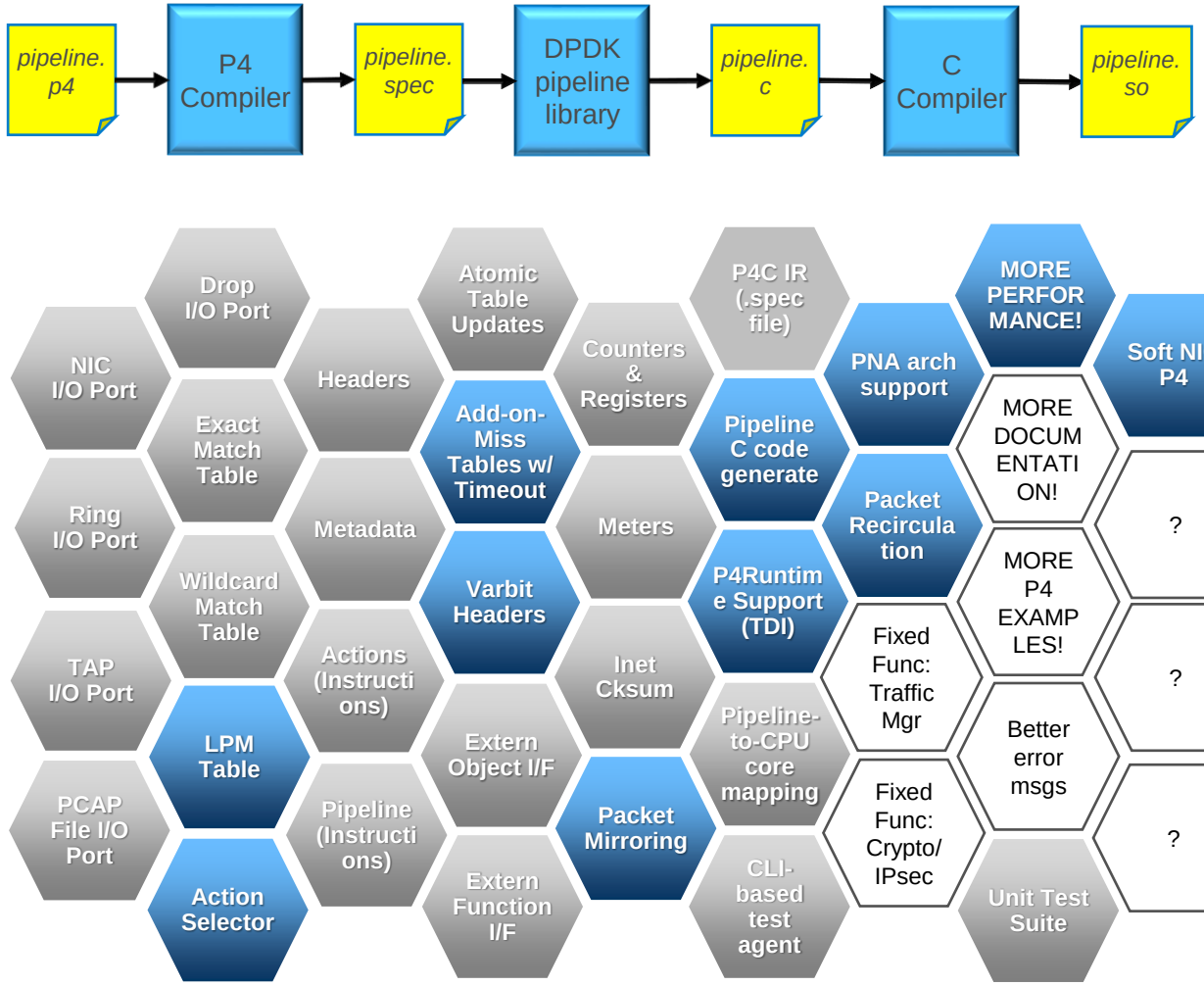
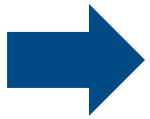
P4-based Network Stack Example: P4-DPDK

- P4-DPDK: Complete P4-based network stack in Linux user space with performance focus.
- Open-source project: P4C compiler back-end on P4.org (link), P4 data plane engine on dpdk.org (link).
- Actively maintained and developed.
- See also: P4 talk, DKDP talk



