Programmable Data Plane at Terabit Speeds

Vladimir Gurevich

May 16, 2017

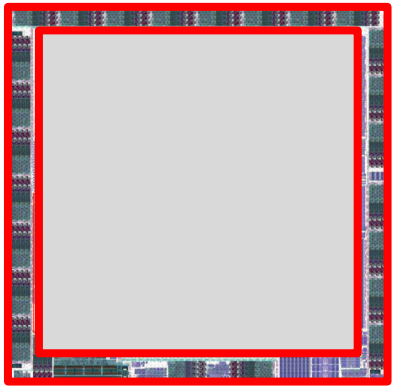
“Programmable switches are 10-100x slower than fixed-function switches. They cost more and consume more power.”

Conventional wisdom in networking

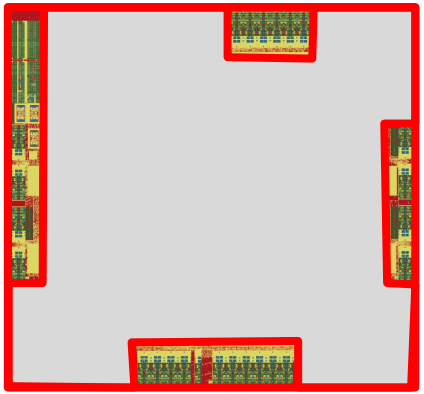
BAREFOOT NETWORKS

Moore’s Law and Simple Math

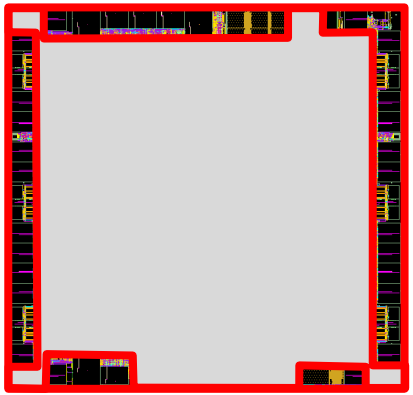
Serial I/O: About 30% of switch chip area



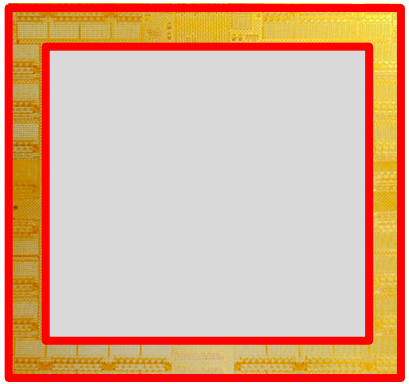
Intel Alta (2011)



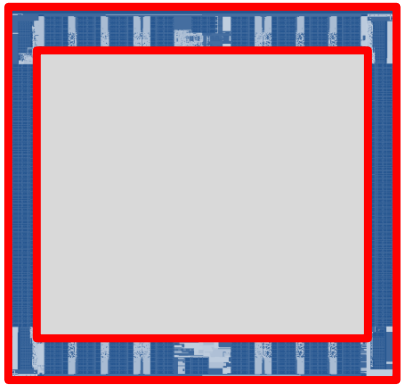
Cisco (2011)



Ericsson Spider (2011)

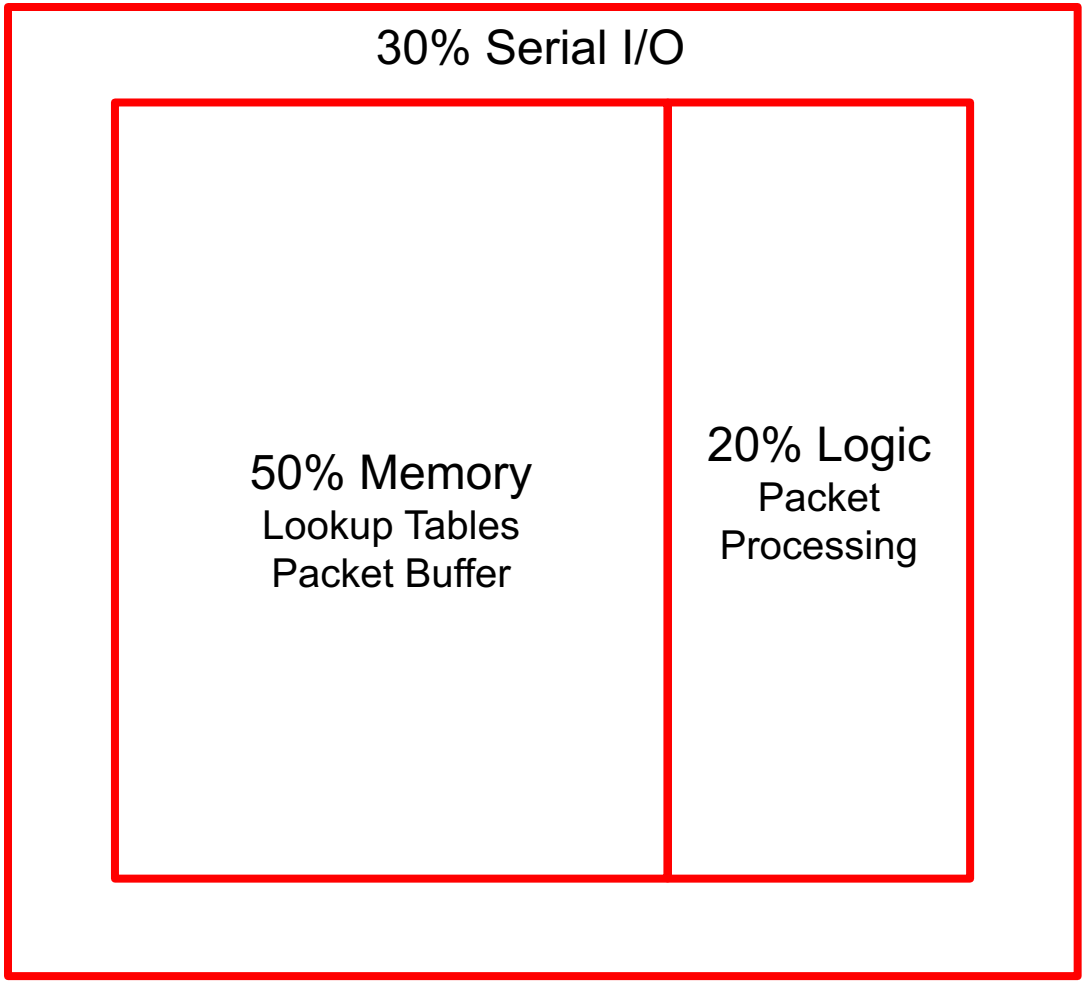


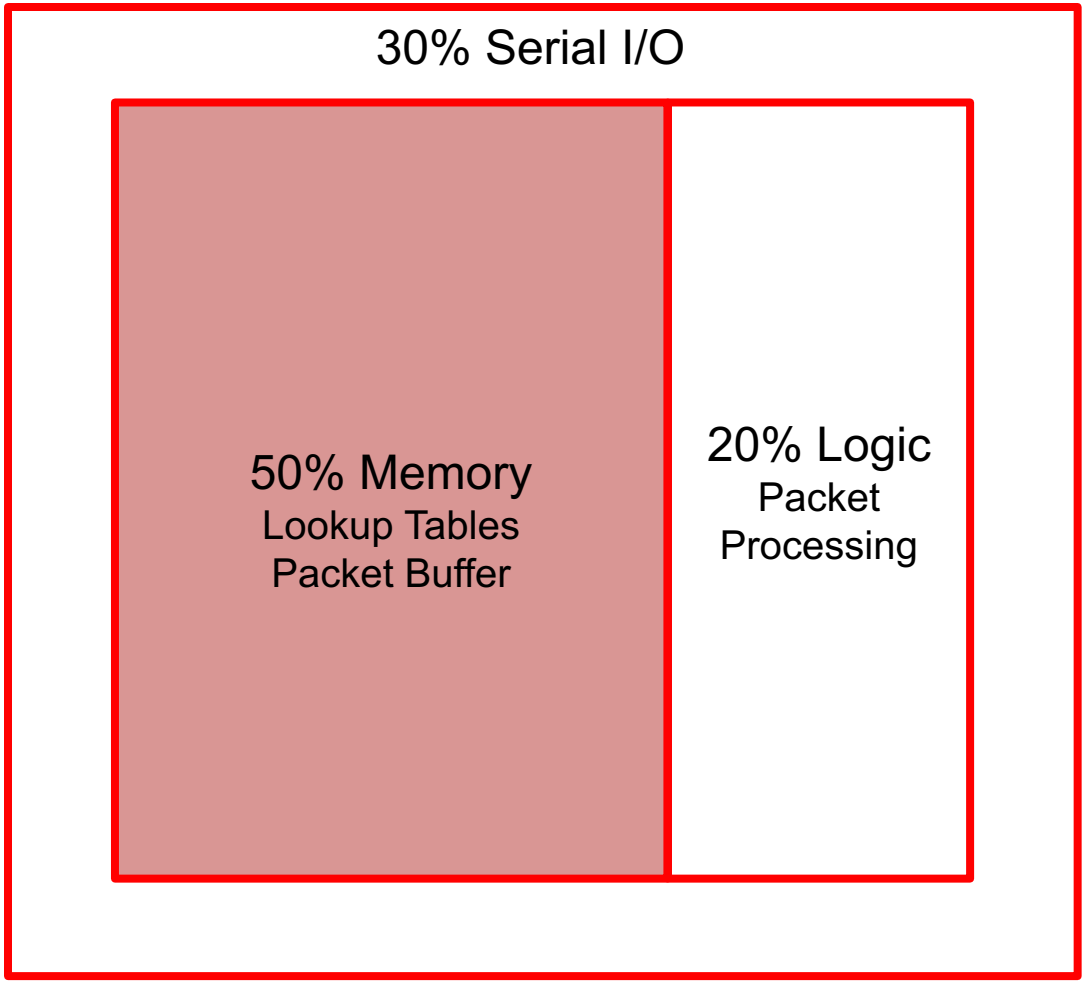
Broadcom Tomahawk (2014)



