Science & Technology Facilities Council

UK Research and Innovation

DUNE Ethernet Readout
100G UDP IP CORE

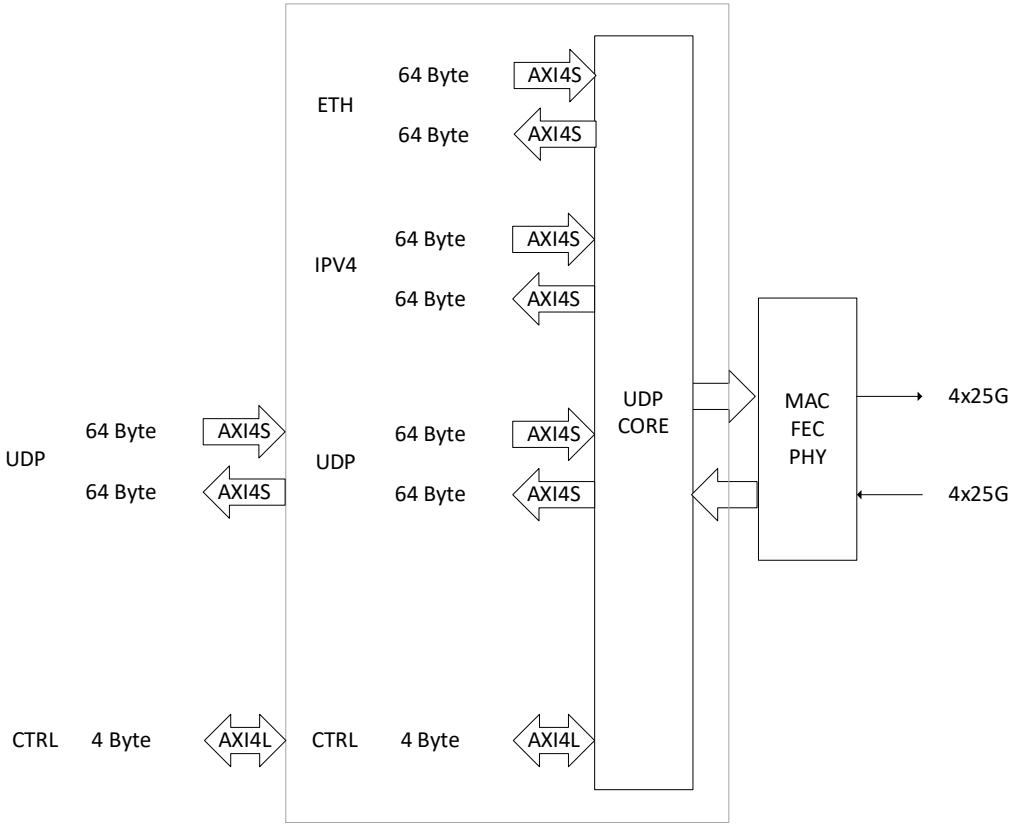
Rob Halsall

Matt Roberts

Chris Lyford

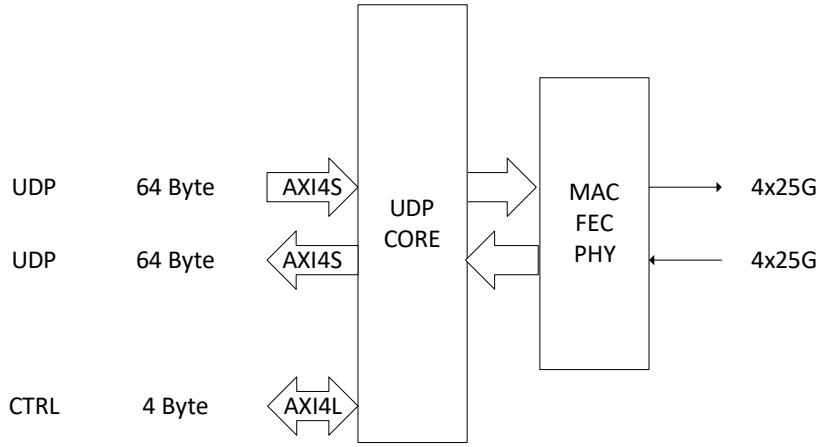
14-10-2021

100G UDP Core



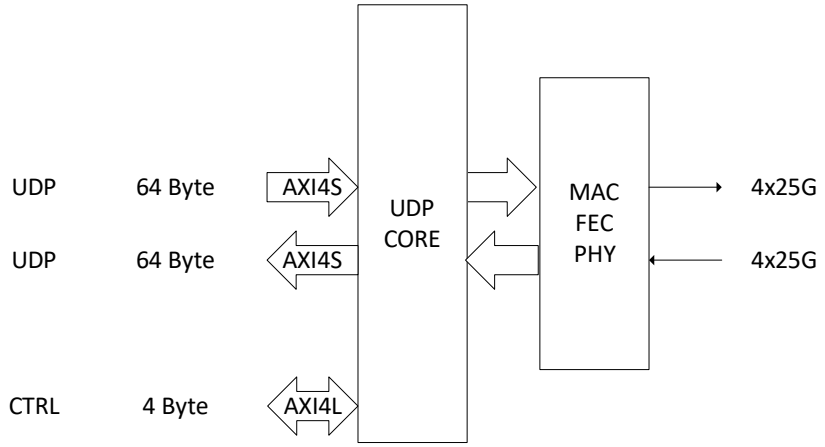
- Parameterised & Scalable
- UDP plus extendable – via ETH & IPV4 ports
- ARP & PING compile options
- Control Registers Mem Mapped via AXI4L
- UDP Tx & Rx via AXI4S
- Uses TKEEP to allow ‘any’ packet size
- Supports Jumbo frames
- Optional ‘Farm Mode’ on Tx Channel using TID
- Traffic Filter on Rx Channel
- Clocks - AXI4S ~195MHz & PHY ~322MHz
- Wrap to hide excess functionality

100G UDP Core



- AXI4S
- 64Byte @ 195MHz nominal
- AXI4S Tx data frame -> Outgoing UDP packet
- Incoming UDP packet -> AXI4S Rx data frame
- Loopback will return tx data frame on rx channel

100G UDP Core



- AXI4L
  xml file – flat memory map
- Approximately 32x 32 bit Control Registers
  - Source & destination addresses - mac/ip/port
  - Rx Filter settings
  - Stats registers
  - ….
- Tx Channel Farm Mode RAM
  - 256 entry mac/ip/port table
  - Selected by Tx AXI4S 8 bit TID signal
  - mac entries can be arp populated

100G UDP Core – Entity/generic

```
G_FPGA_VENDOR          : string  := "xilinx";       --! Selects the FPGA Vendor for Compilation
G_FPGA_FAMILY          : string  := "all";          --! Selects the FPGA Family for Compilation
G_FIFO_IMPLEMENTATION  : string  := "auto";         --! Selects how the Fitter implements the FIFO Memory
G_FIFO_TYPE            : string  := "inferred_mem"; --! Selects how to implement the Core's FIFOs
