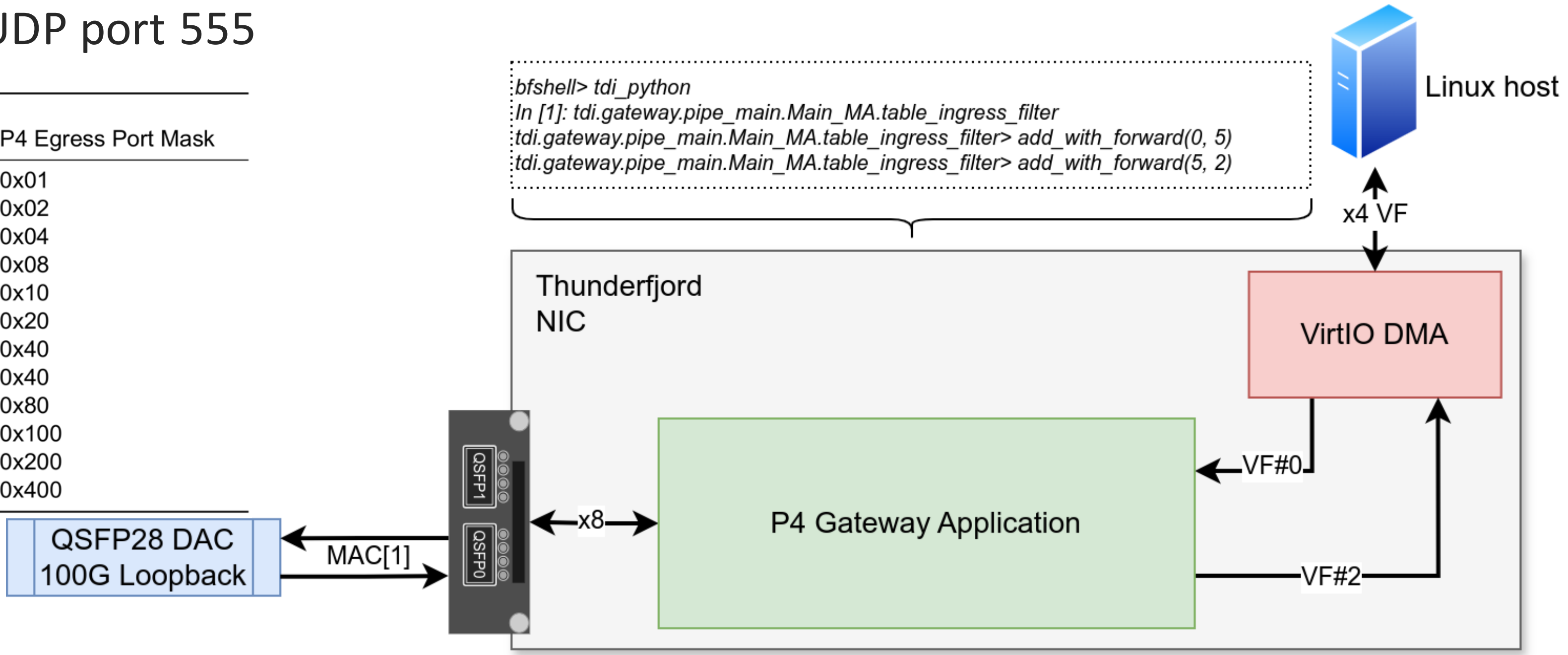
```
// Packet modification and checksum update
InternetChecksum() csum;

action mod_l4_srcPort(bit<16> srcPort) {
    if (hdr.udp.isValid()){
        csum.initialize(hdr.udp.checksum);
        csum.subtract(hdr.udp.srcPort);
        hdr.udp.srcPort = srcPort;
        csum.add(hdr.udp.srcPort);
        hdr.udp.checksum = csum.get();
    }
}

// Architecture owned lookup table
@Intel_BindTo[arch_owned_iface="mbl_0_if"]
table table_ingress_filter {
    key = {
        in_md_intr.common.ingress_port : exact;
    }
    actions = {
        forward;
        NoAction;
        drop;
    }
    size = 8;
}
```

