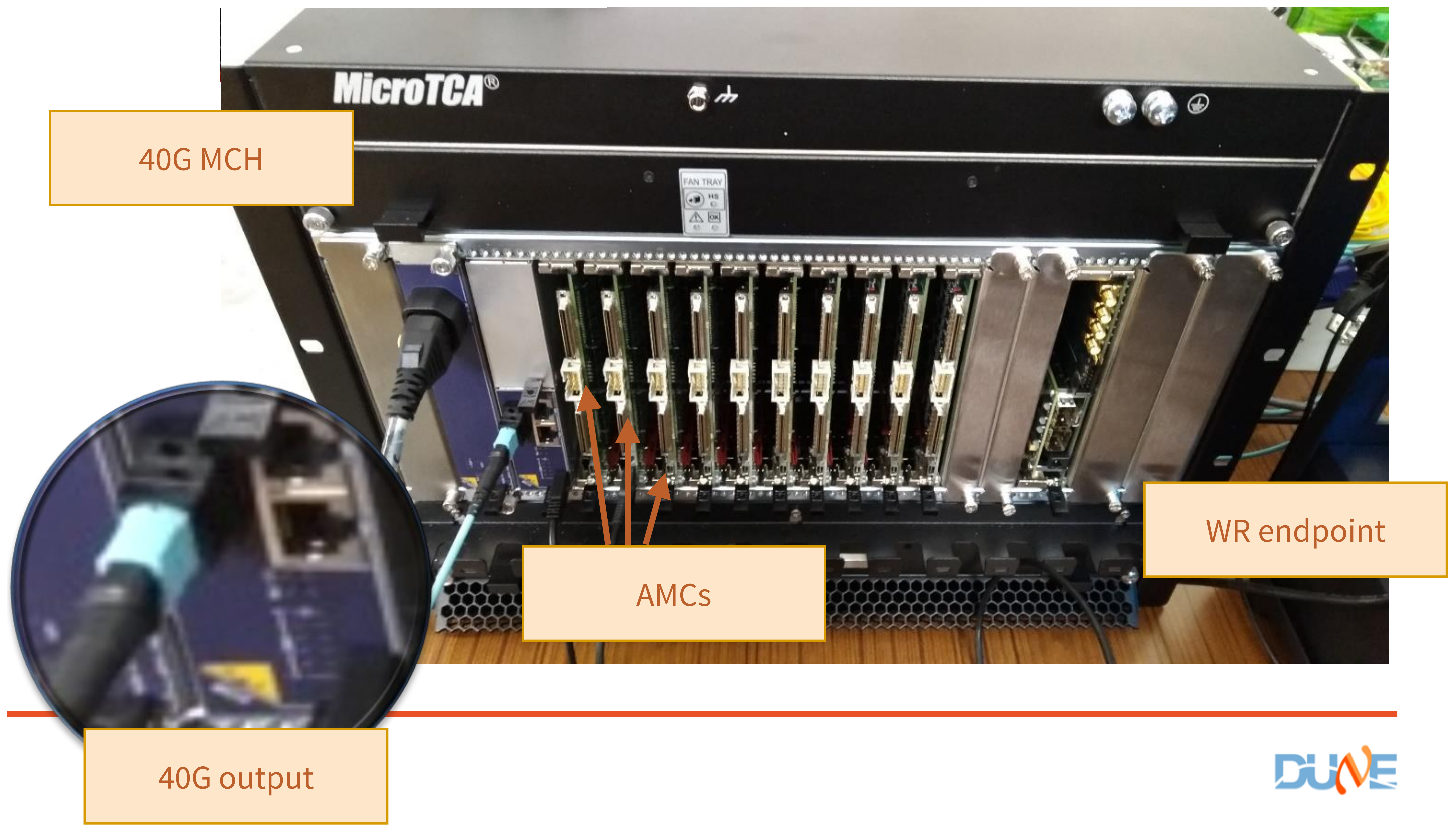
Ethernet Readout Progress

Jim Brooke

With contributions from Rob Halsall, Chris Lyford, Dave Newbold, Alessandro Thea

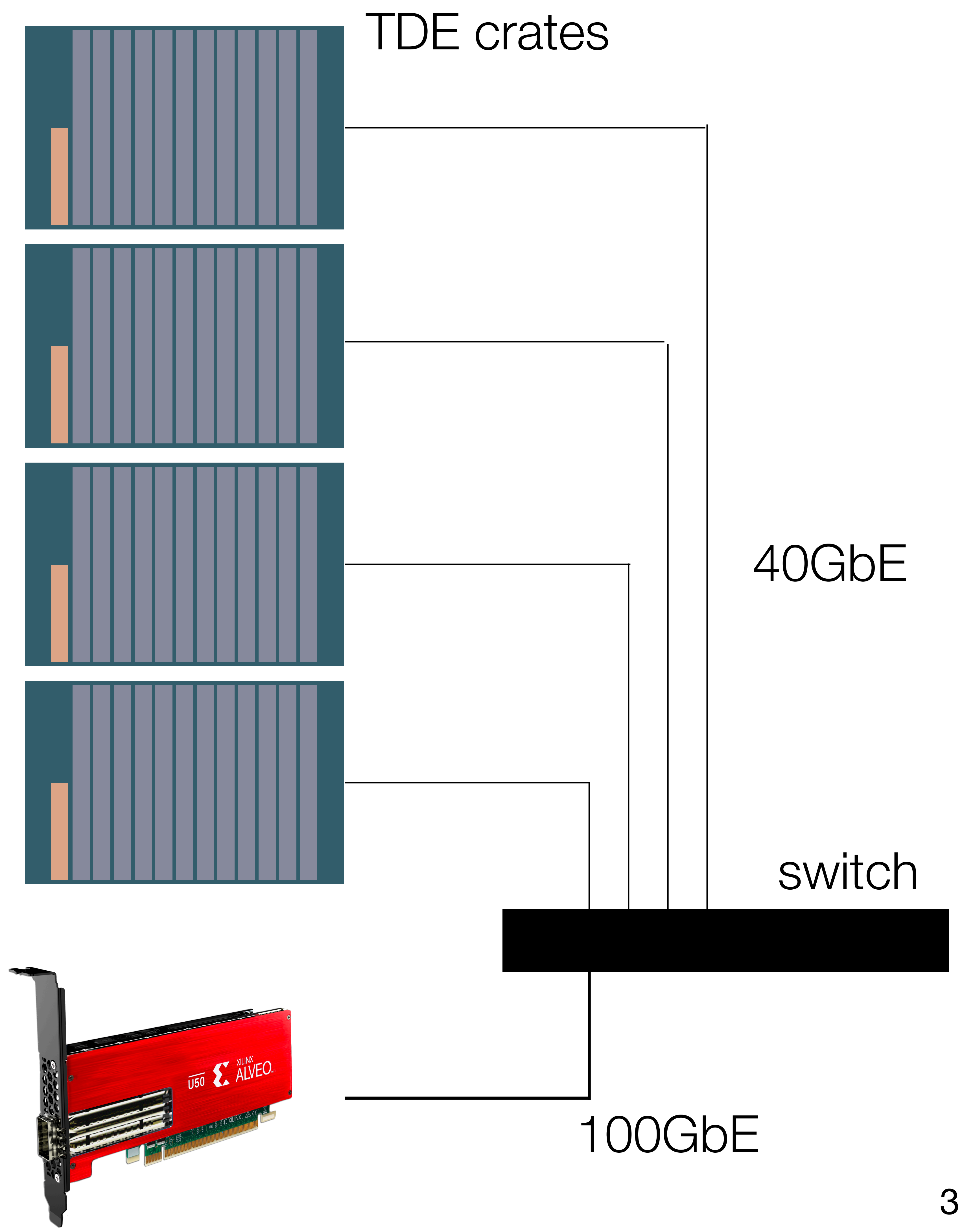
Motivation

- **Ethernet readout will be used throughout the DUNE Far Detector**
  - Use of COTS components eliminates the need for custom hardware
  - Removes significant risks
  - Should be cost neutral, or better
- **Example shown for VD TDE - where this originated**
  - Readout card sends data via 10GbE backplane
  - Aggregated by network switch in uTCA MCH
  - One 40 GbE link per crate at ~24 Gb/s utilisation
  - Further aggregation by switch : 4 crates onto 100GbE
  - Receive 100 GbE in COTS PCIe card / smart NIC
