Portable NIC Architecture Update

Andy Fingerhut, Gordon Brebner
Co-chairs, P4.org Architecture working group
May 2021

Questions

- Why a new P4 architecture?
- How is PNA different than switch architectures?
- What data plane features are enabled?
- When will PNA be ready?
- What is left to do?
- How can I help?

Why a new P4 architecture?

- The physical interfaces of a NIC are different than a switch
- Like a switch, a NIC has one or more Ethernet ports

Why a new P4 architecture?

- The physical interfaces of a NIC are different than a switch
- Like a switch, a NIC has one or more Ethernet ports

- but also a host memory interface, typically PCI
  - Tx and Rx descriptors written by drivers of multiple OSes, interrupts
  - Descriptors can point at blocks of host memory much larger than 9 KByte Ethernet jumbo frame

Why a new P4 architecture?

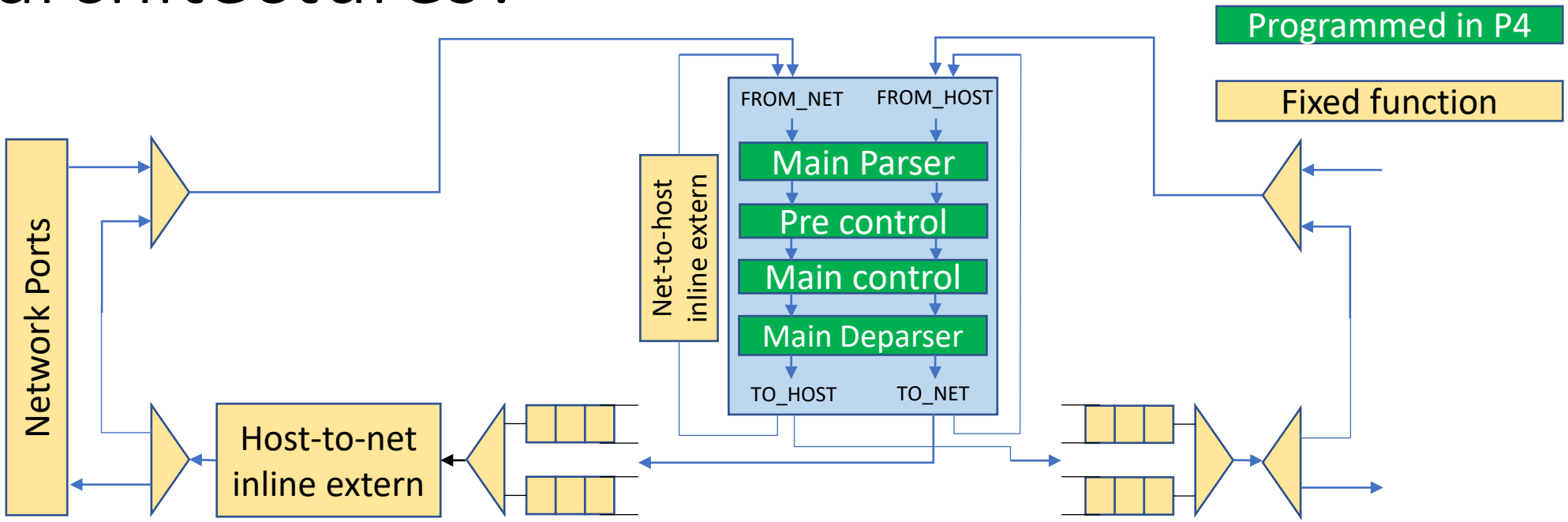
- Many features are as useful in NICs as they are in switches:
  - L2/L3 forwarding
  - Tunnel encap/decap
  - QoS: policing, marking, queue selection

Why a new P4 architecture?

- Many features are as useful in NICs as they are in switches:
  - L2/L3 forwarding
  - Tunnel encap/decap
  - QoS: policing, marking, queue selection

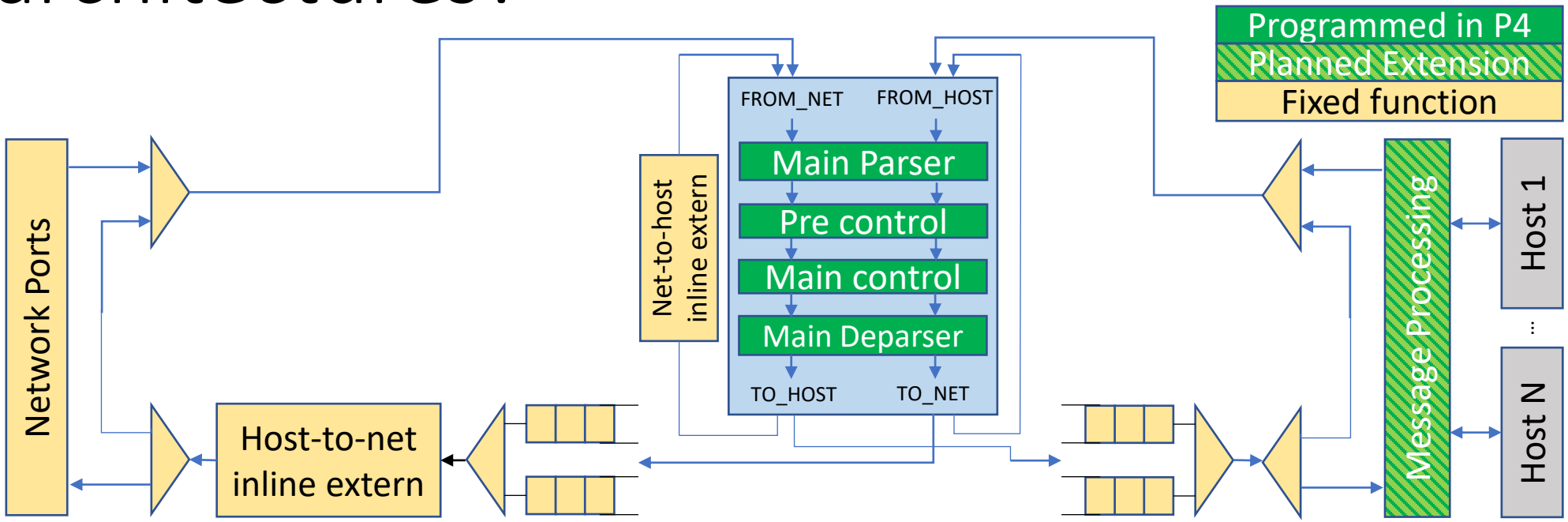
- … but many features are often seen in NICs, rarely in switches:
  - Remote DMA offload (RDMA)
  - IPsec encrypt/decrypt
  - TCP connection tracking
  - TCP Segmentation Offloading (TSO)
  - Large Receive Offload (LRO)
  - Receive Side Scaling (RSS)