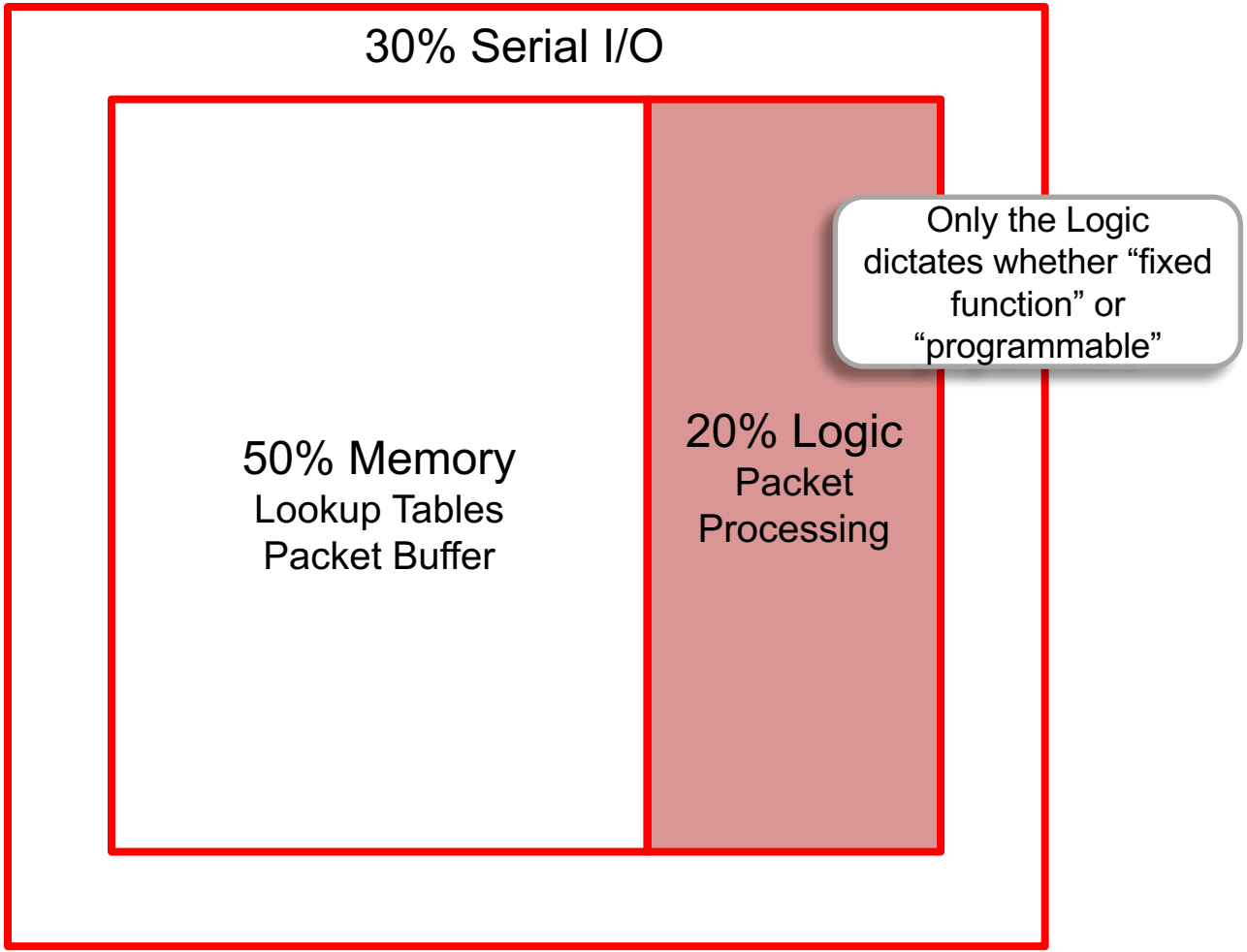
Barefoot Tofino (2016)