- **Other detectors will differ only in network topology**



Motivation

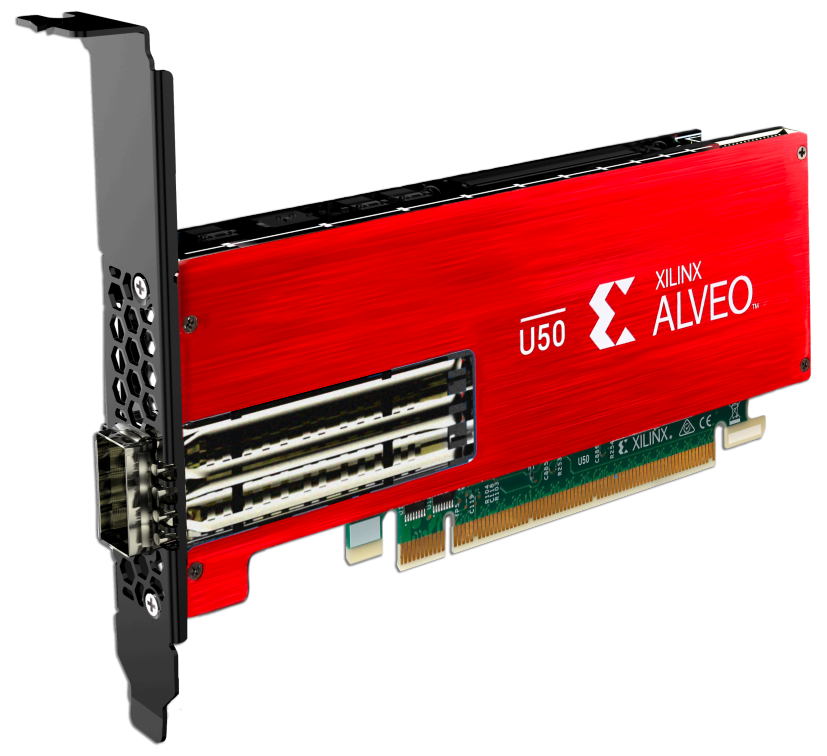
- **Ethernet readout will be used throughout the DUNE Far Detector**
  - Use of COTS components eliminates the need for custom hardware
  - Removes significant risks
  - Should be cost neutral, or better
- **Example shown for VD TDE - where this originated**
  - Readout card sends data onto ethernet backplane
  - Aggregated by network switch in uTCA MCH
  - One 40 GbE link per crate at ~24 Gb/s utilisation
  - Further aggregation by switch : 4 crates onto 100GbE
  - Receive 100 GbE in COTS PCIe card / smart NIC
- **Other detectors will differ only in network topology**