How is PNA different than switch architectures?



- Split into two parts
  - Packet processing: operates on individual packets at most MTU in size

How is PNA different than switch architectures?

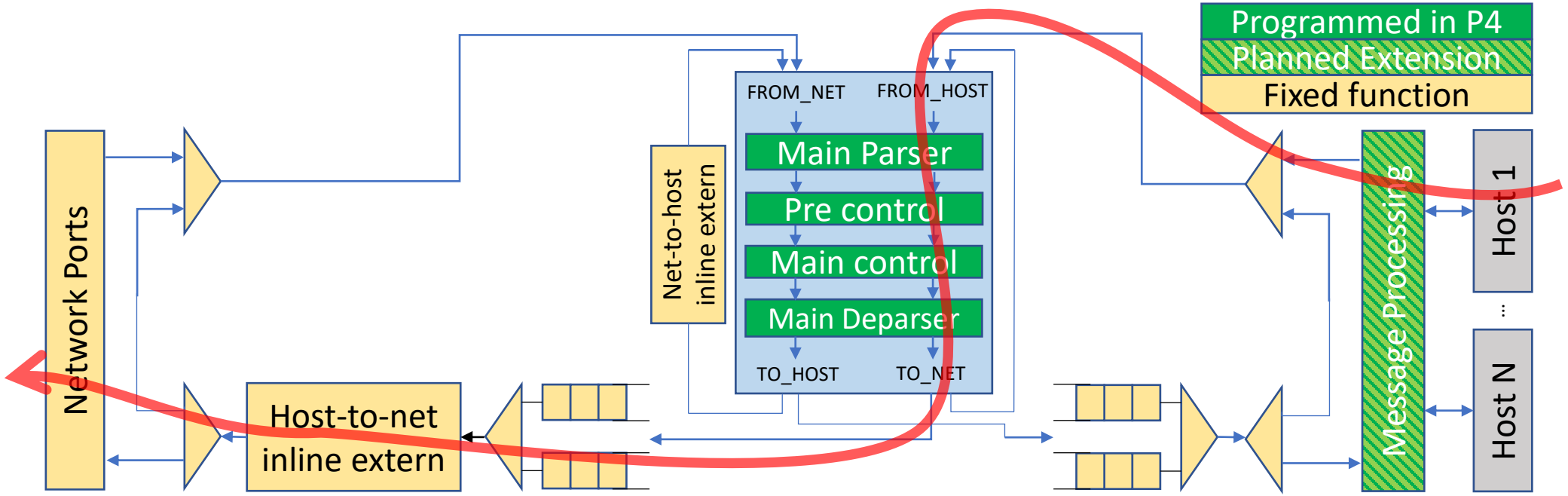


- Split into two parts
  - Packet processing: operates on individual packets at most MTU in size
  - Message processing: operates on larger blocks of data to/from host memory

What data plane features are enabled?

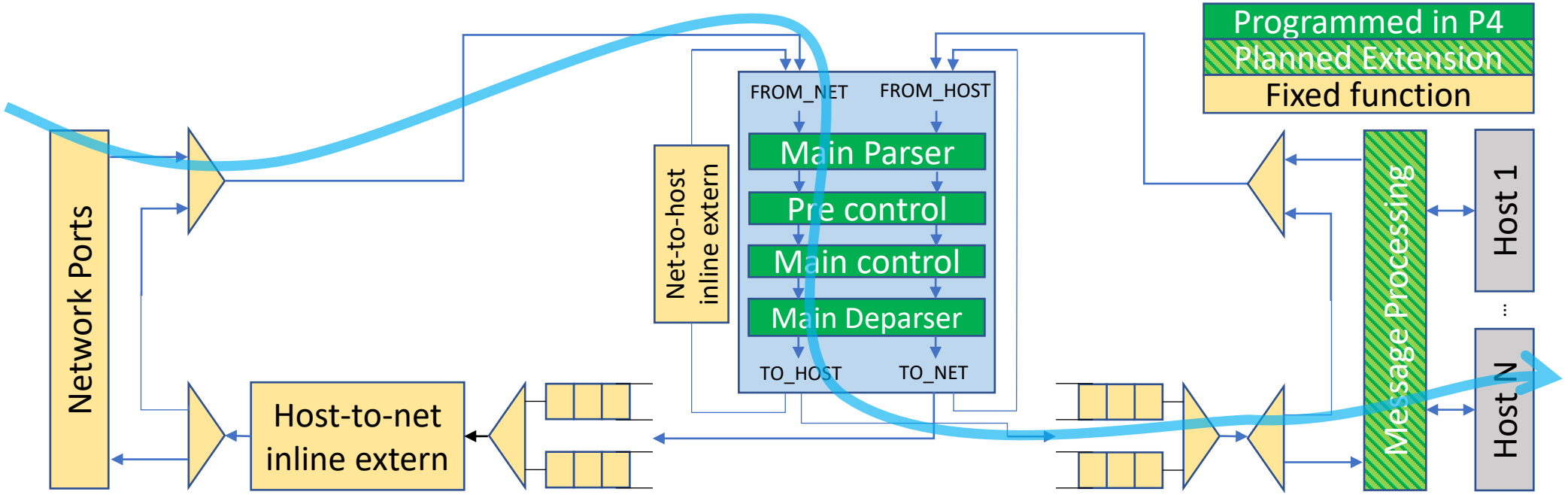
- Early in work group meetings, we decided to focus first on packet processing
- And on a few features distinctive to PNA
  - Directionality
  - TCP connection tracking
  - IPsec encrypt/decrypt
  - Goal: cover vSwitch feature set