P4-DPDK Parser Implementation

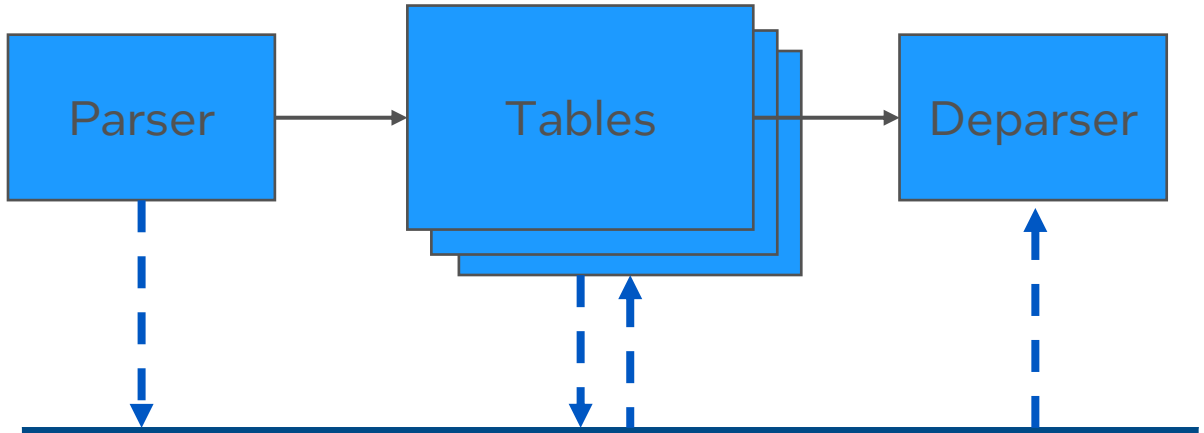
```
Checksum() ipv4_checksum;

state start {
    transition parse_ethernet;
}

// Ethernet
state parse_ethernet {
    pkt.extract(hdr.ethernet);
    transition select(hdr.ethernet.ether_type) {
        ether_type_t.TPID :  parse_vlan_tag;
        ether_type_t.IPV4 :  parse_ipv4;
        ether_type_t.IPV6 :  parse_ipv6;
        default :  accept;
    }
}

// VLAN label
state parse_vlan_tag {
    pkt.extract(hdr.vlan_tag.next);
    transition select(hdr.vlan_tag.last.ether_type) {
        ether_type_t.TPID :  parse_vlan_tag;
        ether_type_t.IPV4 :  parse_ipv4;
        default: accept;
    }
}

// IPv4
state parse_ipv4 {
    pkt.extract(hdr.ipv4);
        ipv4_checksum.add(hdr.ipv4);
        transition select(hdr.ipv4.ihl) { ...
```



```
//rx &init
rx m.port_in
mov m.drop 0
mov h.ipv4_cksum.cksum 0x0

//state parse_ethernet
PARSE_ETHERNET : extract h.ethernet
jmpeq PARSE_VLAN_TAG
h.ethernet.ethertype 0x8100
jmpeq PARSE_IPV4 h.ethernet.ethertype 0x800
jmpeq PARSE_IPV6 h.ethernet.ethertype 0x86dd
jmp ACCEPT

//state parse_vlan_tag
PARSE_VLAN_TAG : extract h.vlan_tag_0
jmpeq PARSE_VLAN_TAG1 h.vlan_tag_0.ethertype 0x8100
jmpeq PARSE_IPV4 h.vlan_tag_0.ethertype 0x800
jmp ACCEPT  PARSE_VLAN_TAG1 : extract h.vlan_tag_1
jmpeq PARSE_IPV4 h.vlan_tag_1.ethertype 0x800
jmp ACCEPT

//state parse_ipv4
PARSE_IPV4 : extract h.ipv4
ckadd h.ipv4_cksum.cksum h.ipv4
jmpeq PARSE_IPV4_NO_OPTIONS h.ipv4.version_ihl 0x45
jmplt DROP h.ipv4.version_ihl 0x45
jmpgt DROP h.ipv4.version_ihl 0x4F

...
```

P4-DPDK Parser Implementation (2)

- The *parser* extracts (and removes) headers from the input packet.
- The *pipeline* (control apply, table actions) can read & modify the extracted headers and may generate new headers.
- The *deparser* inserts valid headers into the output packet.
  - When an extracted header is put back at the same position, we try to avoid the memory copy.



Header Instance	Pkt Buffer Offset (If extracted)	Header Buffer (If generated)	Header Length (Bytes)
Ethernet	xxx		14
VLAN0	xxx		4
VLAN1	xxx		4
IPv4		xxx	20
Outer Ethernet		xxx	14
Outer IPv4		xxx	20