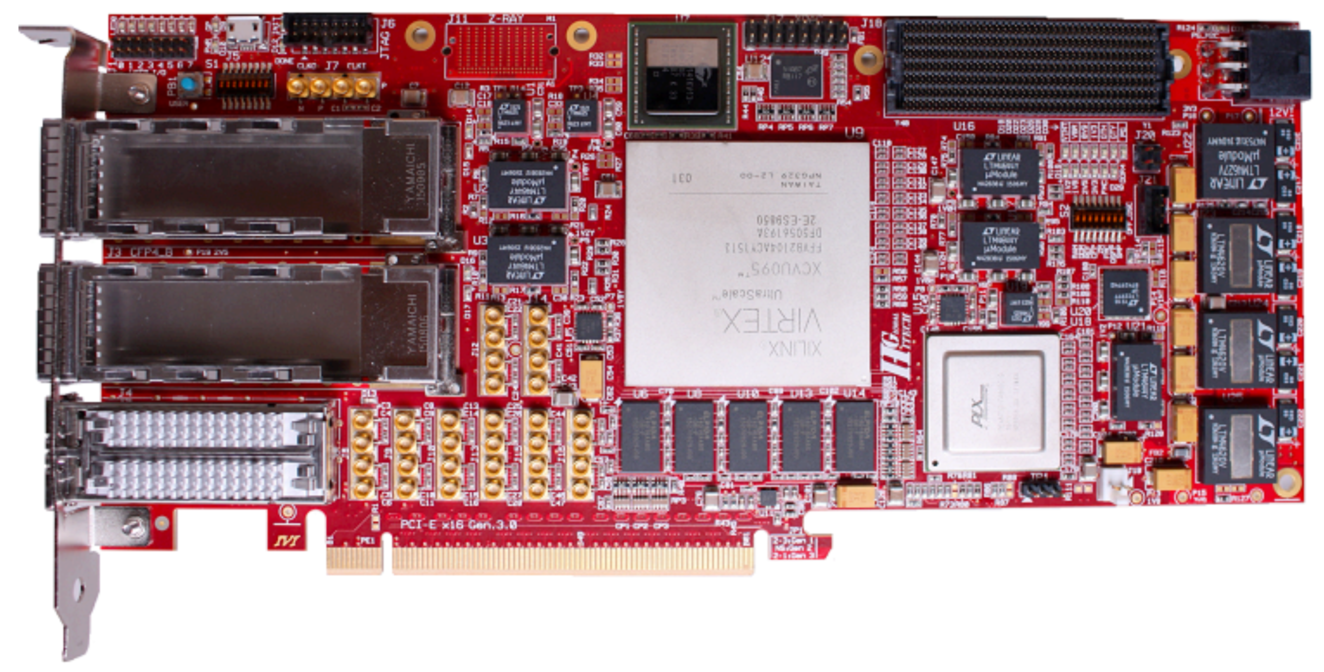


Readout Card

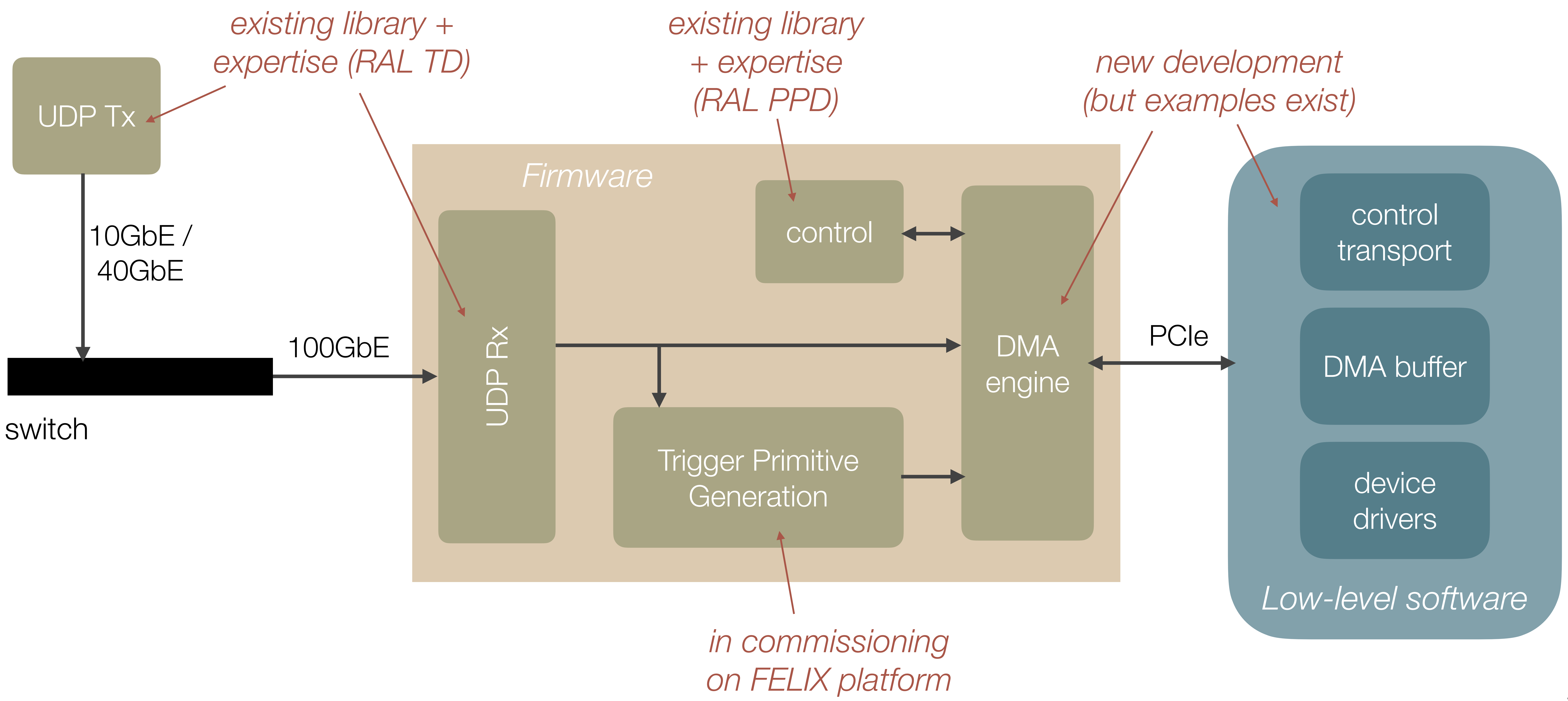
- Wide range of FPGA-based PCIe cards on the market
  - Large FPGA + O(100Gb/s) PCI bandwidth + O(1-200Gb/s external I/O)
  - Currently using **Xilinx Alveo U50** for development
  - Few £k per card

- Future procurement would involve market survey, requirements optimisation etc.