Directionality



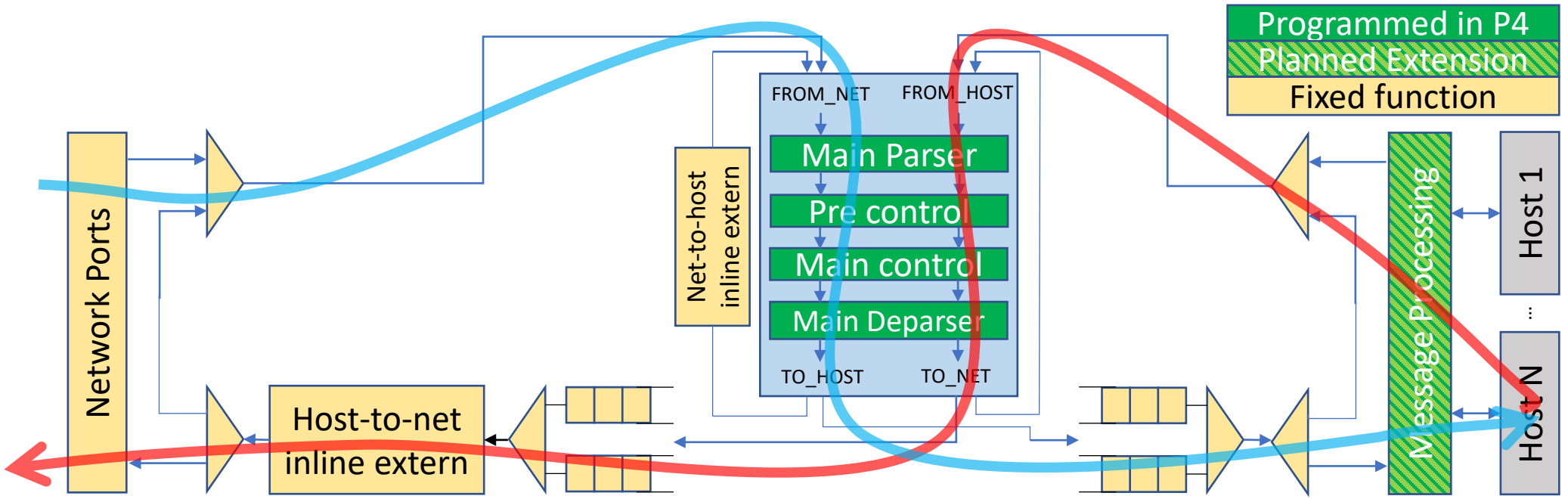
- Host-to-net packet

Directionality



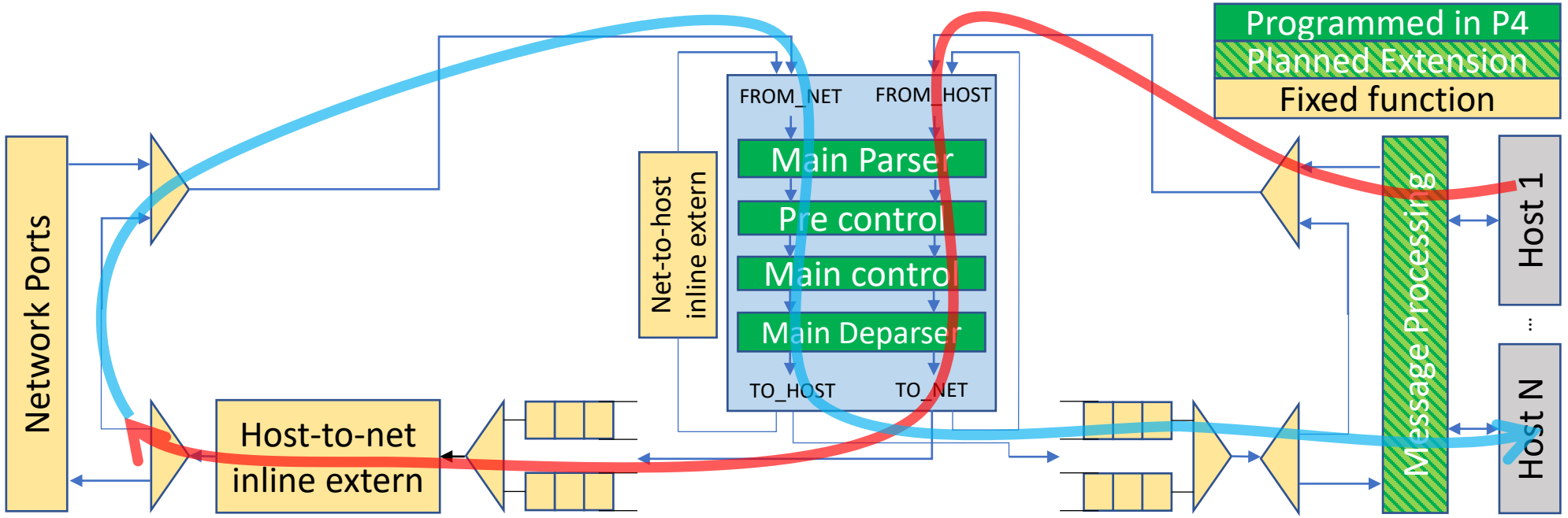
- Net-to-host packet

Directionality



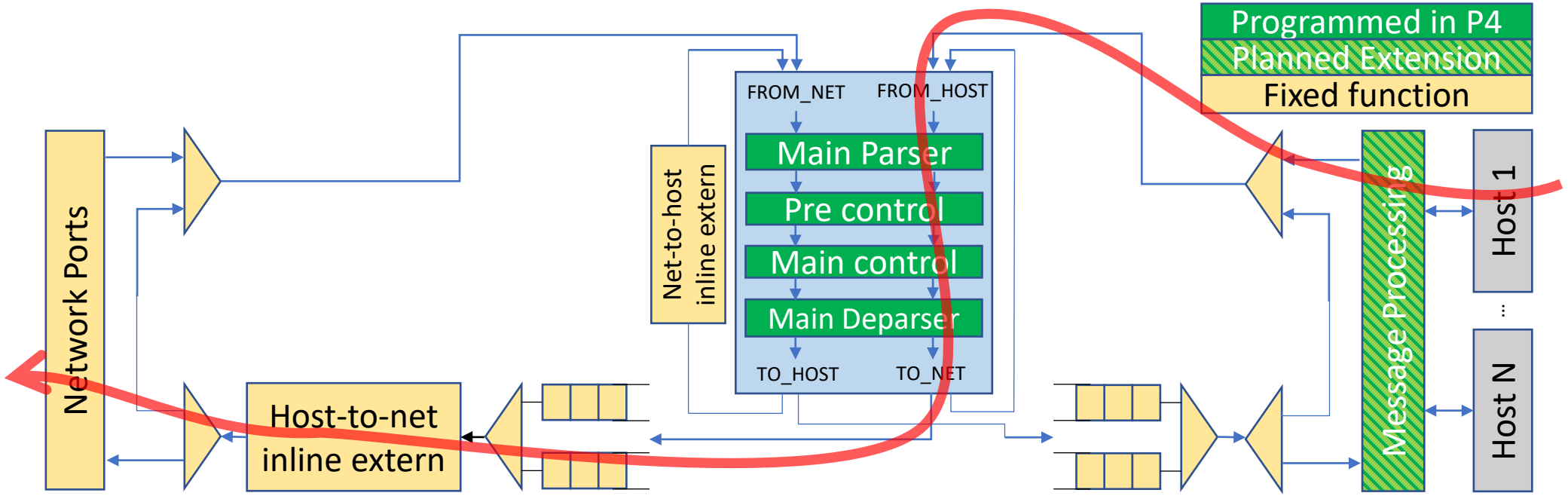
- Port-to-port packet starts as net-to-host
  - a table action sets destination to Ethernet port configured by control plane
  - then loop back in host and become host-to-net packet