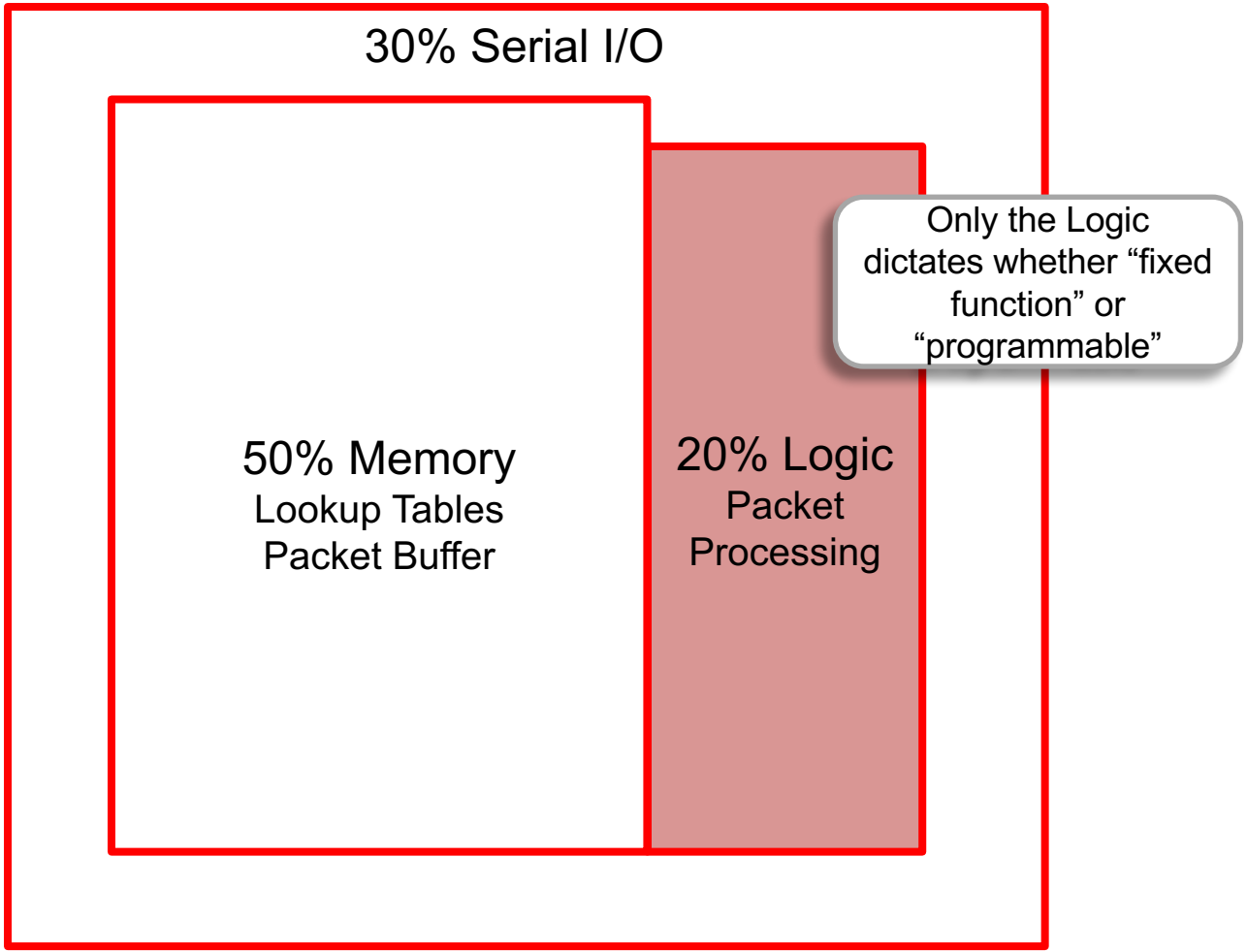
BAREFOOT NETWORKS

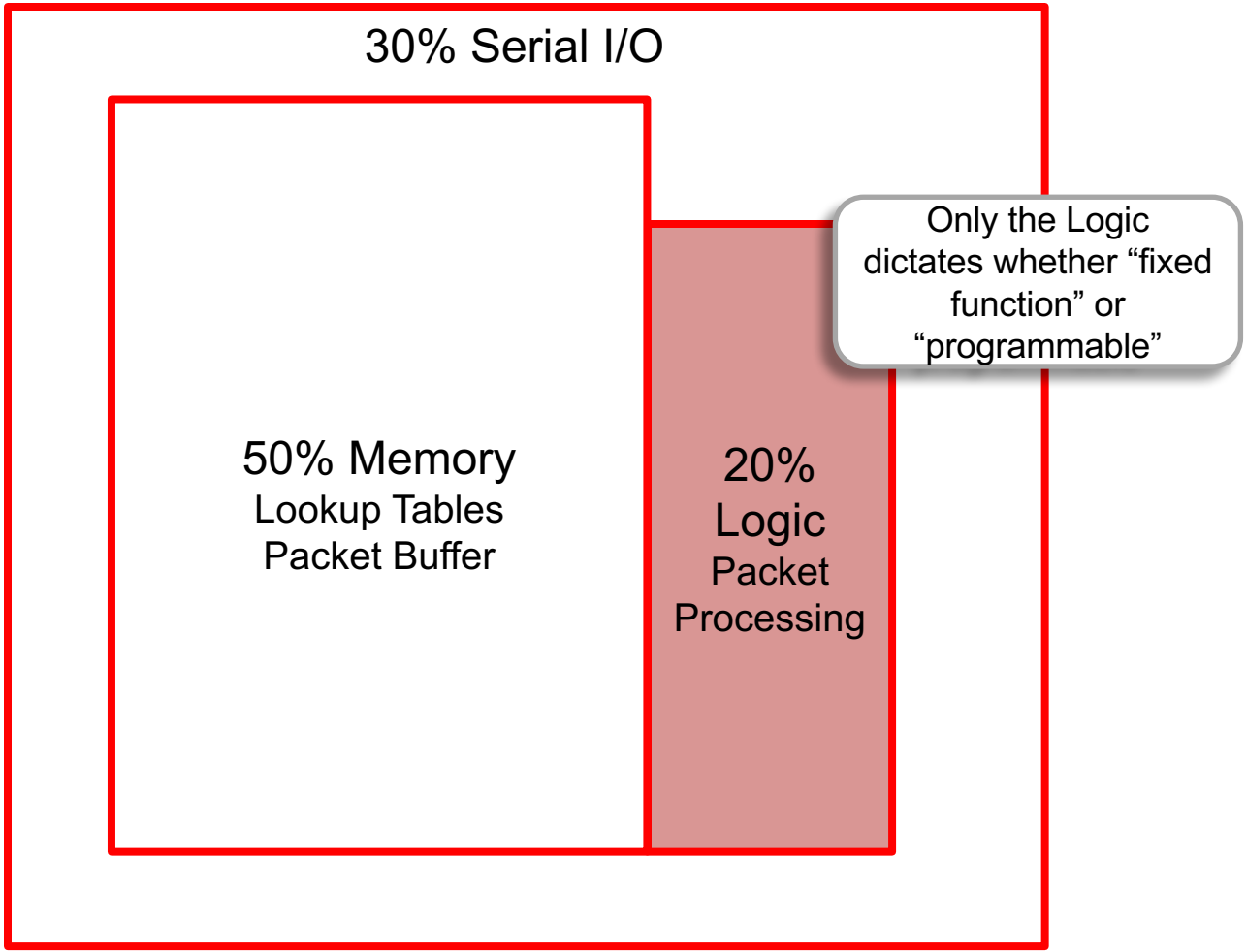


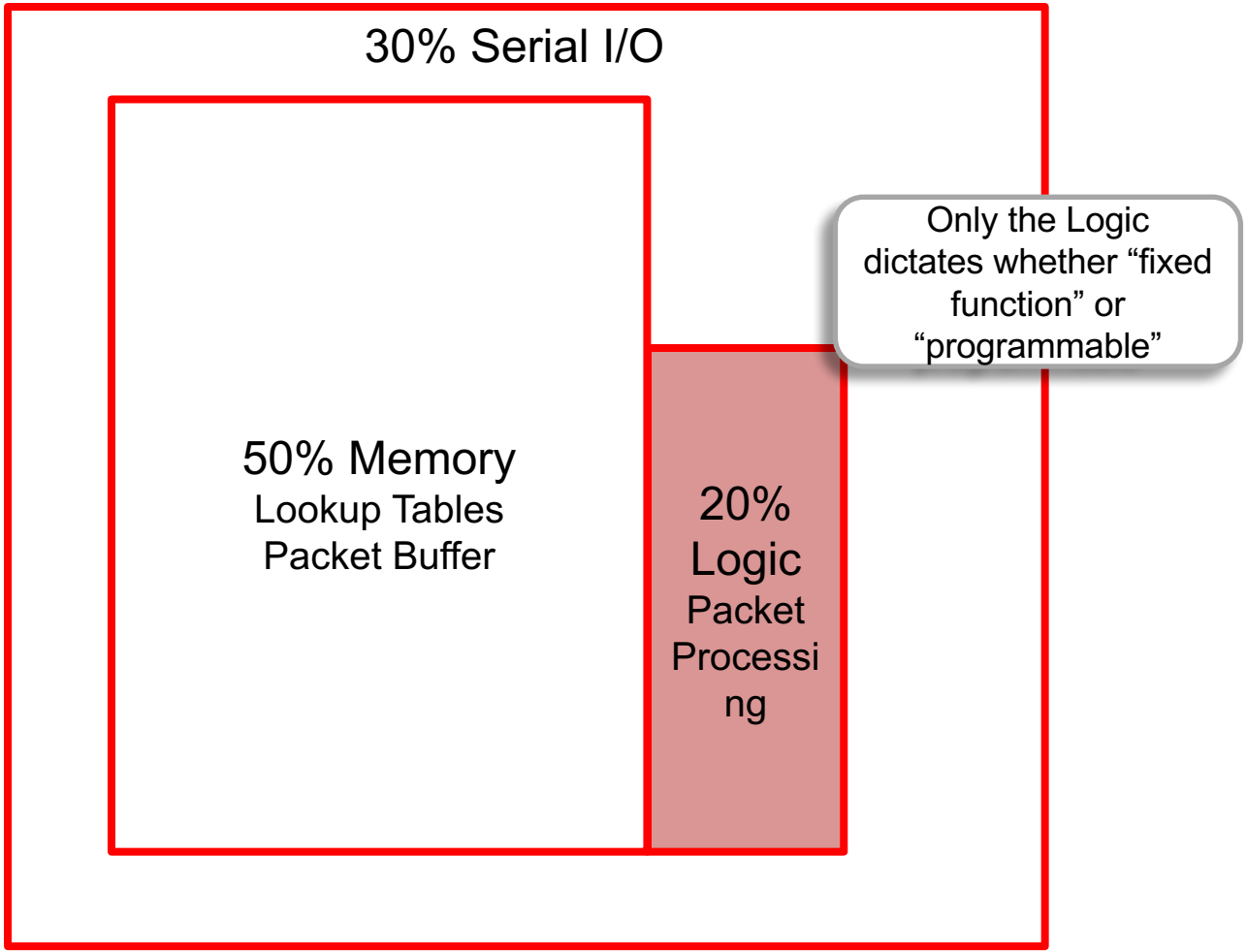


BAREFOOT NETWORKS





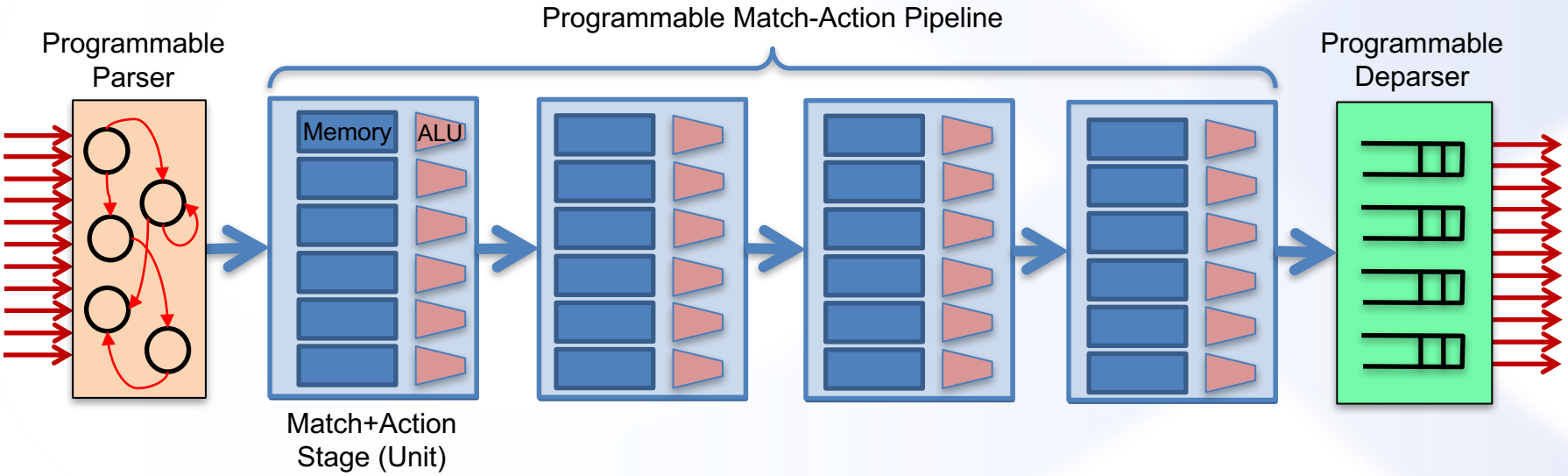




Parallelism and alternatives

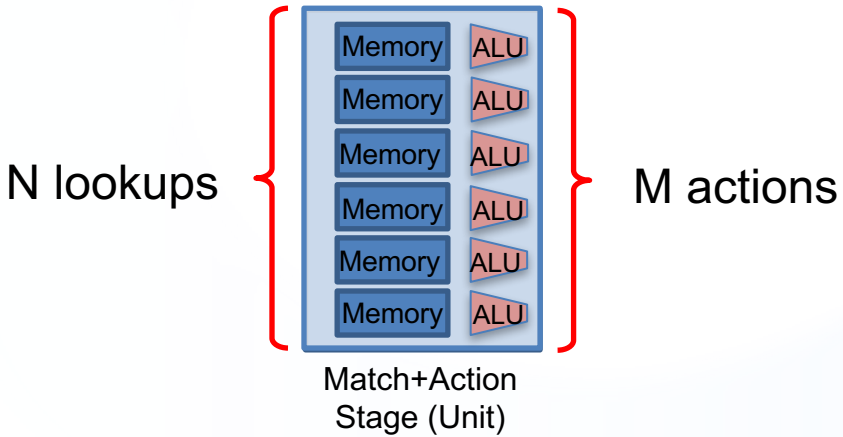
- Sequential semantics does not prohibit parallelism
- Doing everything does not mean doing everything all the time

PISA: Important Details



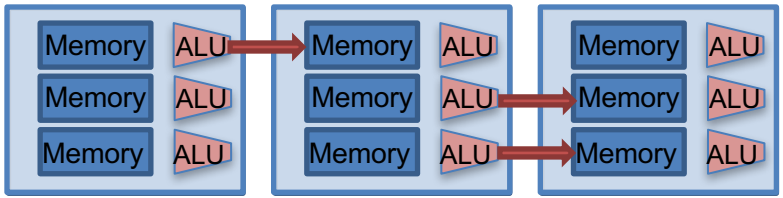
PISA: Important Details

- **Multiple simultaneous lookups and actions can be supported**



PISA: Match and Action are Separate Phases

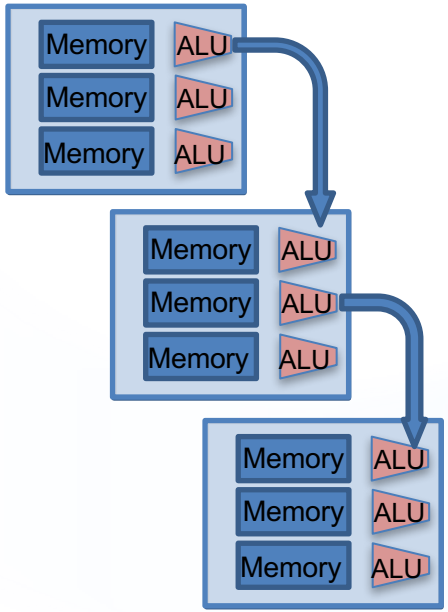
- **Sequential Execution (Match dependency)**



Total Latency = 3

PISA: Match and Action are Separate Phases

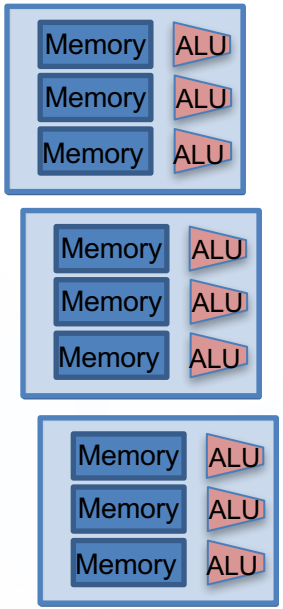
- Sequential Execution (Match Dependency)
- Staggered Execution (Action Dependency)