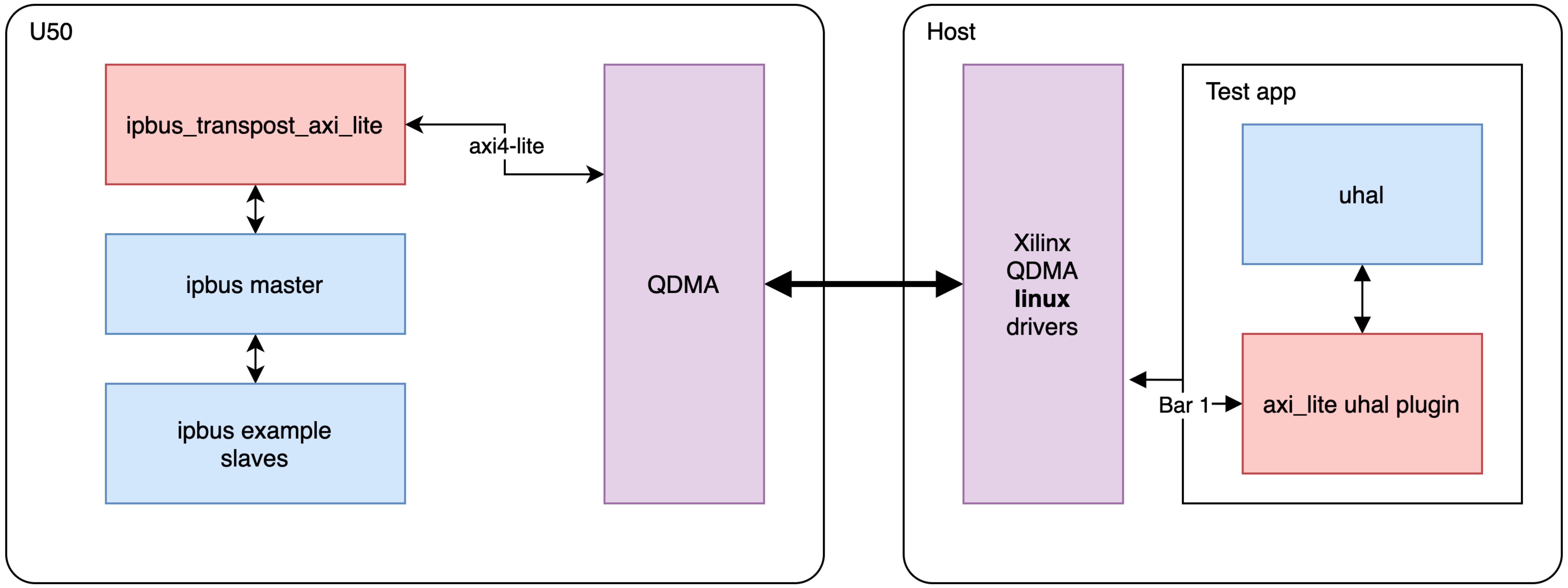




Firmware & Software



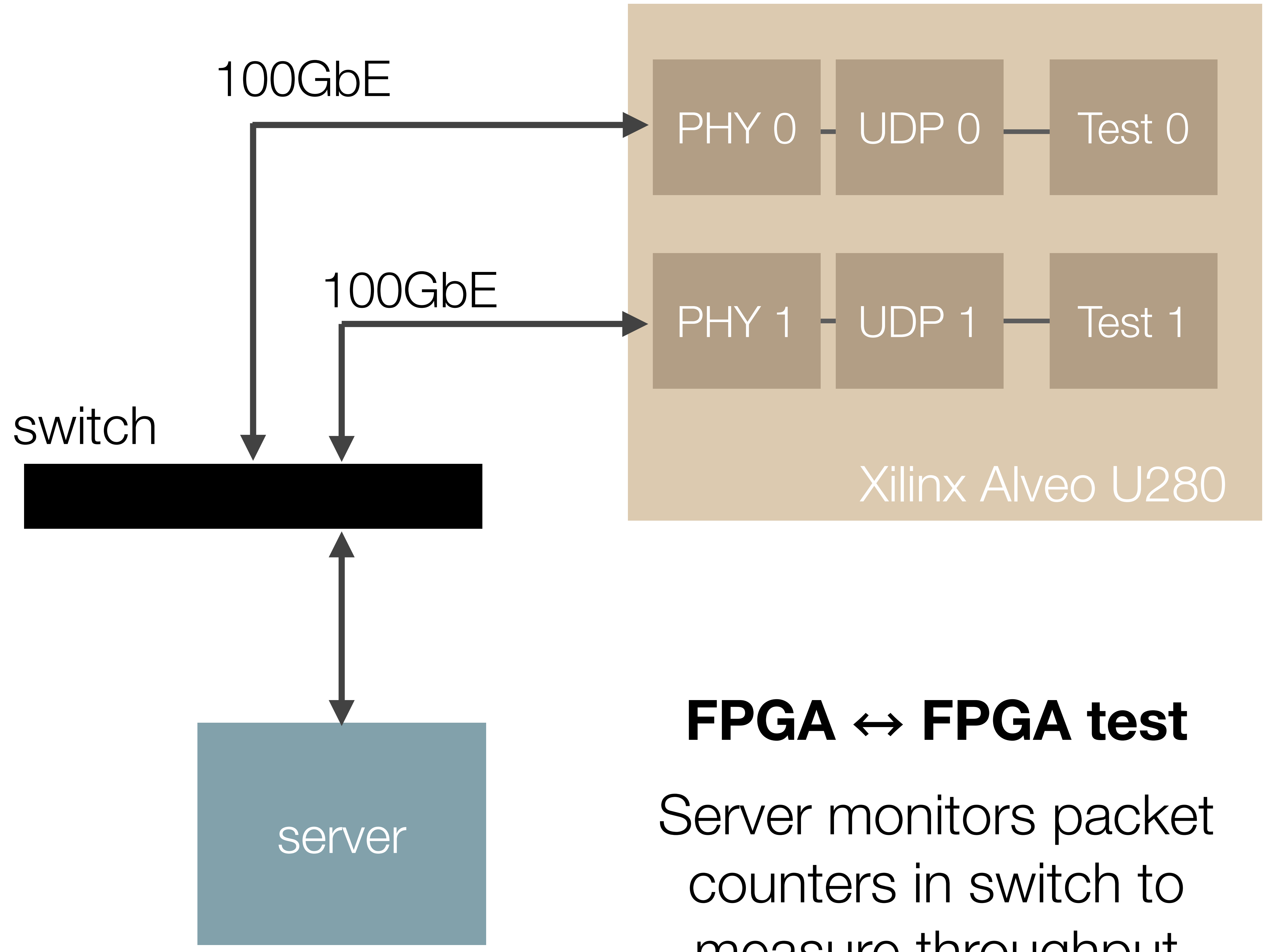
Control plane



- ipBus is a control protocol used widely in HEP (CMS, ATLAS, …) including DUNE TPG
  - Bridge axi4-lite control bus provided by XDMA to ipBus
  - Read/write established from host to registers in ipBus example slave