G_NUM_OF_ARP_POS       : natural := 32;             --! Number Of Positions In ARP Table
G_UDP_CLK_FIFOS        : boolean := false;          --! Generate Clk Crossing FIFOs At The I/O of UDP Payloads
G_EXT_CLK_FIFOS        : boolean := false;          --! Generate Clk Crossing FIFOs At The I/O of Unsupported Packet Types
G_TX_EXT_IP_FIFO_CAP   : integer := 2048;           --! Capacity In Bytes Of Uns IPV4 Protocol Packets FIFOs in Tx Path
G_TX_EXT_ETH_FIFO_CAP  : integer := 2048;           --! Capacity In Bytes Of Uns Ethernet Type Packets FIFOs in Tx Path
G_TX_OUT_FIFO_CAP      : integer := 2048;           --! Capacity Of FIFO at End Of Tx Path. Necessary For 100GbE, Less Important for 1/10GbE. Can Be 0.
G_RX_IN_FIFO_CAP       : integer := 2048;           --! Capacity Of FIFO at Start Of Rx Path. At least 16 Words Recommended.
G_RX_INPUT_PIPE_STAGES : integer := 1;              --! Pipeline Stages From External MAC/PHY To Rx Path
G_INC_PING             : boolean := true;           --! Generate Logic For Internal Ping Replies
G_INC_ARP              : boolean := true;           --! Generate Logic For Interal ARP Requests And Replies
G_CORE_FREQ_KHZ        : integer := 322262;         --! KHz Of Tx Path, Used For ARP Refresh Timers, Not Essential
G_INC_ETH              : boolean := false;          --! Generate Logic To Transmit Externally Provided Ethernet Payloads
G_INC_IPV4             : boolean := false           --! Generate Logic To Transmit Externally Provided IPV4 Payloads
```

- Porting across Vendor/families requires wrapped/configured/licensed mac/fec/phy