Total Latency = 2

PISA: Match and Action are Separate Phases

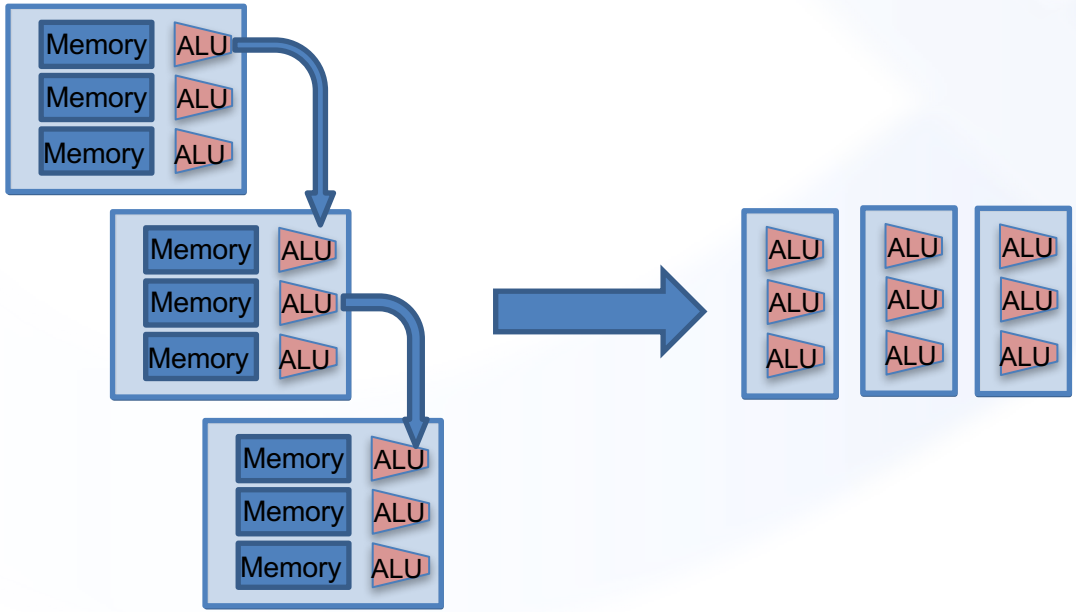
- Sequential Execution (Match Dependency)
- Staggered Execution (Action Dependency)
- Parallel Execution (No Dependencies)



Total Latency = 1.1

PISA: Match is not required

- A sequence of actions is an imperative program



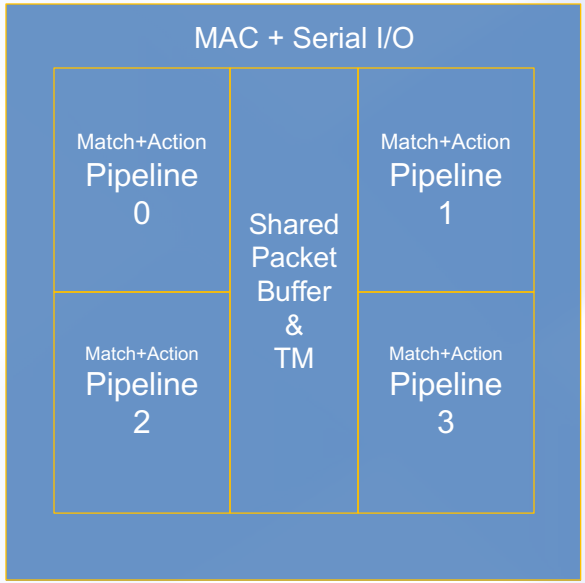
Symmetric Switch Model

- **Why not share the hardware between ingress and egress?**

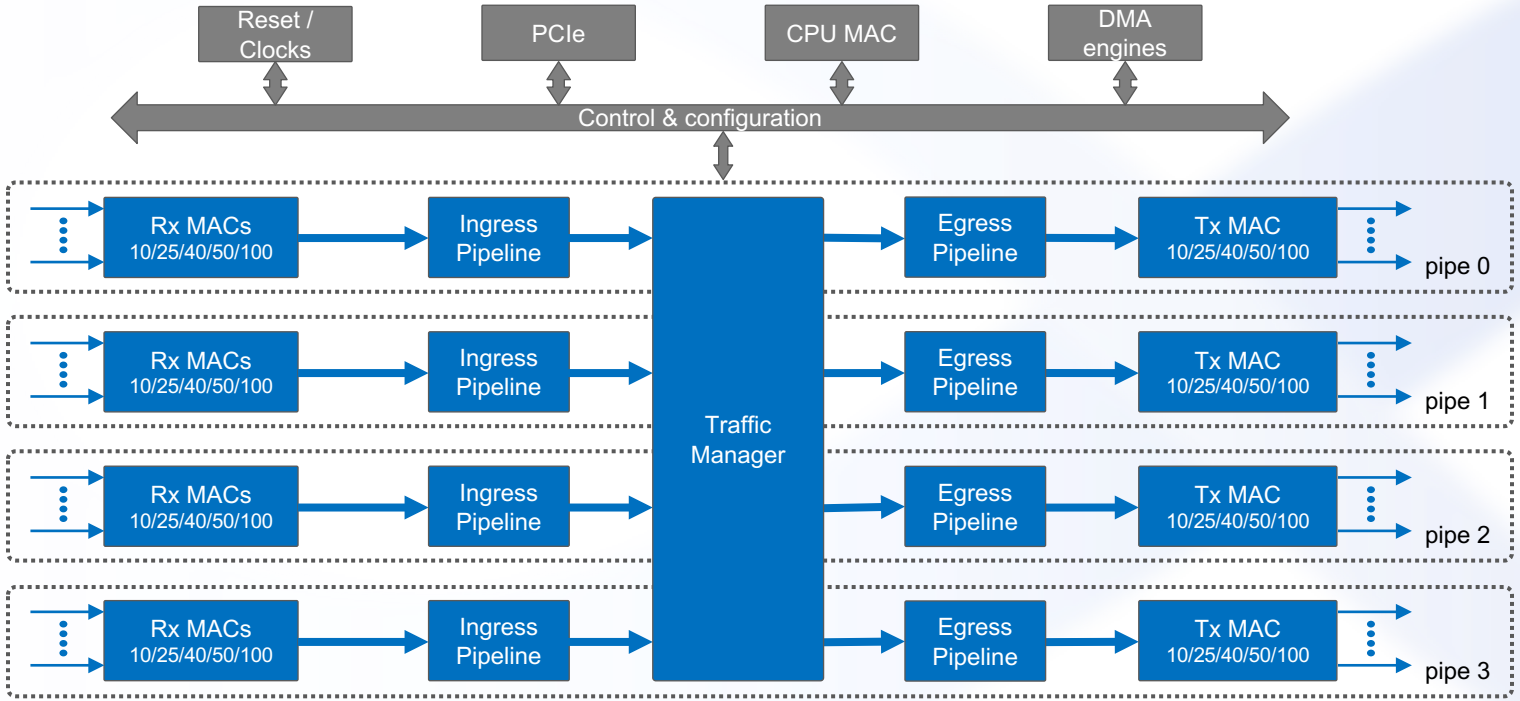
Introducing Tofino

6.5Tb/s Tofino™ Summary

- **State of the art design**
  - Single Shared Packet Buffer
  - TSMC 16nm FinFET+
- **Four Match+Action Pipelines**
  - Fully programmable PISA Embodiment
  - All compiled programs run at line-rate.
  - Up to 1.3 million IPv4 routes
- **Port Configurations**
  - 65 x 100GE/40GE
  - 130 x 50GE
  - 260 x 25GE/10GE
- **CPU Interfaces**
  - PCIe: Gen3 x4/x2/x1
  - Dedicated 100GE port



Tofino. Simplified Block Diagram

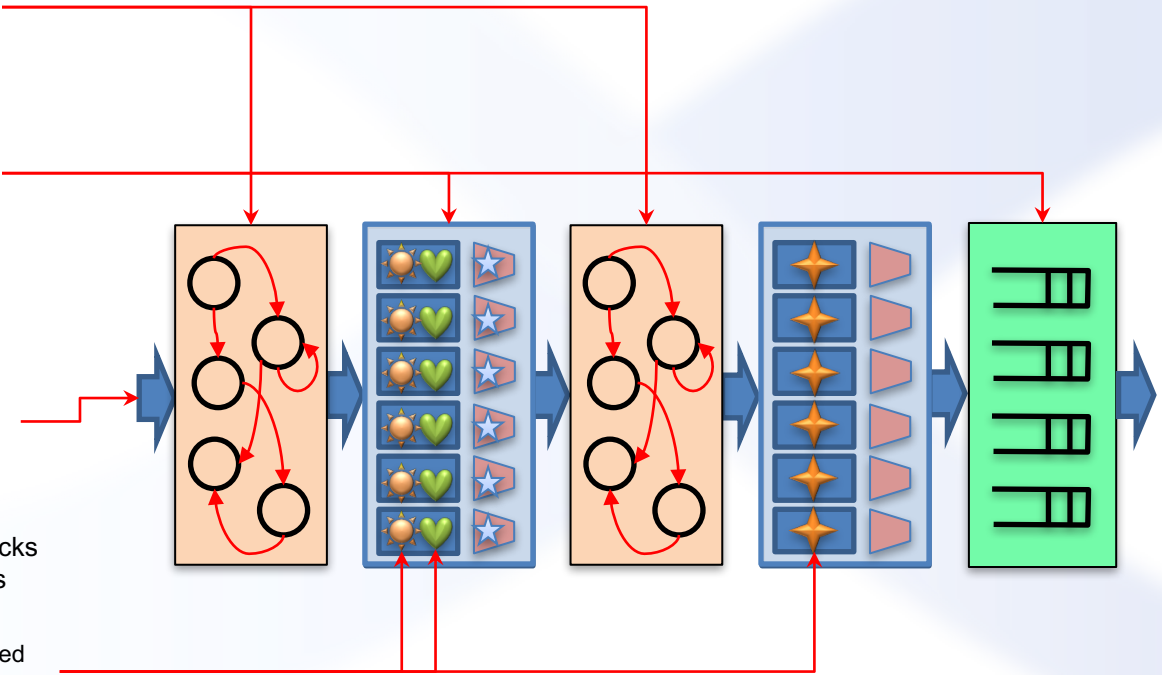


Each pipe has 16x100G MACs + a Packet
Additional ports for recirculation, Packet Generator, CPU

P4$_{16}$ Language Elements

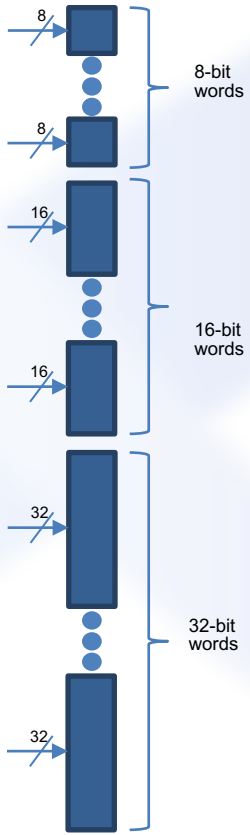
- **Parsers** — State machine, bitfield extraction
- **Controls** — Tables, Actions, control flow statements
- **Expressions** — Basic operations and operators
- **Data Types** — Bistrings, headers, structures, arrays