UDP Rx/Tx

- Developed by RAL TD for another project
  - Originally 10GbE, recently ported to 100GbE
  - Built and tested on Xilinx hardware - transfer rate **99.8 Gb/s** overnight w/o packet loss
- Significant work done to integrate UDP library into DUNE build environment
  - 10GbE as test data source
  - 100GbE data reception in U50



FPGA ↔ FPGA test

Server monitors packet counters in switch to measure throughput

DMA engine

- **Investigated use of Xilinx QDMA core - queue-based DMA engine**
  - Some features map well onto our use-case
  - But - fragile build, driver issues, high resource use, …
- **Switched to a homegrown design based on Xilinx XDMA**
  - Design features inspired by FELIX and other applications
  - Operates in ‘continuous DMA’ mode
  - Incoming packets are chopped into fixed size transfers
    - Important for robustness against errors

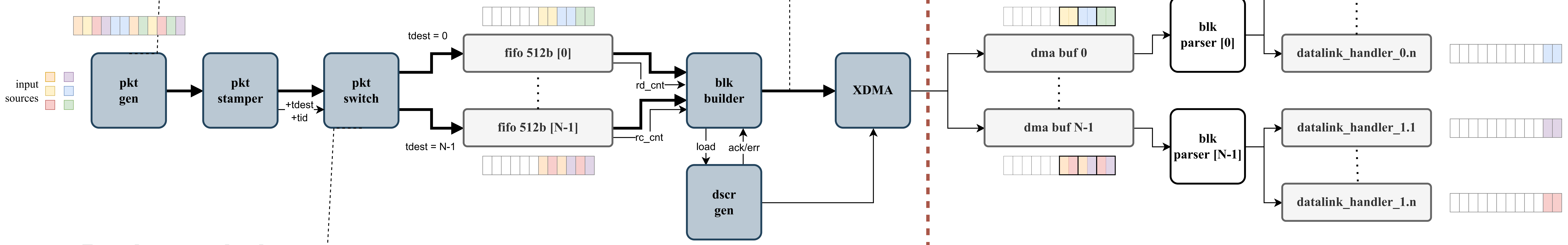
DMA engine

Firmware | *Software*

Packet generation
mimics multiple packet sources in network, variable size packets

Block builder
Maps packets onto fixed-size blocks, inserting headers & trailers for packet reconstruction

Block parser
Reconstruct network packets from fixed-size blocks



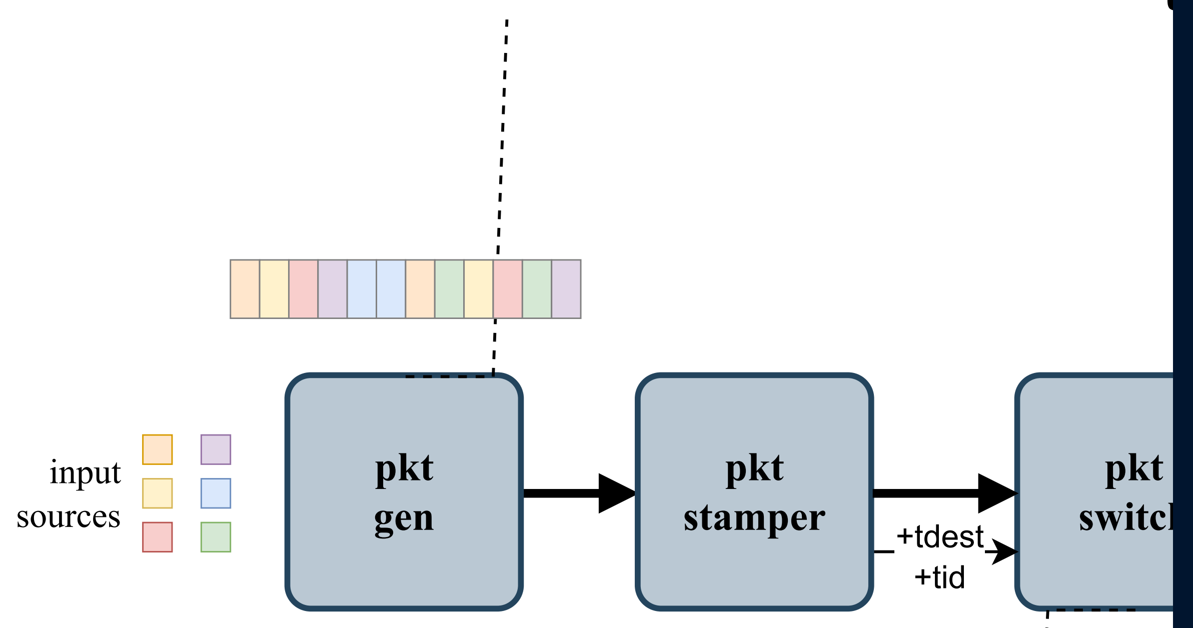
Packet switch
Sends packets to buffers depending on source ID