- Standard metadata field **`direction`** can be used in P4 code to process packets differently, if desired

Directionality



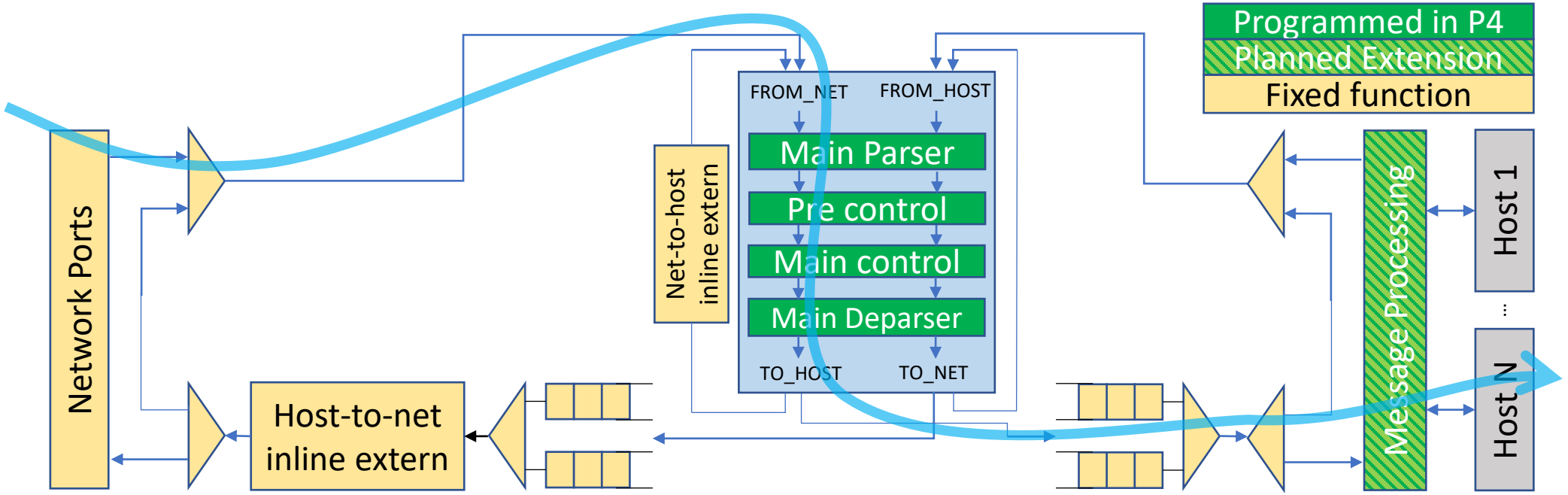
- VM-to-VM packet starts as host-to-net
  - a table action sets destination to a vport configured by control plane
  - then loop back near network ports and become net-to-host packet