- **Architecture Description** — Programmable blocks and their interfaces
- **Extern Libraries** — Support for specialized components



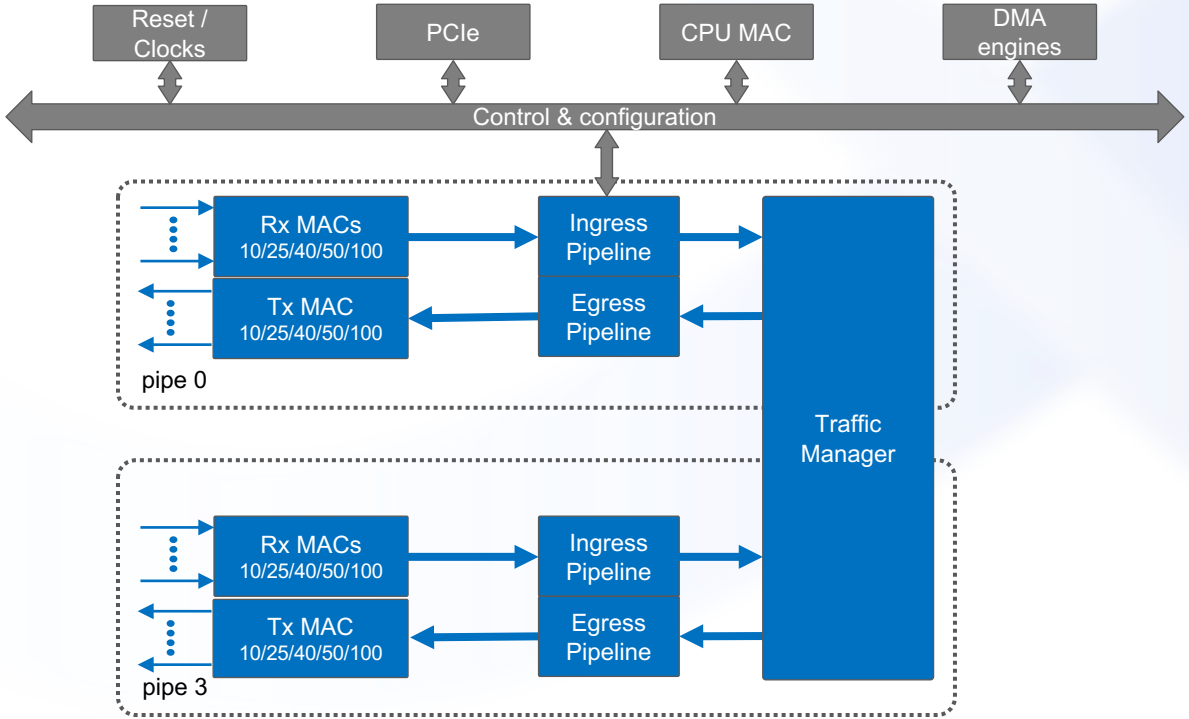
Packet Header Vector (PHV)

- A set of uniform containers that carry the headers and metadata along the pipeline
- Fields can be packed into any container or their combination
- PHV Allocation step in the compiler decides the actual packing

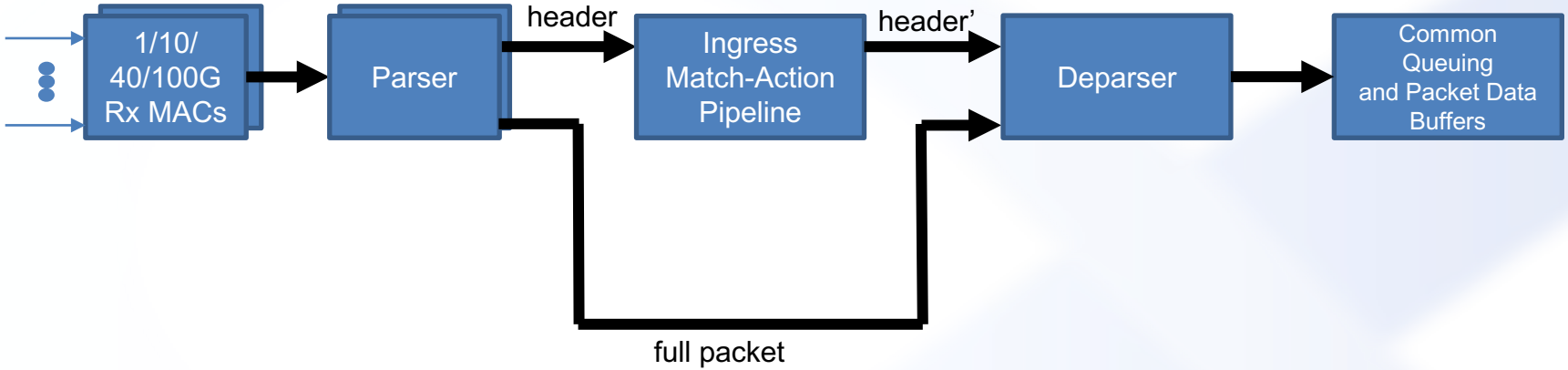


Unified Pipeline

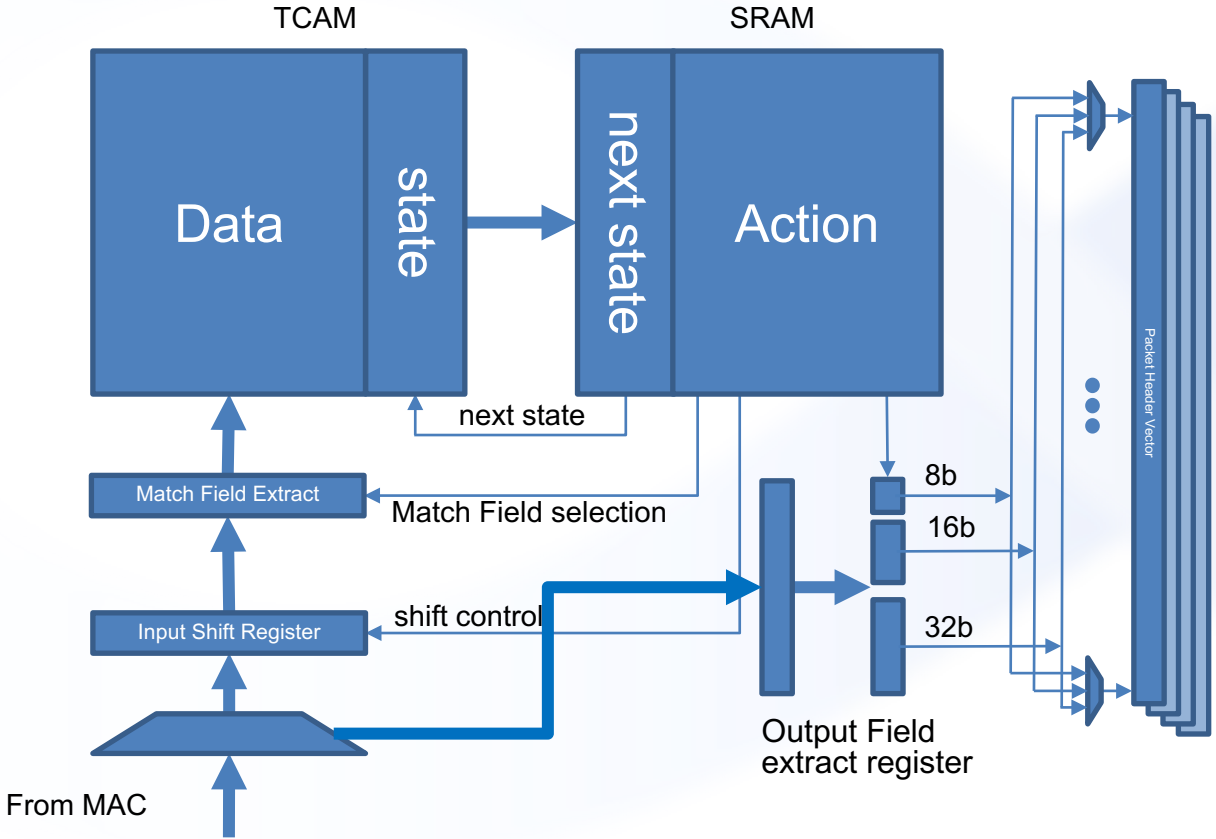
- **There is no difference between ingress and egress processing**
  - The same blocks can be efficiently shared



The Basic Structure



Parser



Parser Visualization

Row 147

State start_parse_ethernet (from state <POV initialization>_<Ingress intrinsic metadata>_<Phase 0>)

[▪] Raw register data

[▪] Input buffer

PHV 58 |= 0x400000

PHV 58 |= 0x1

[▪] Transitions

16b	
0000 && fe00	-> Row 146 (state parse_llc_header)
0000 && fa00	-> Row 145 (state parse_llc_header)
9000	-> Row 144 (state parse_fabric_header)
8100	-> Row 143 (state parse_vlan)
9100	-> Row 142 (state parse_qinq)
8847	-> Row 141 (state parse_mpls__it0)
0800	-> Row 140 (state parse_ipv4)
86dd	-> Row 139 (state parse_ipv6)
0806	-> Row 138 (state parse_arp_rarp)
88cc	-> Row 137 (state parse_set_prio_high)
8809	-> Row 136 (state parse_set_prio_high)
Default	-> Row 135 (state <leaf>)

Previous states: Row 255

Row 140

State parse_ipv4 (from state start_parse_ethernet)