XDMA
Xilinx DMA block. Sends fixed sized blocks to host memory buffers

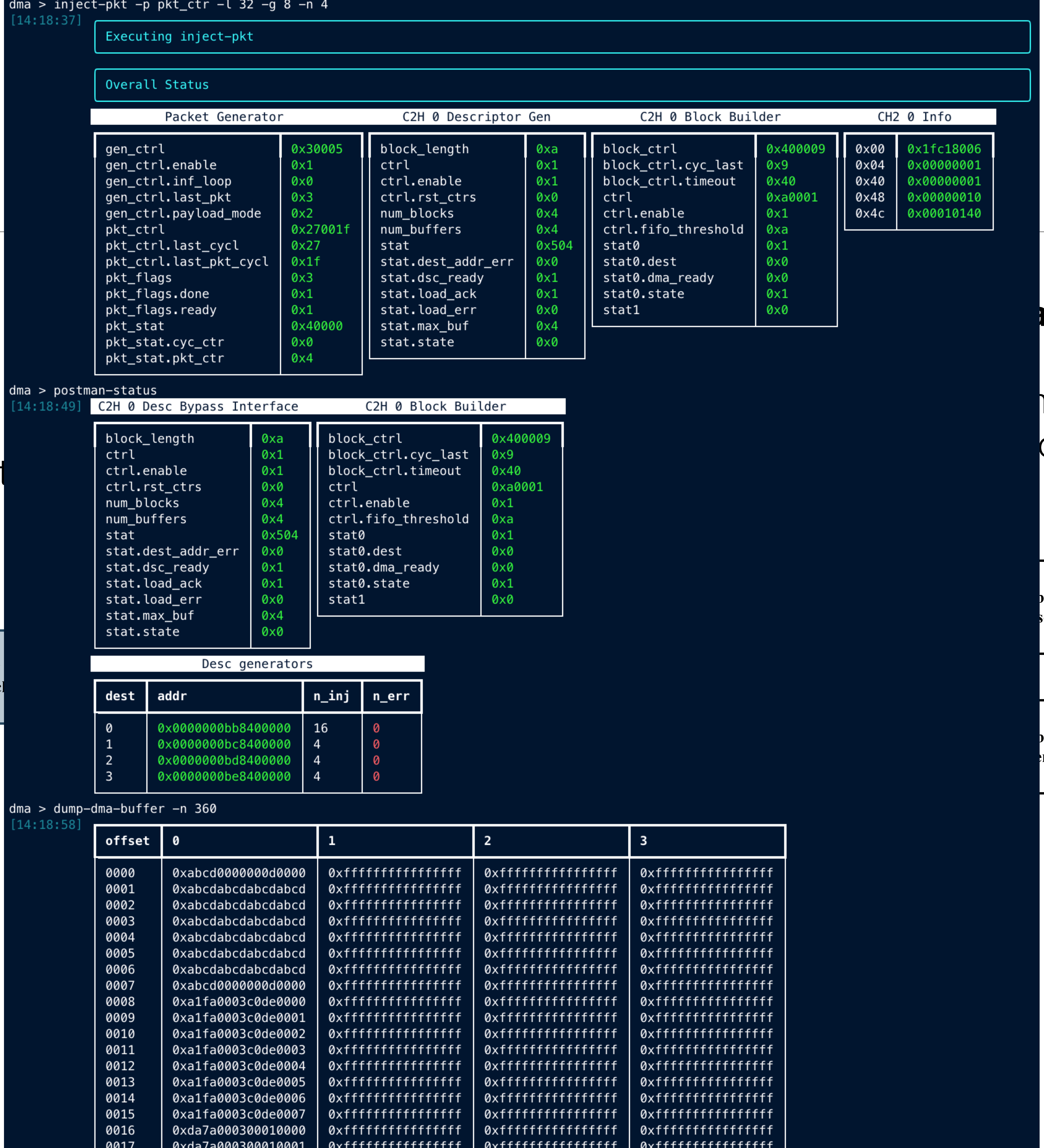
Final buffers
Reconstructed network packets stored in one buffer per source ID

DMA engine

Packet generation
mimics multiple packet sources in network, variable size packets