TCP connection tracking



- Host-to-net TCP SYN packet searches in conntrack table for 5-tuple key
  - Exact match: (IP src addr, IP dest addr, protocol, TCP src port, TCP dest port)
- Get miss for first packet on flow
  - Data plane does add-on-miss at high rate, without control plane involved

TCP connection tracking



- Later TCP SYN+ACK packet response comes from other host
  - Look up in the same physical table the same key (almost)
    - except **swap** IP src/dest address, and TCP src/dest port
  - Hit -> keep packet, otherwise drop

TCP connection tracking

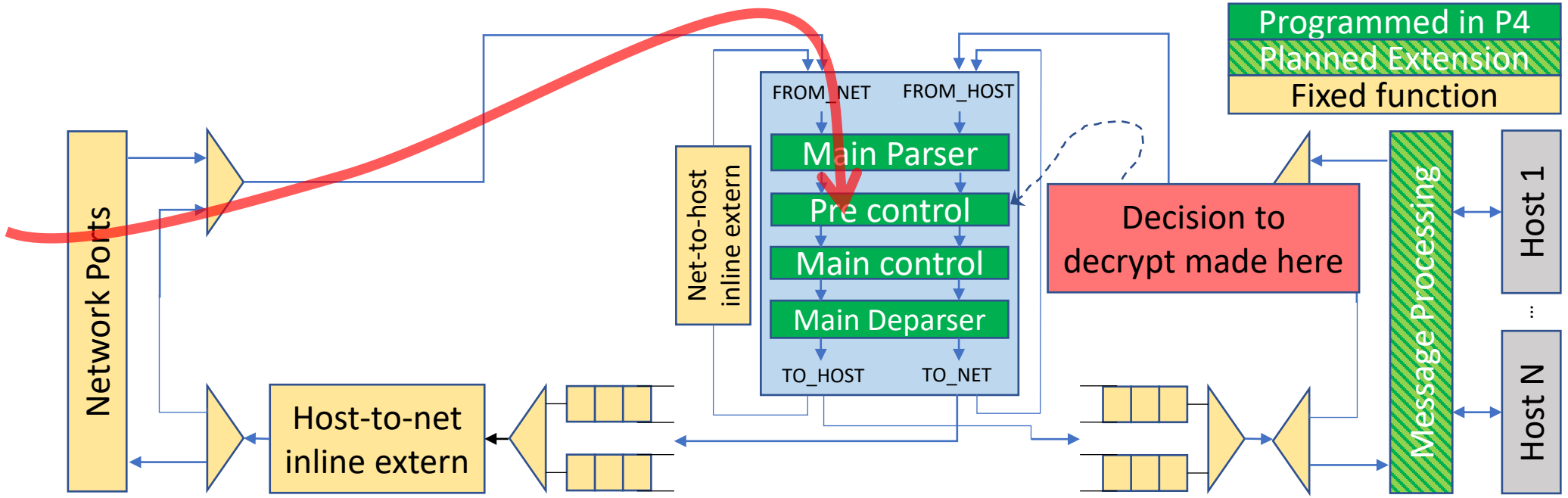
- Keep existing idle timeout option
  - Per-entry time-since-last-matched “entry age” maintained in data plane
  - Notify control plane when entry age reaches configured timeout value
  - Only the control plane can delete entries

TCP connection tracking

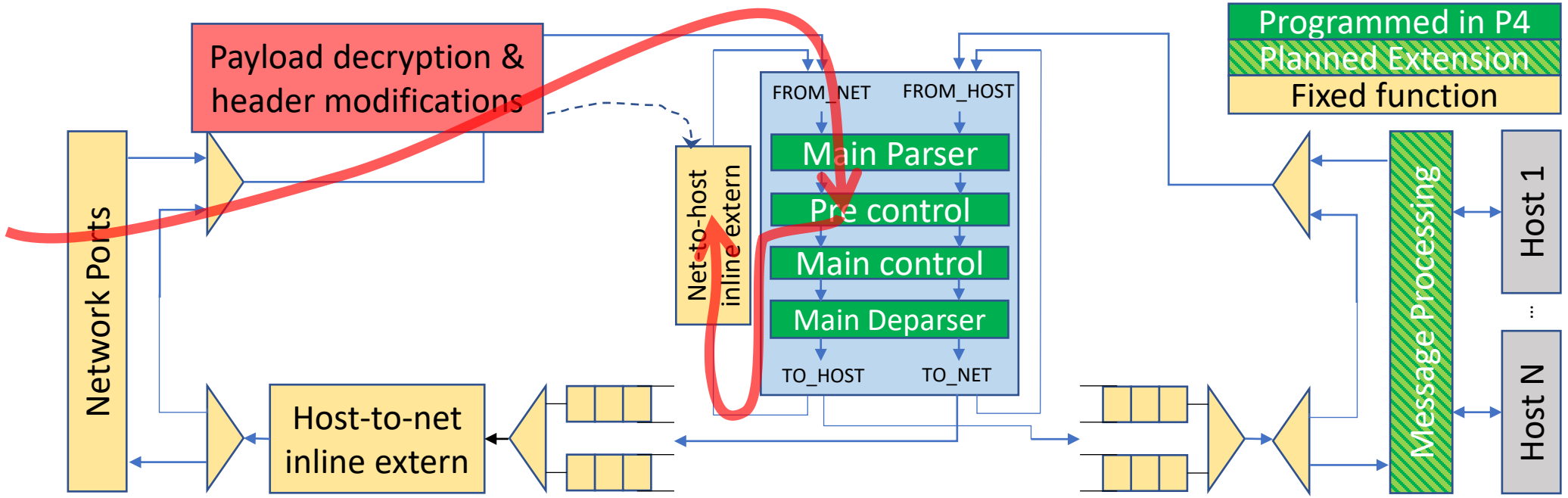
- Keep existing idle timeout option
  - Per-entry time-since-last-matched “entry age” maintained in data plane
  - Notify control plane when entry age reaches configured timeout value
  - Only the control plane can delete entries