[▪] Raw register data

[▪] Input buffer

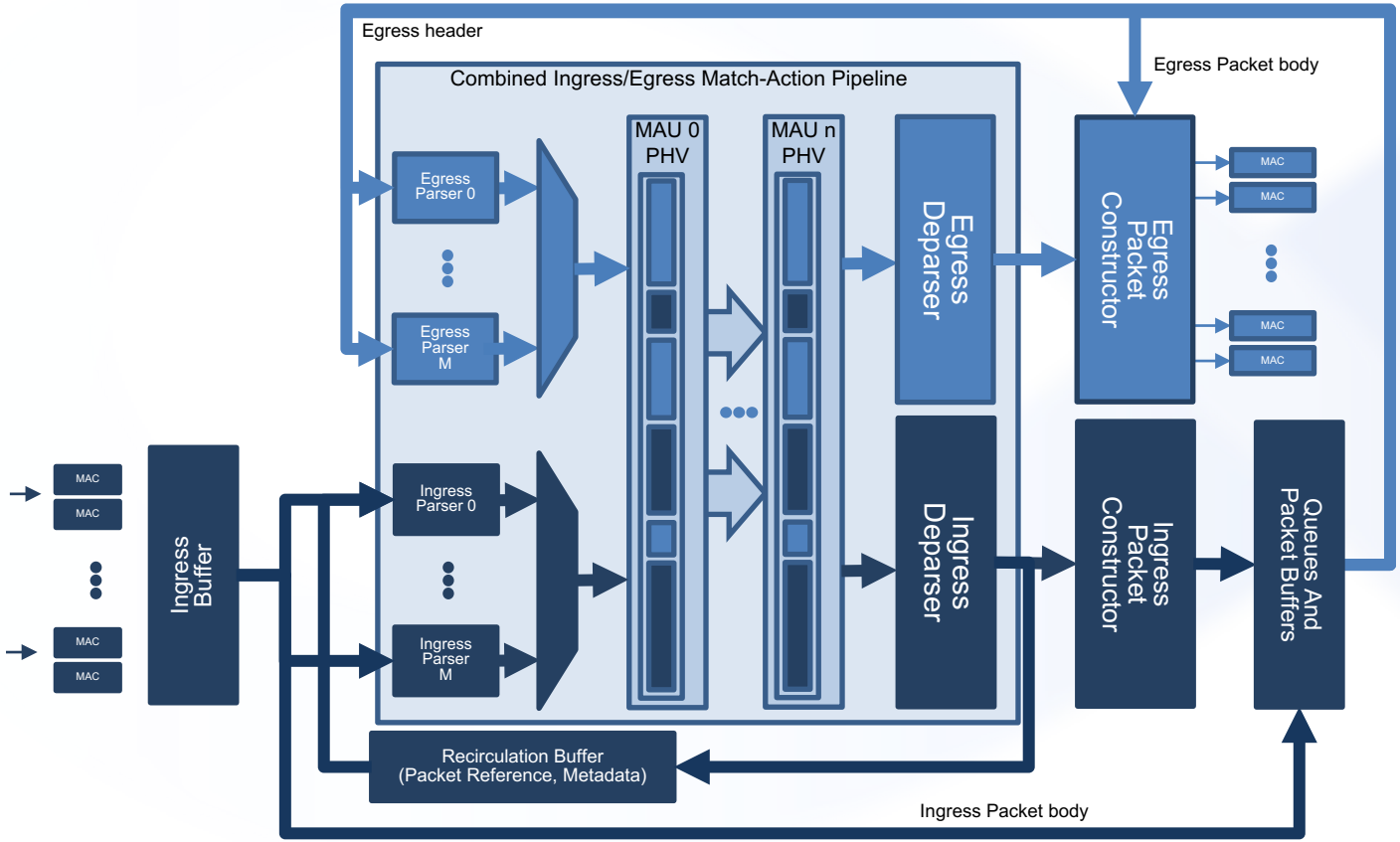
PHV 58 |= 0x8

[▪] Transitions

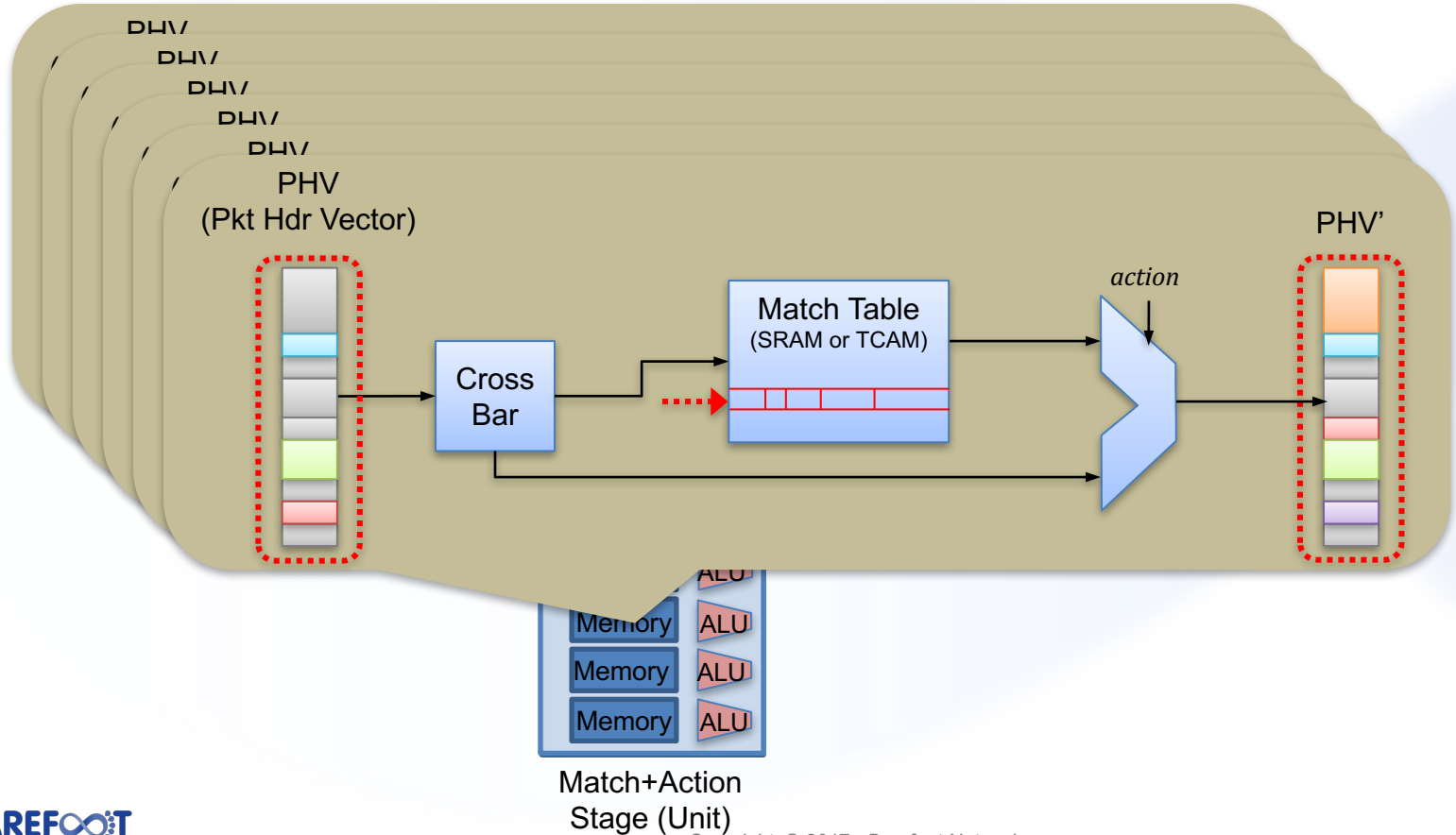
16b	8b[0]	8b[1]	
0000 && 1fff	05 && 0f	01	-> Row 217 (state parse_icmp)
0000 && 1fff	05 && 0f	06	-> Row 216 (state parse_tcp)
0000 && 1fff	05 && 0f	11	-> Row 215 (state parse_udp)
0000 && 1fff	05 && 0f	2f	-> Row 214 (state parse_gre)
0000 && 1fff	05 && 0f	04	-> Row 213 (state parse_ipv4_in_ip)
0000 && 1fff	05 && 0f	29	-> Row 212 (state parse_ipv6_in_ip)
0000 && 1fff	00 && 0f	02	-> Row 211 (state parse_set_prio_med)
0000 && 1fff	00 && 0f	58	-> Row 210 (state parse_set_prio_med)
0000 && 1fff	00 && 0f	59	-> Row 209 (state parse_set_prio_med)
0000 && 1fff	00 && 0f	67	-> Row 208 (state parse_set_prio_med)
0000 && 1fff	00 && 0f	70	-> Row 207 (state parse_set_prio_med)
Default			-> Row 206 (state <leaf>)

Previous states: Row 147

Pipeline Organization



What Happens Inside?



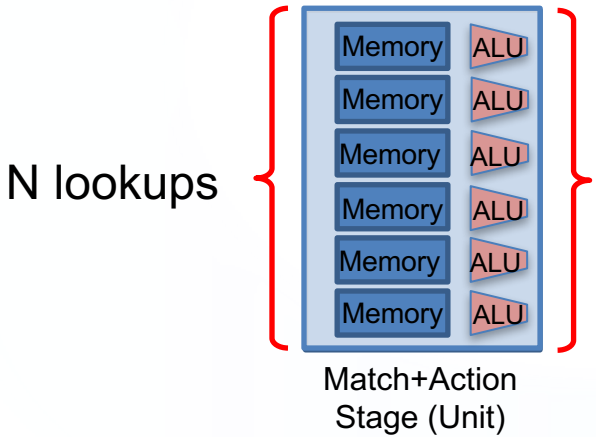
Parallelism in P4

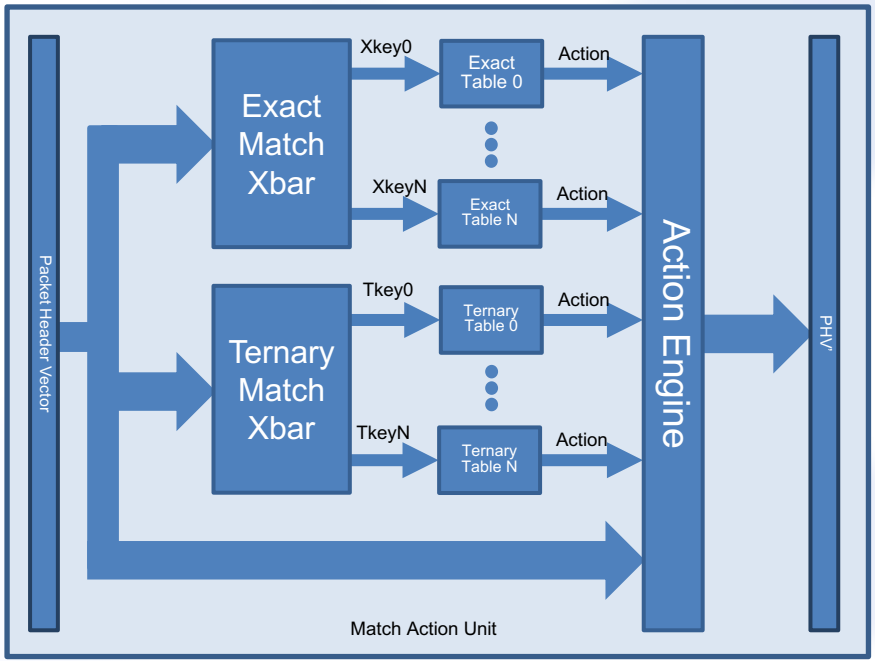
```
apply {
    /* Parallel lookups possible */
    subnet_vlan.apply();
    mac_vlan.apply();
    protocol_vlan.apply()
    port_vlan.apply();

    /* Resolution in next stage */
    resolve_vlan.apply();
}

apply {
    if (!subnet_vlan.apply().hit) {
        if (!mac_vlan.apply().hit) {
            if (!protocol_vlan.apply().hit) {
                port_vlan.apply();
            }
        }
    }
}
```