100G UDP Core – Entity/Ports/UDP AXI4S

```
-- AXI4-Stream UDP Transmit Data:
udp_axis_s_clk    : in std_logic;                       --! UDP Tx In Clk
udp_axis_s_reset  : in std_logic;                       --! UDP Tx In Reset
udp_axis_s_tdata  : in std_logic_vector(511 downto 0);  --! UDP Transmit Axi4s
udp_axis_s_tvalid : in std_logic;                       --! UDP Transmit Axi4s
udp_axis_s_tkeep  : in std_logic_vector(63 downto 0);   --! UDP Transmit Axi4s
udp_axis_s_tlast  : in std_logic;                       --! UDP Transmit Axi4s
udp_axis_s_tid    : in std_logic_vector(7 downto 0);    --! UDP Transmit Axi4s, route packets via LUT entry using this signal & Farm Mode
udp_axis_s_tuser  : in std_logic_vector(31 downto 0);   --! UDP Transmit Axi4s
udp_axis_s_tready : out std_logic;                      --! UDP Transmit Axi4s
-- AXI4-Stream UDP Receive Data:
udp_axis_m_clk    : in std_logic;                       --! UDP Rx Out Clk
udp_axis_m_reset  : in std_logic;                       --! UDP Rx Out Reset
udp_axis_m_tdata  : out std_logic_vector(511 downto 0); --! UDP Receive Axi4s
udp_axis_m_tvalid : out std_logic;                      --! UDP Receive Axi4s
udp_axis_m_tkeep  : out std_logic_vector(63 downto 0);  --! UDP Receive Axi4s
udp_axis_m_tlast  : out std_logic;                      --! UDP Receive Axi4s
udp_axis_m_tid    : out std_logic_vector(7 downto 0);   --! UDP Receive Axi4s
udp_axis_m_tuser  : out std_logic_vector(31 downto 0);  --! UDP Receive Axi4s
udp_axis_m_tready : in std_logic;
```

- Non-record structure AXI4S Wrapper – required for IP Packaging/Vitis

100G UDP Core – Entity/Ports/PHY AXI4S

```
tx_core_clk      : in std_logic;                      --! Tx Path Main Clock, Set To PHY Tx Clk or Set to Lower Clk and Buffer Full Packets Before PHY
rx_core_clk      : in std_logic;                      --! Rx Path Main Clock, can set to PHY Rx Clock, or any f down to ~200MHz
-- AXI4-Stream Slave Side (UDP Core Tx Data):
rx_axis_s_clk    : in std_logic;                      --! UDP Core Rx PHY Clock
rx_axis_s_rst    : in std_logic;                      --! UDP Core Rx Active-High Reset
rx_axis_s_tdata  : in std_logic_vector(511 downto 0); --! Rx Axi4s Signals From PHY
rx_axis_s_tvalid : in std_logic;                      --! Rx Axi4s Signals From PHY
rx_axis_s_tkeep  : in std_logic_vector(63 downto 0);  --! Rx Axi4s Signals From PHY
rx_axis_s_tlast  : in std_logic;                      --! Rx Axi4s Signals From PHY
rx_axis_s_tid    : in std_logic_vector(7 downto 0);   --! Rx Axi4s Signals From PHY (unused)
rx_axis_s_tuser  : in std_logic_vector(31 downto 0);  --! Rx Axi4s Signals From PHY (unused)
rx_axis_s_tready : out std_logic;                     --! Rx Path Backpressure, only used for debugging
-- AXI4 Stream Master Out (UDP Core Rx Data):
tx_axis_m_clk    : in std_logic;                      --! UDP Core Tx PHY Clock
tx_axis_m_rst    : in std_logic;                      --! UDP Core Tx Active-High Reset
tx_axis_m_tdata  : out std_logic_vector(511 downto 0); --! Tx Axi4s Signals To PHY
tx_axis_m_tvalid : out std_logic;                     --! Tx Axi4s Signals To PHY
tx_axis_m_tkeep  : out std_logic_vector(63 downto 0); --! Tx Axi4s Signals To PHY
tx_axis_m_tlast  : out std_logic;                     --! Tx Axi4s Signals To PHY
tx_axis_m_tid    : out std_logic_vector(7 downto 0);  --! Tx Axi4s Signals To PHY (unused)
tx_axis_m_tuser  : out std_logic_vector(31 downto 0); --! Tx Axi4s Signals To PHY (unused)
tx_axis_m_tready : in std_logic;                      --! Tx PHY Backpressure, USED
```