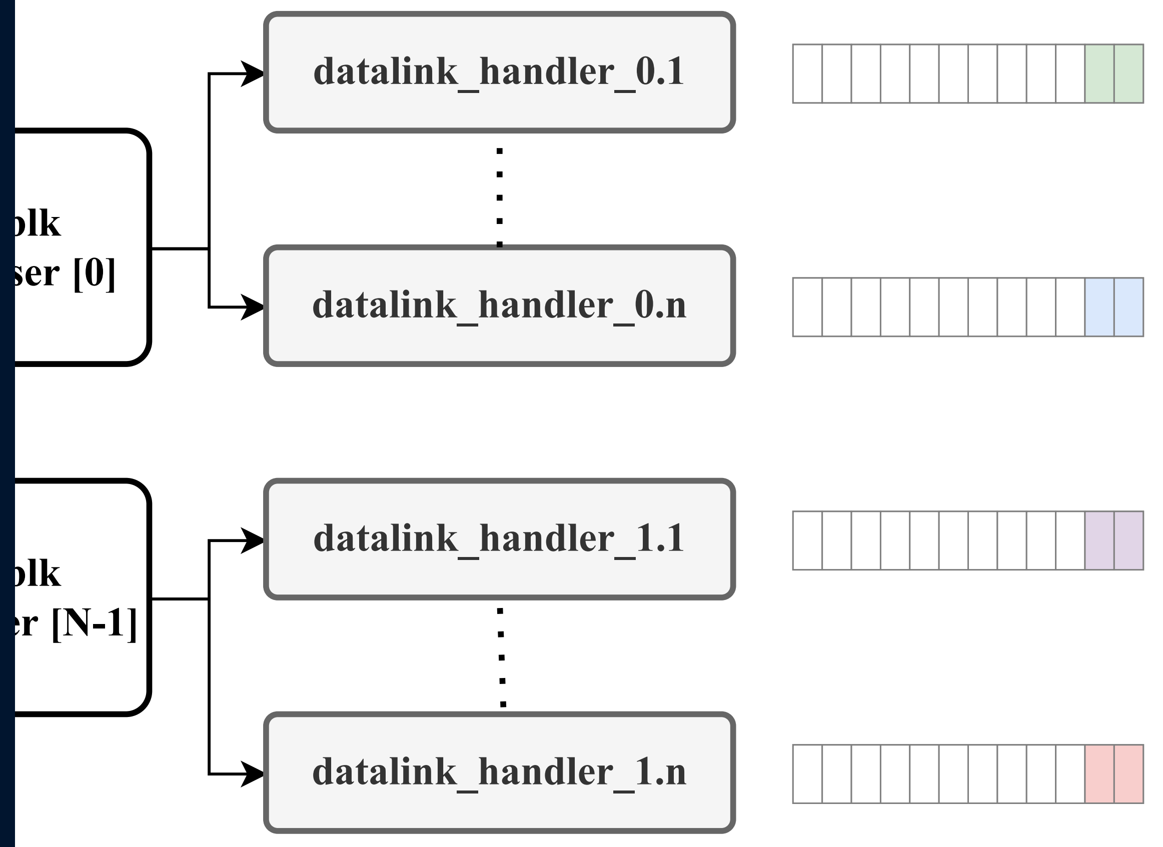


Packet switch
Sends packets to buffers depending on source ID



```
dma > inject-pkt -p pkt_ctr -l 32 -g 8 -n 4
[14:18:37]
  Executing inject-pkt

  Overall Status

         Packet Generator                  C2H 0 Descriptor Gen        C2H 0 Block Builder               CH2 0 Info

  gen_ctrl                  0x30005     block_length         0xa     block_ctrl               0x400009   0x00  0x1fc18006
  gen_ctrl.enable           0x1         ctrl                 0x1     block_ctrl.cyc_last      0x9        0x04  0x00000001
  gen_ctrl.inf_loop         0x0         ctrl.enable          0x1     block_ctrl.timeout       0x40       0x40  0x00000001
  gen_ctrl.last_pkt         0x3         ctrl.rst_ctrs        0x0     ctrl                     0xa0001    0x48  0x00000010
  gen_ctrl.payload_mode     0x2         num_blocks           0x4     ctrl.enable              0x1        0x4c  0x00010140
  pkt_ctrl                  0x27001f    num_buffers          0x4     ctrl.fifo_threshold      0xa
  pkt_ctrl.last_cycl        0x27        stat                 0x504   stat0                    0x1
  pkt_ctrl.last_pkt_cycl    0x1f        stat.dest_addr_err   0x0     stat0.dest               0x0
  pkt_flags                 0x3         stat.dsc_ready       0x1     stat0.dma_ready          0x0
  pkt_flags.done            0x1         stat.load_ack        0x1     stat0.state              0x1
  pkt_flags.ready           0x1         stat.load_err        0x0     stat1                    0x0
  pkt_stat                  0x40000     stat.max_buf         0x4
  pkt_stat.cyc_ctr          0x0         stat.state           0x0
  pkt_stat.pkt_ctr          0x4

dma > postman-status
[14:18:49]  C2H 0 Desc Bypass Interface       C2H 0 Block Builder

  block_length          0xa     block_ctrl               0x400009
  ctrl                  0x1     block_ctrl.cyc_last      0x9
  ctrl.enable           0x1     block_ctrl.timeout       0x40
  ctrl.rst_ctrs         0x0     ctrl                     0xa0001
  num_blocks            0x4     ctrl.enable              0x1
  num_buffers           0x4     ctrl.fifo_threshold      0xa
  stat                  0x504   stat0                    0x1
  stat.dest_addr_err    0x0     stat0.dest               0x0
  stat.dsc_ready        0x1     stat0.dma_ready          0x0
  stat.load_ack         0x1     stat0.state              0x1
  stat.load_err         0x0     stat1                    0x0
  stat.max_buf          0x4
  stat.state            0x0

            Desc generators

  dest   addr                  n_inj   n_err

  0      0x0000000bb8400000    16      0
  1      0x0000000bc8400000    4       0
  2      0x0000000bd8400000    4       0
  3      0x0000000be8400000    4       0

dma > dump-dma-buffer -n 360
[14:18:58]
  offset   0                    1                    2                    3

  0000     0xabcd0000000d0000   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0001     0xabcdabcdabcdabcd   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0002     0xabcdabcdabcdabcd   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0003     0xabcdabcdabcdabcd   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0004     0xabcdabcdabcdabcd   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0005     0xabcdabcdabcdabcd   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0006     0xabcdabcdabcdabcd   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0007     0xabcd0000000d0000   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0008     0xa1fa0003c0de0000   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0009     0xa1fa0003c0de0001   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0010     0xa1fa0003c0de0002   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0011     0xa1fa0003c0de0003   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0012     0xa1fa0003c0de0004   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0013     0xa1fa0003c0de0005   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0014     0xa1fa0003c0de0006   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0015     0xa1fa0003c0de0007   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0016     0xda7a000300010000   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
  0017     0xda7a000300010001   0xffffffffffffffff   0xffffffffffffffff   0xffffffffffffffff
```