- **Most P4 programs have inherent parallelism**
- **Others can be executed speculatively**
- **Switch.p4**
  - ~100 tables and if() statements
  - ~22 stages divided between ingress and egress
  - Degree of parallelism ~4.5

How Tofino Supports Parallel Processing

- **Multiple tables mean multiple parallel lookups**
- **All actions from all active tables are combined**

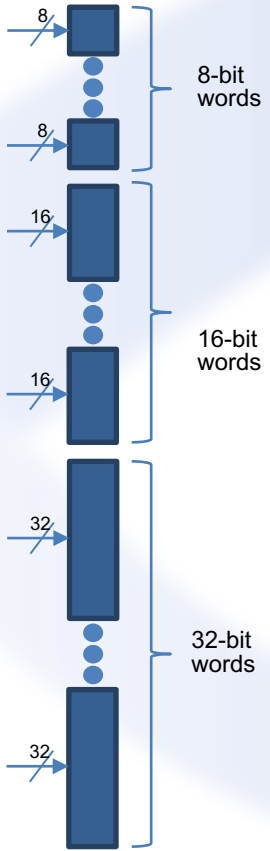




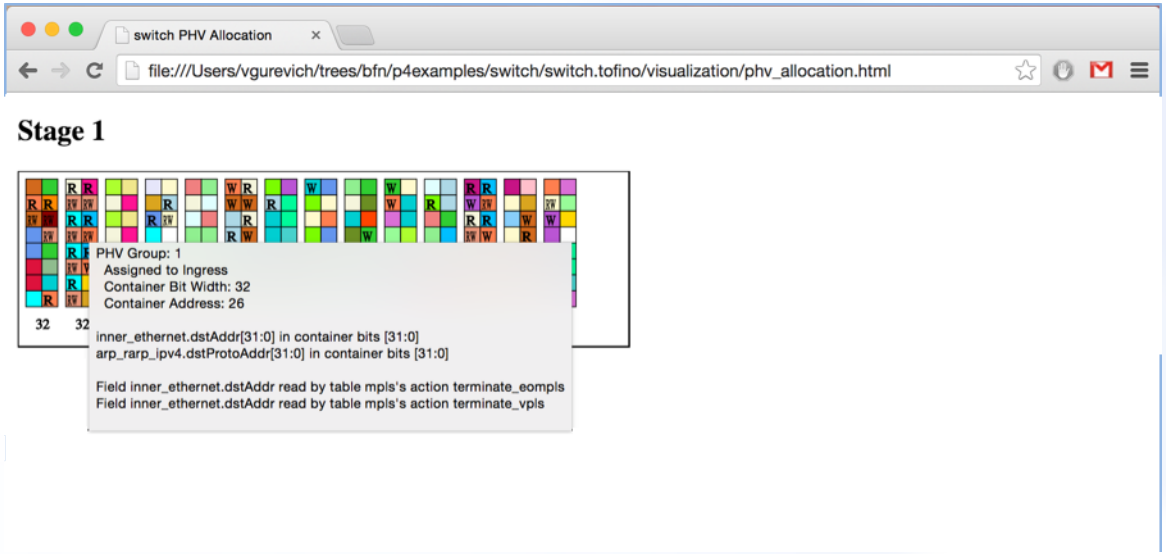
Parallelism in P4

```
action ipv4_in_mpls(in bit<20> label1, in bit<20> label2) {
    hdr.mpls[0].setValid();
    hdr.mpls[0].label = label1;
    hdr.mpls[0].exp   = 0;
    hdr.mpls[0].bos   = 0;
    hdr_mpls[0].ttl   = 64;

    hdr.mpls[1].setValid();
    hdr.mpls[1] = { label2, 0, 1, 128 };

    if (hdr.vlan_tag.isValid()) {
        hdr.vlan_tag.etherType = 0x8847;
    } else {
        hdr.ethernet.etherType = 0x8847;
    }
}
```

- **Most actions can be easily parallelized**
  - This action can be executed in 1 cycle
    - Number of parallel operations: 12
- **Keep fields in separate containers**

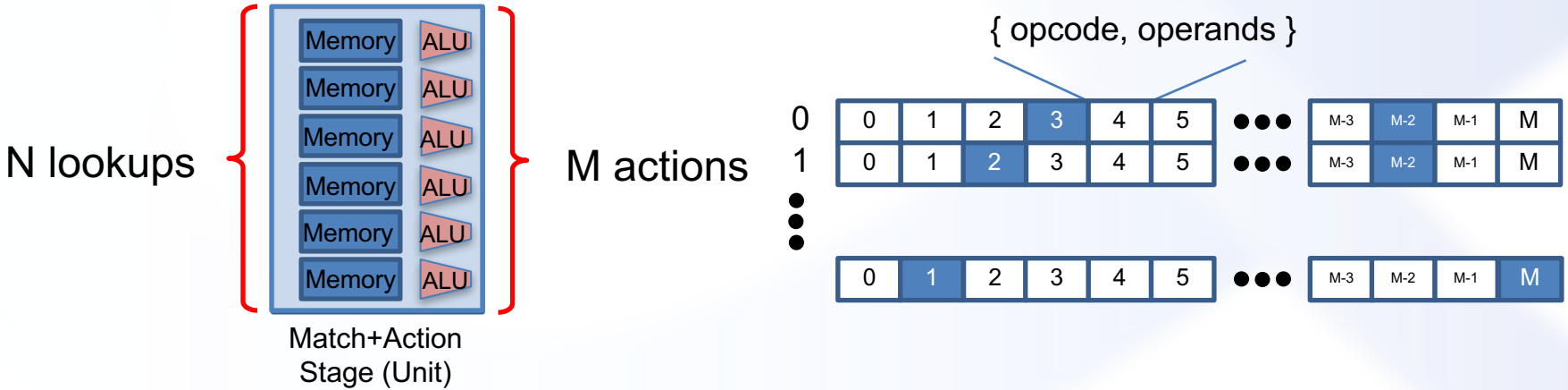


P4 Visualizations (PHV Allocation)



How Tofino Supports Parallel Processing

- **Multiple tables mean multiple parallel lookups**
- **All actions from all active tables are combined**
- **Each PHV container has its own, independent processor**

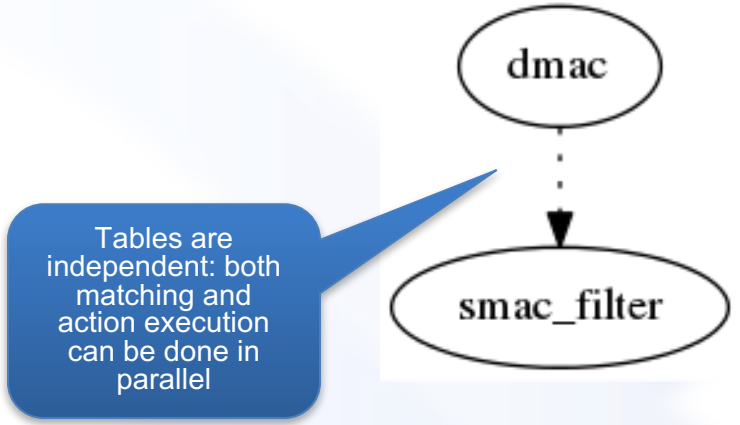


Dependency Analysis

- Independent tables
- Match Dependency
- Action Dependency

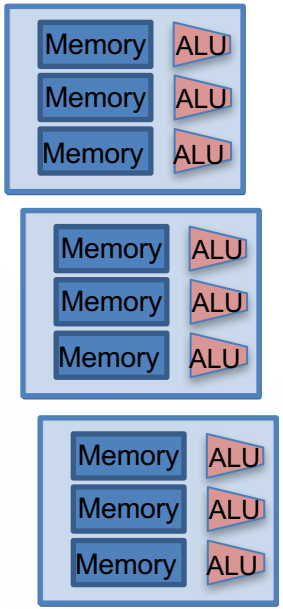
Independent Tables

```
action ing_drop() {
    modify_field(ing_metadata.drop, 1);
}

action set_egress_port(egress_port) {
    modify_field(ing_metadata.egress_spec, egress_port);
}

table dmac {
    reads {
        ethernet.dstAddr : exact;
    }
    actions {
        nop;
        set_egress_port;
    }
}

table smac_filter {
    reads {
        ethernet.srcAddr : exact;
    }
    actions {
        nop;
        ing_drop;
    }
}

control ingress {
    apply(dmac);
    apply(smac_filter);
}
```



Match and Action are Separate Phases

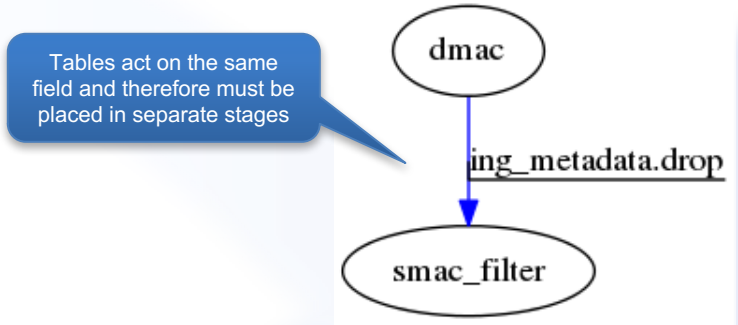
- **Parallel Execution (No Dependencies)**