Demo Time

P4 Gateway Demo on ThunderFjord

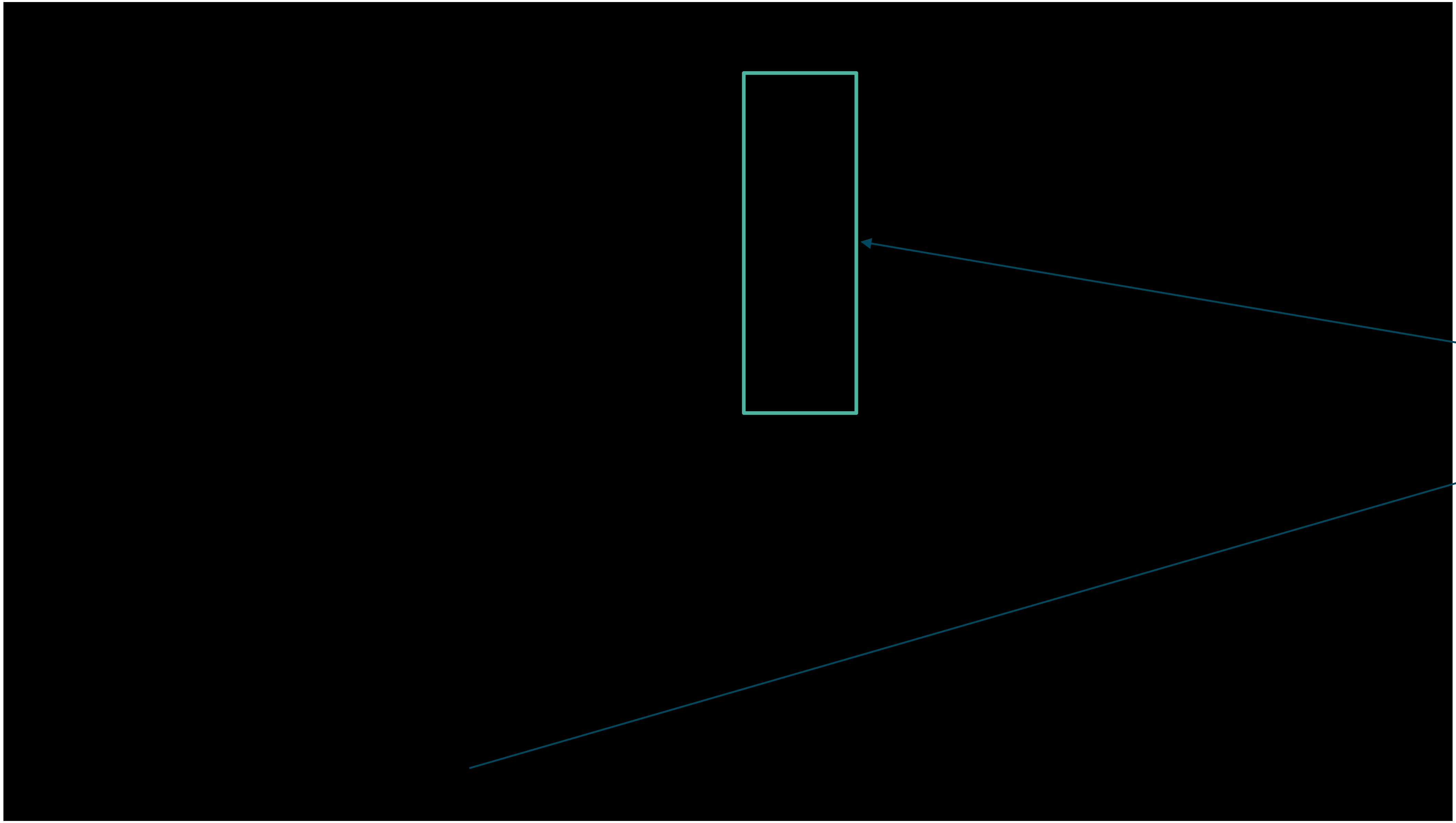
- Thunderfjord (TFJ) hosted on a standard Linux server, configured with Intel's FSF
- Wireshark monitors traffic through SSH on VF#2
- TCPREPLAY applies UDP traffic from port 160 to port 176 on VF#0
- Forwarding VF#0 to egress MAC[1] (P4 port 5)
- Forwarding ingress MAC[1] (P4 port 5) to VF#2
- Intercepted traffic is forwarded to UDP port 555

P4 Pipeline	Interface	Port (MAC)	P4 Ingress Port	P4 Egress Port Mask
0	VirtIO: VF0 on PF0	0	0	0x01
0	VirtIO: VF1 on PF0	1	1	0x02
0	VirtIO: VF2 on PF0	2	2	0x04
0	VirtIO: VF3 on PF0	3	3	0x08
0	QSFPDD 0	0	4	0x10
0	QSFPDD 0	1	5	0x20
0	QSFPDD 0	2	6	0x40
0	QSFPDD 0	3	7	0x40
0	QSFPDD 1	4	8	0x80
0	QSFPDD 1	5	9	0x100
0	QSFPDD 1	6	10	0x200
0	QSFPDD 1	7	11	0x400



Configuring P4 rules on Thunderfjord from TDI CLI

Monitoring forwarded/modified UDP traffic on VF#2



UDP ports modified at runtime through TDI CLI

Silicom Denmark AS

FPGA Solutions

Questions ?

P4 on Silicom's ThunderFjord NIC