arser
network
m fixed-
cks

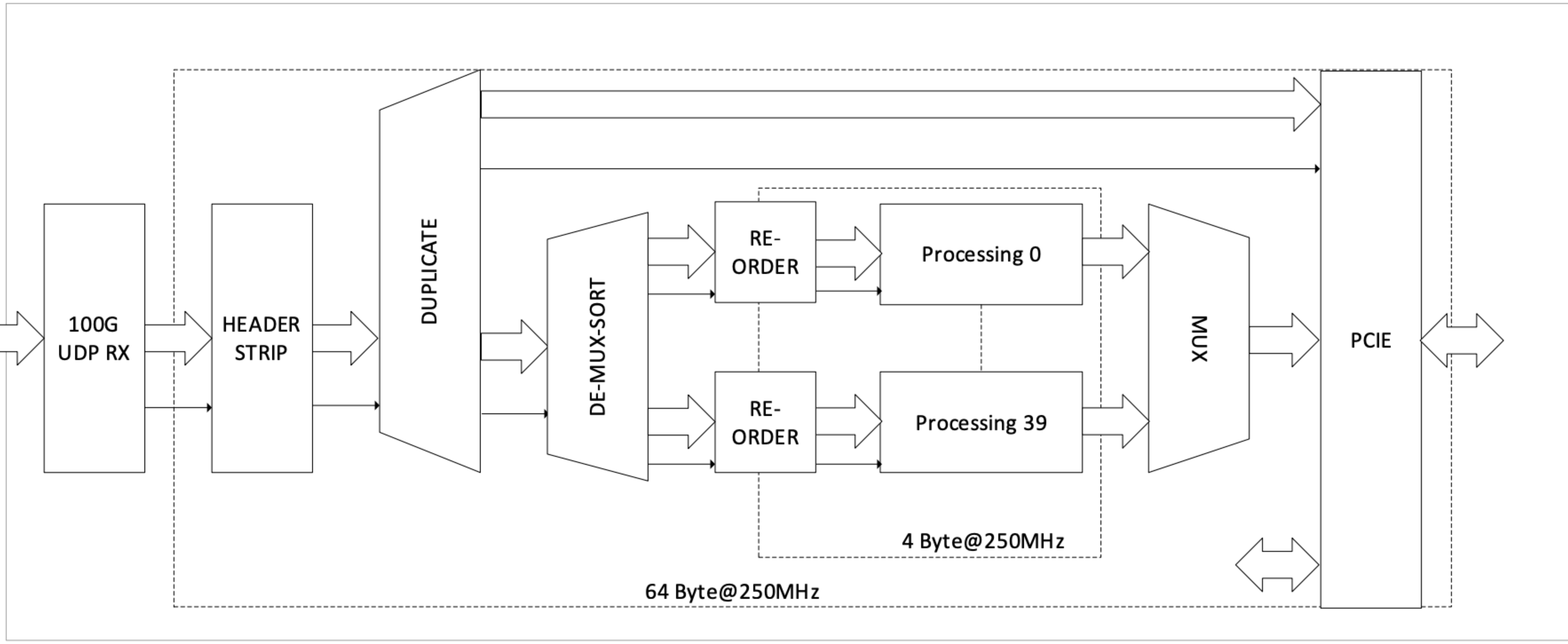


Final buffers
Reconstructed network packets stored in one buffer per source ID

.0

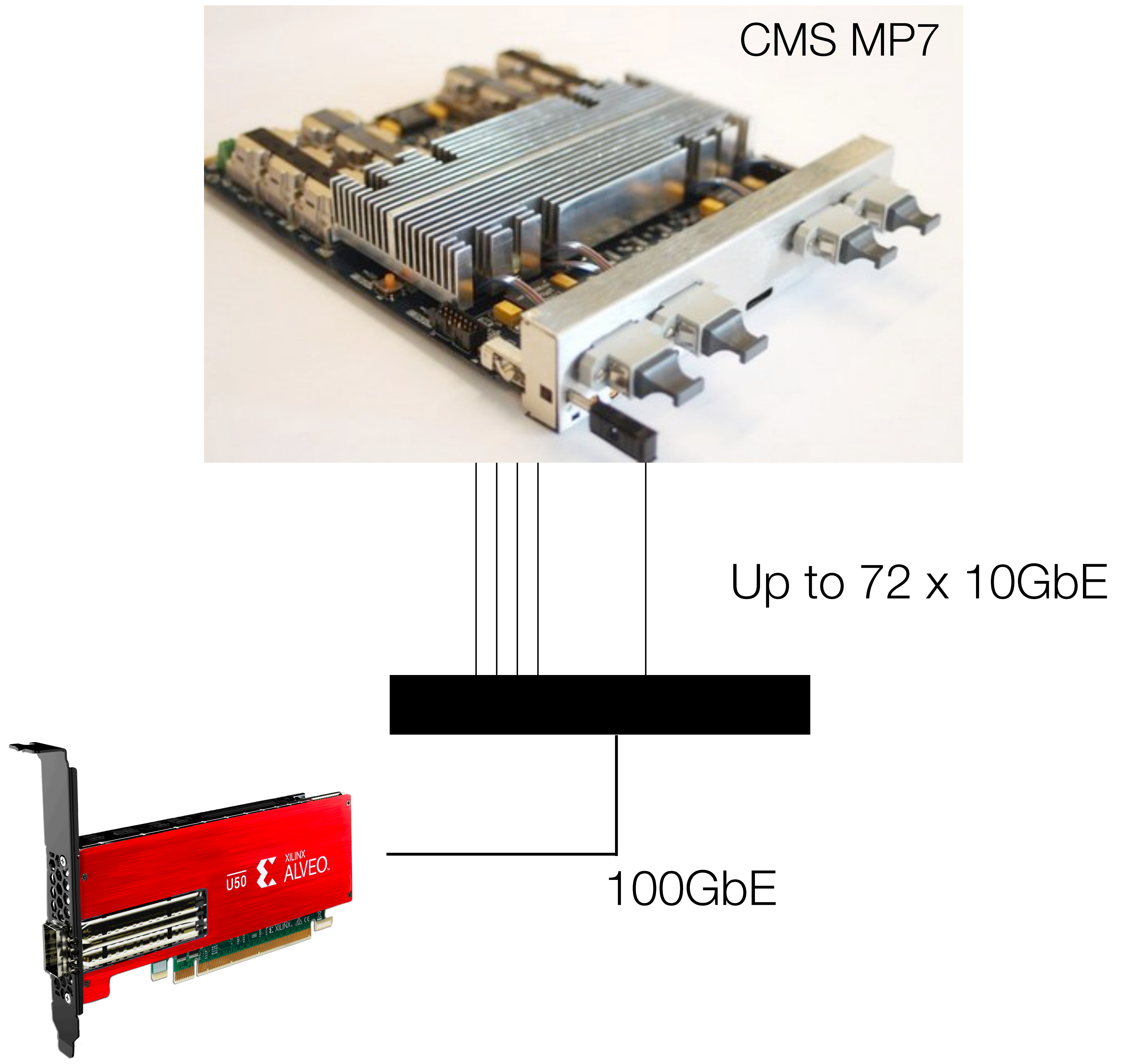
Trigger Primitive Generation

- Trigger primitives generated in pipeline processors as for FELIX TPG
- Routing challenge :
  - Receive packets on 512b bus from UDP core
  - Distribute each TDE channel to a specified trigger pipeline
- Data routing implemented using Xilinx AXI4 IP cores
  - Iteration on design currently in use with FELIX



Test Stand

- Need to mimic multiple front-end cards feeding network switch
  - Use a CMS MP7 as data-source
  - Replicate TPC readout :
    - 1 APA = 10 x 10 GbE links
    - 1 CRP = 12 x 10 GbE links
- Test stand in assembly in RAL TD lab
  - uTCA crate (on loan from timing system)
  - MP7 card
  - Mellanox switch
  - Patch panels
  - U50 + server



Status

- **Priority goal is to demonstrate readout path**
  - **MP7 ⟶ switch ⟶ U50 ⟶ DMA buffer ⟶ final SW buffers**
  - Key pieces are essentially ready for integration
    - UDP 10 GbE, UDP 100GbE, DMA engine, device drivers, very low level software
    - Work still needed on some monitoring/test components
- **Now need to start putting things together and testing in the lab**
  - Basic connectivity (ie low rate tests)
  - Increase complexity and data throughput
  - **Hope to have something ready for tests with VD CRP3 in Sept**
- **Integration of trigger path will wait until readout path demonstrated**

Summary

- **Ethernet readout will be used throughout DUNE Far Detector**
    - Removes the need for custom hardware production, and associated significant risks and resources
    - Should be cost neutral, or better
- **Initial prototype design has now been finalised**
    - Following initial studies of alternative options
    - All key components have been implemented
- **Now ready to start integration**
    - Later than expected, but we hope to have something ready for testing with VD CRP3 in Sept
    - Expect a more clear outlook on schedule in 1-2 weeks