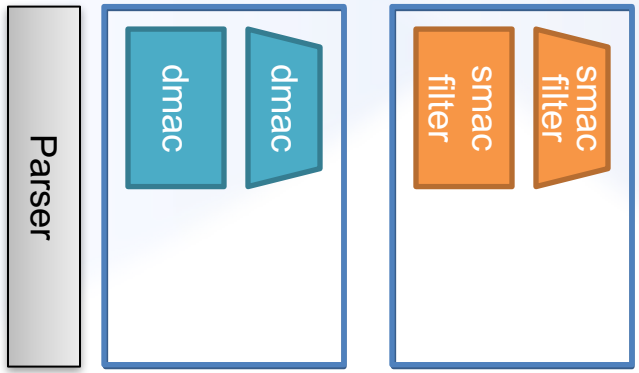


Total Latency = 1.1

Action Dependency

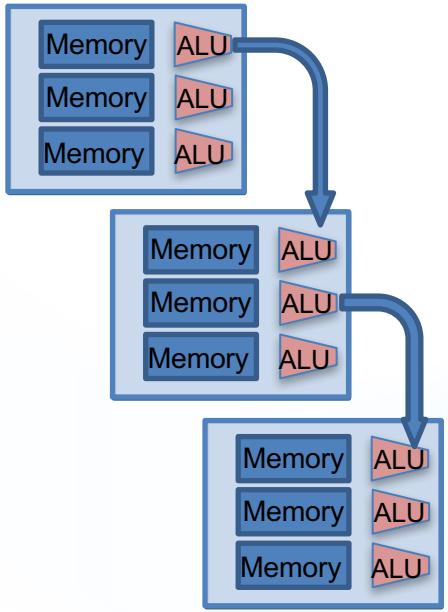
```
action ing_drop() {
    modify_field(ing_metadata.drop, 1);
}

action set_egress_port(egress_port) {
    modify_field(ing_metadata.egress_spec, egress_port);
}

table dmac {
    reads {
        ethernet.dstAddr : exact;
    }
    actions {
        nop;
        ing_drop;
        set_egress_port;
    }
}

table smac_filter {
    reads {
        ethernet.srcAddr : exact;
    }
    actions {
        nop;
        ing_drop;
    }
}

control ingress {
    apply(dmac);
    apply(smac_filter);
}
```





Match and Action are Separate Phases

- **Staggered Execution (Action Dependency)**

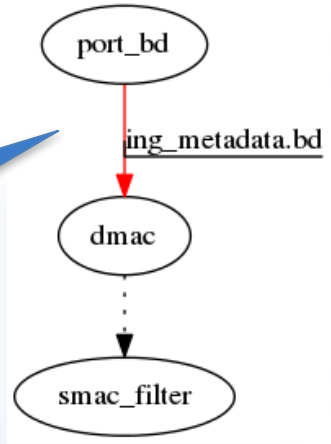


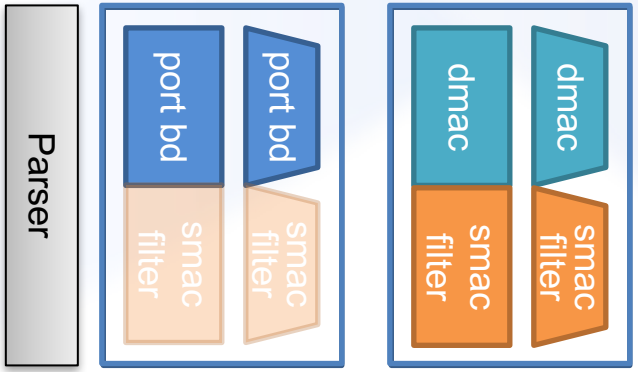
Total Latency = 2

Match Dependency

```
action set_bd(bd) {
    modify_field(ing_metadata.bd, bd);
}

table port_bd {
    reads {
        ing_metadata.ingress_port : exact;
    }
    actions {
        set_bd;
    }
}

table dmac {
    reads {
        ethernet.dstAddr : exact;
        ing_metadata.bd  : exact;
    }
    actions {
        nop;
        set_egress_port;
    }
}

table smac_filter {
    reads {
        ethernet.srcAddr : exact;
    }
    actions {
        nop;
        ing_drop;
    }
}

control ingress {
    apply(port_bd);
    apply(dmac);
    apply(smac_filter);
}
```

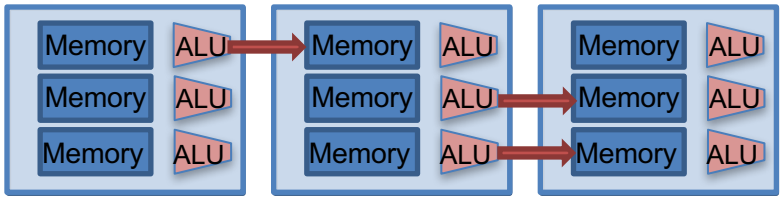
The second table matches on the field, modified by the first.





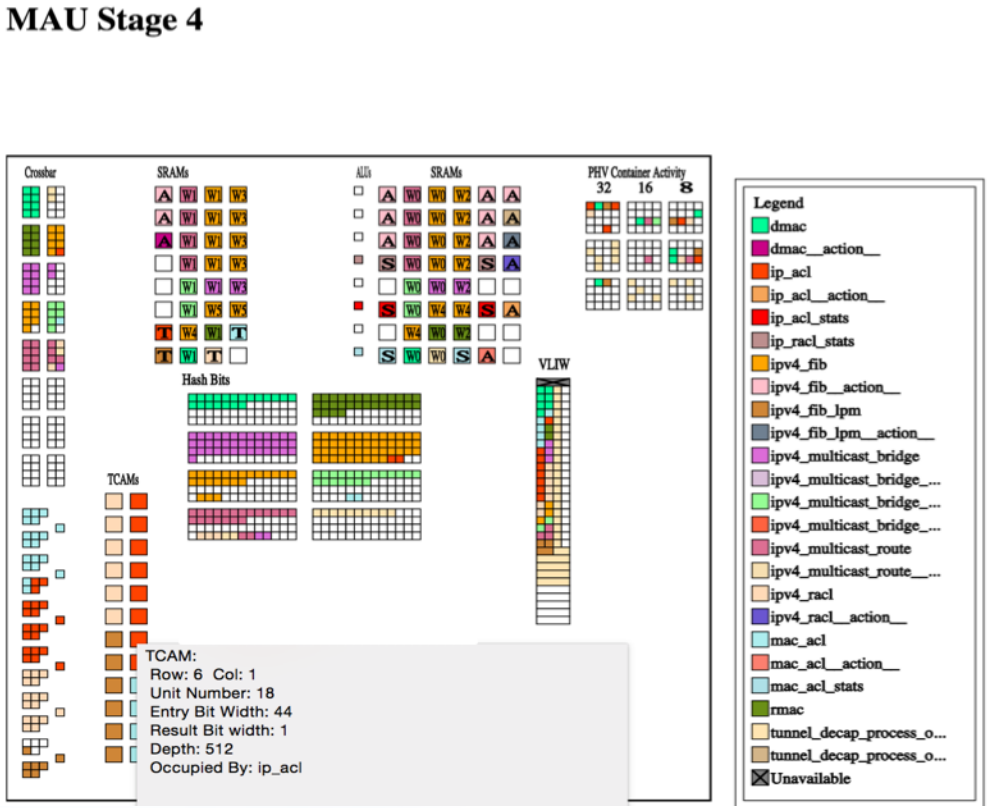
Reducing Latency: Match and Action are Separate Phases

- **Sequential Execution (Match dependency)**

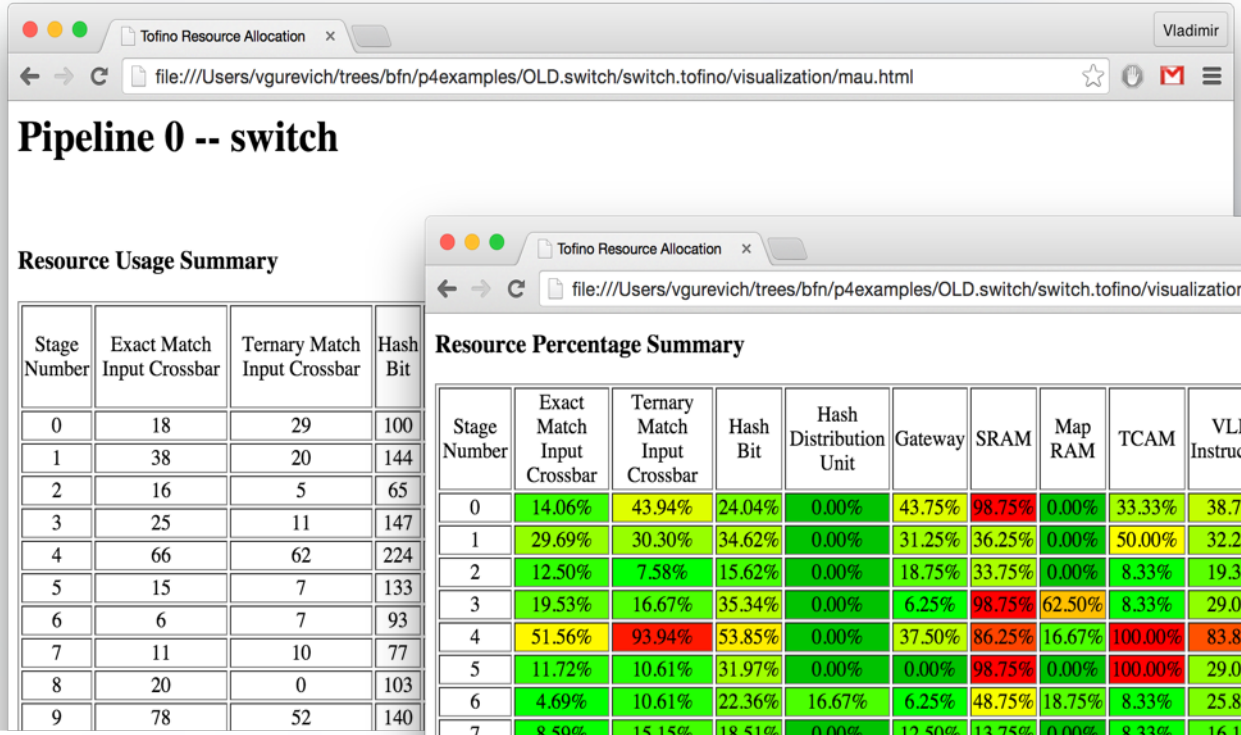


Total Latency = 3

P4 Visualizations (Resource Allocation)



P4 Visualizations (Resource Usage Summary)



Conclusions

- **Programmable Data Plane at the highest performance point is a reality now**
- **P4 and real-world programs provide a plenty of optimization opportunities**

Thank you

BAREFOOT NETWORKS