- New option to delete entries in data plane
  - Same as above, except data plane deletes old entries itself
  - Control plane need not be burdened with potentially high delete rate

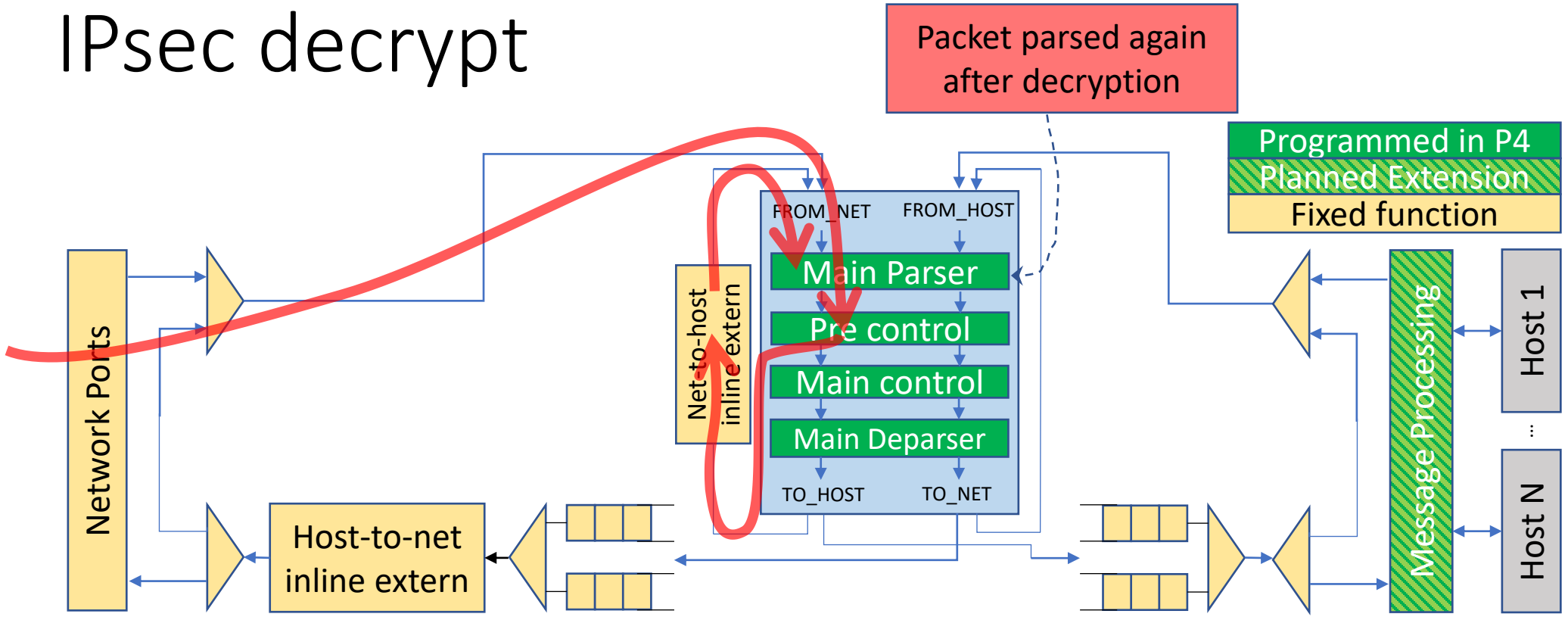
IPsec decrypt



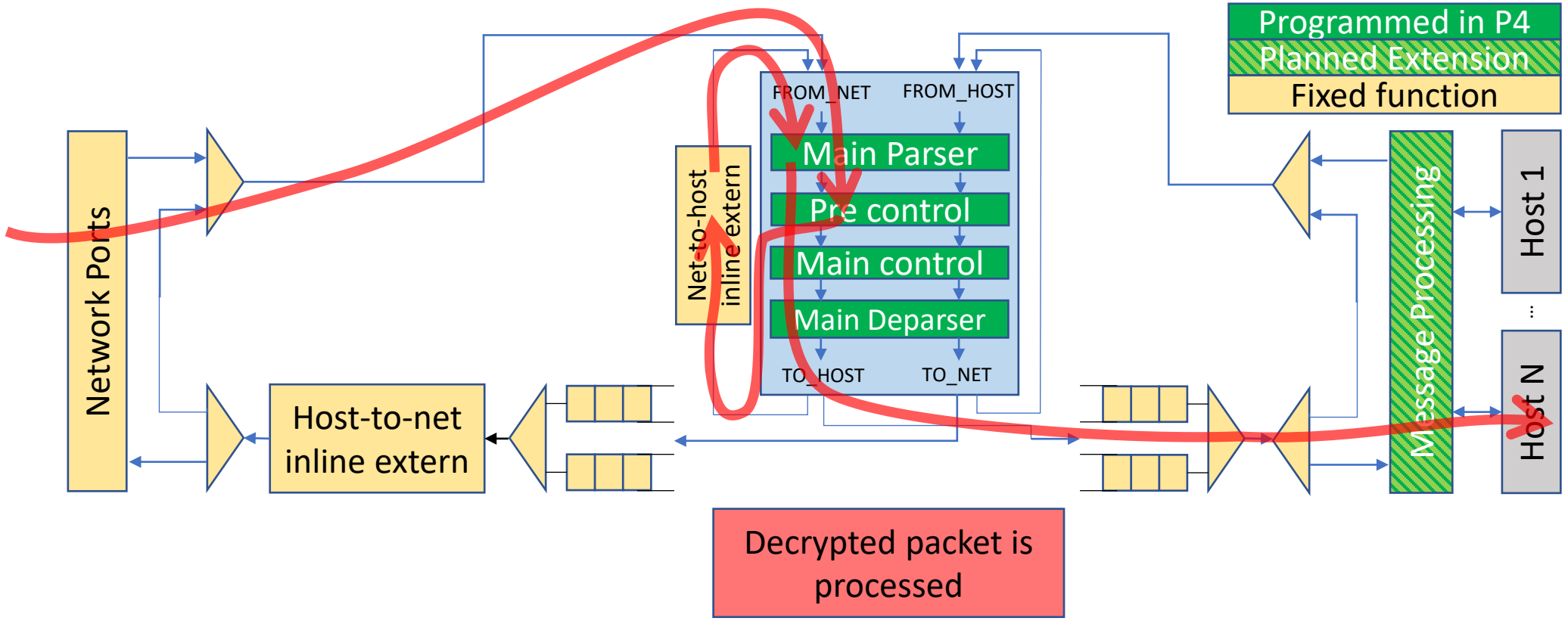
IPsec decrypt



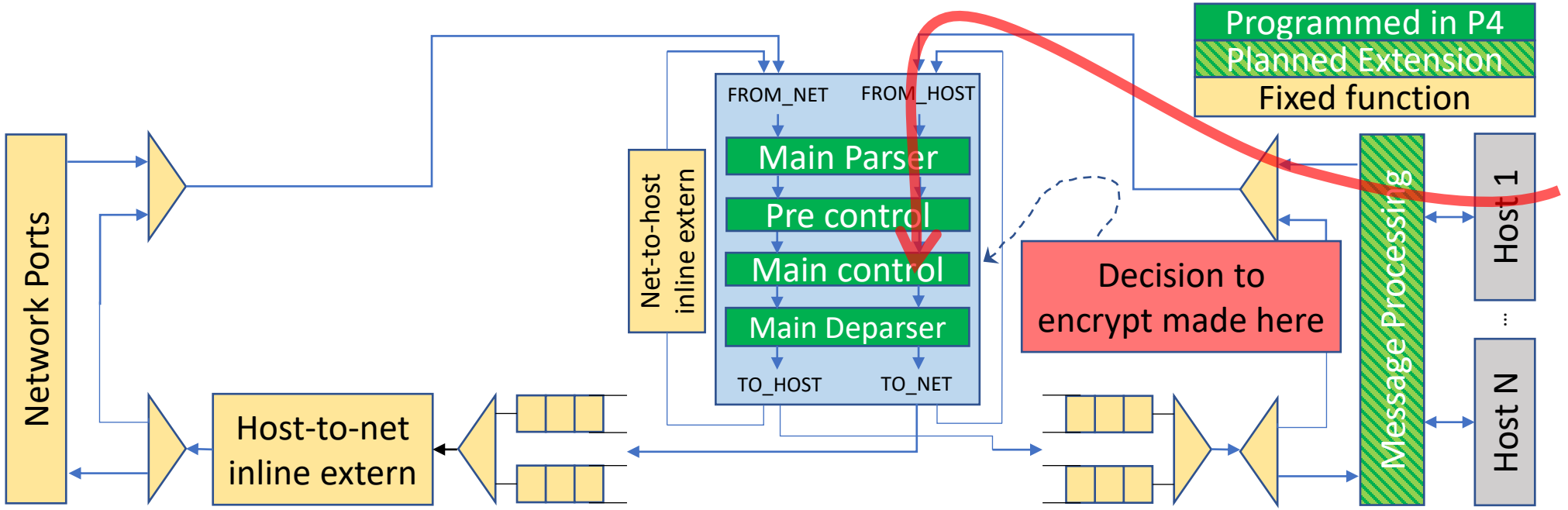
IPsec decrypt



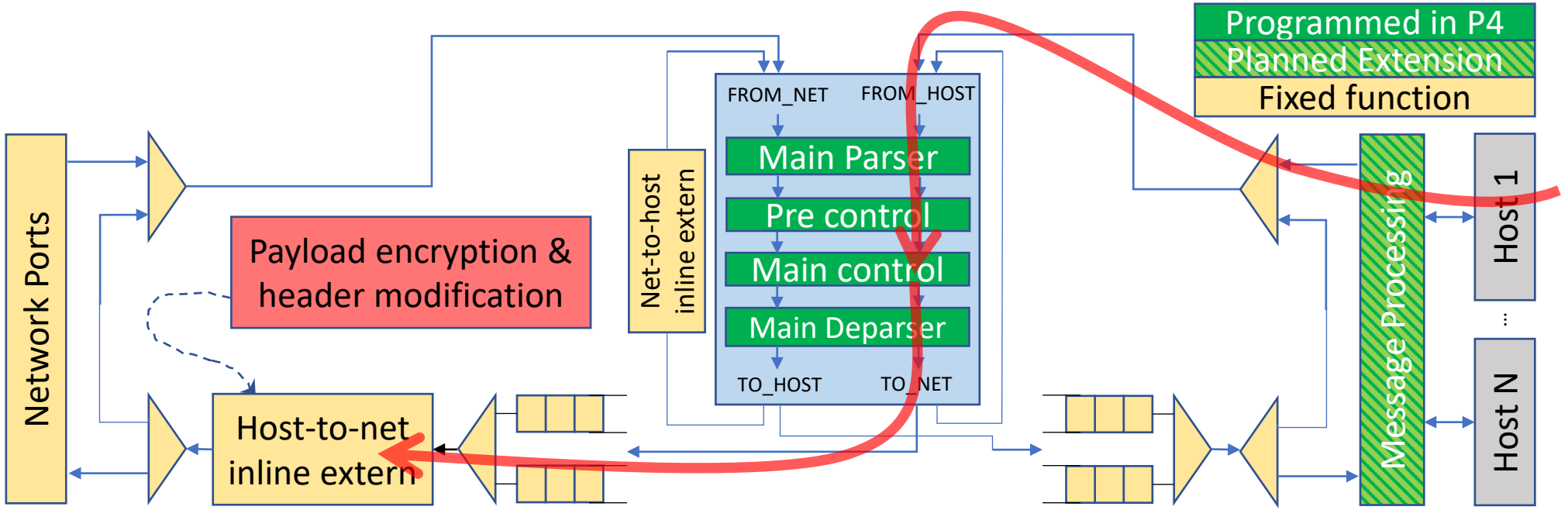
IPsec decrypt



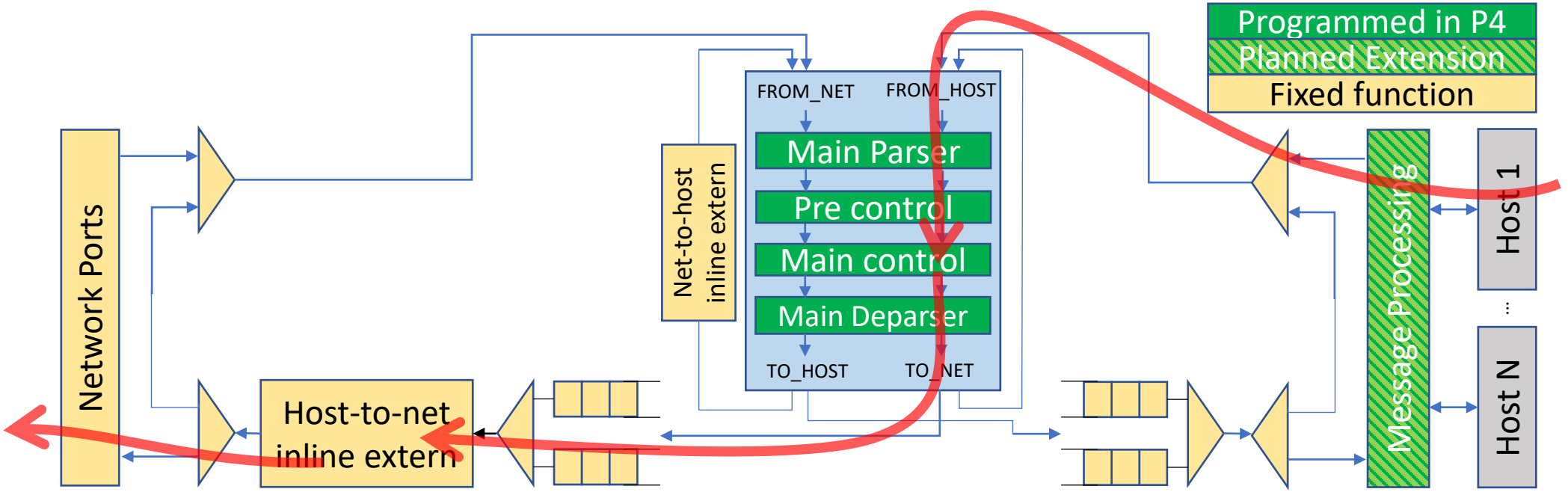
IPsec encrypt



IPsec encrypt



IPsec encrypt



What stays the same?

- Many externs defined in PSA spec used without change
  - Counters, meters, registers
  - action profiles, action selectors
  - hash, checksum, …

- Join p4-arch email list to see meeting invitations for:
  - P4 standard library
  - library of P4 header definitions
  - PSA implementation status
  - https://lists.p4.org

When will PNA be ready? What is left to do?

- Released PNA specification version 0.5 in May 2021
- Continue to work through more details of packet processing
- Message processing will take months of work
- Targeting release of version 1.0 for end of 2021

How can I help?

- Read and comment on the latest version of the spec:
  - https://github.com/p4lang/pna
- Thanks to these PNA work group members:
  - Boon Ang, VMware
  - Mario Baldi, Pensando
  - Gordon Brebner, Xilinx, co-chair P4.org Architecture WG
  - Dan Daly, Intel
  - Andy Fingerhut, Intel, co-chair P4.org Architecture WG
  - Nate Foster, Cornell University
  - Alan Lo, NVIDIA
  - Rip Sohan, Xilinx