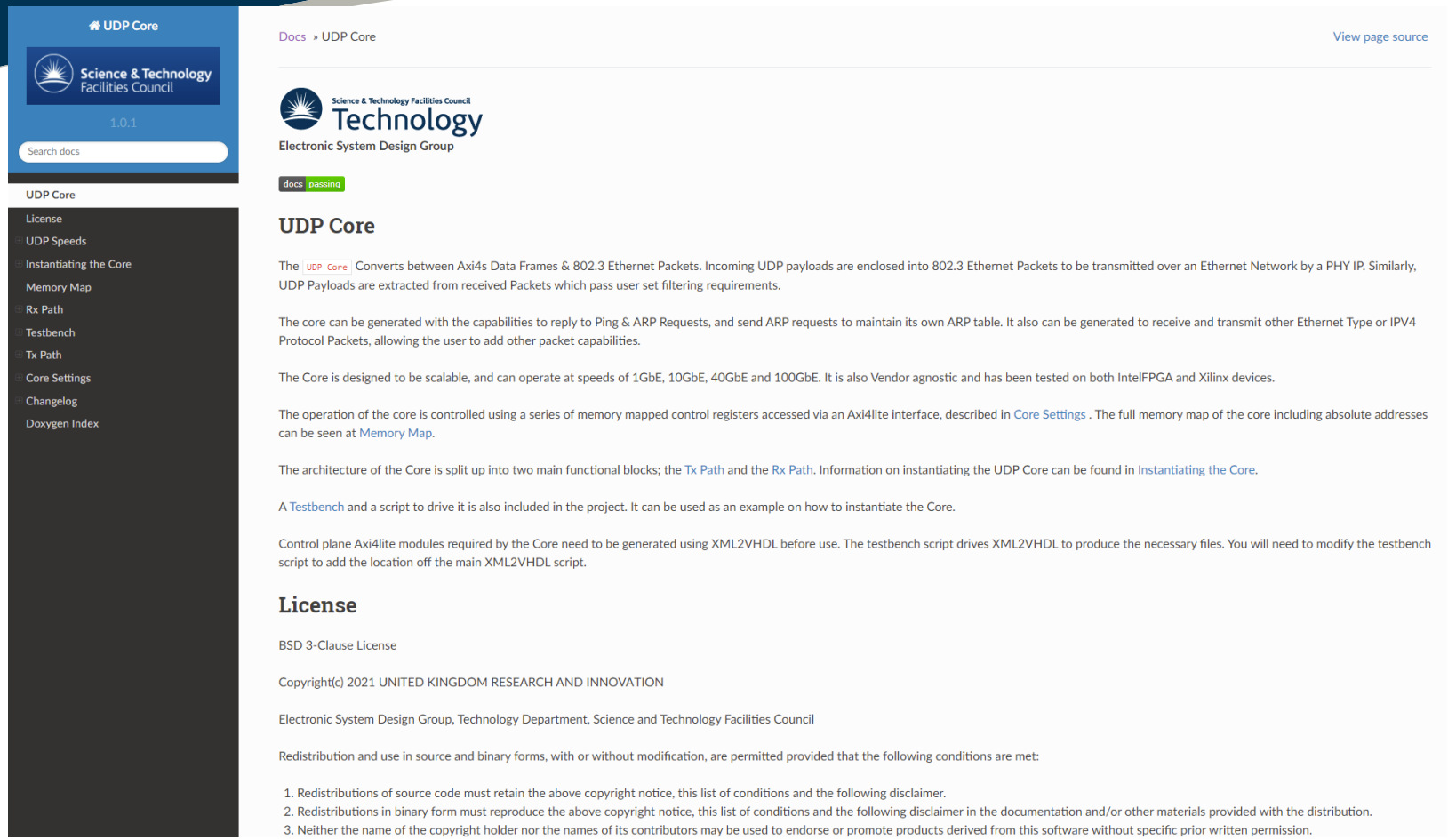
- Interface to Wrapped MAC/FEC/PHY
- Non-record structure AXI4S Wrapper – required for IP Packaging/Vitis

100G UDP Core – Entity/Ports/AXI4L

```
-- AXI4-Lite Interface (defined in axi4lite_pkg):
axi4lite_aclk    : in std_logic;                     --! AXI4-Lite Clock
axi4lite_aresetn : in std_logic;                     --! AXI4-Lite Active-Low Asynchronous Reset
axi4lite_araddr  : in std_logic_vector(31 downto 0); --! Axi4-Lite Signals
axi4lite_awaddr  : in std_logic_vector(31 downto 0); --! Axi4-Lite Signals
axi4lite_arvalid : in std_logic;                     --! Axi4-Lite Signals
axi4lite_awvalid : in std_logic;                     --! Axi4-Lite Signals
axi4lite_bready  : in std_logic;                     --! Axi4-Lite Signals
axi4lite_rready  : in std_logic;                     --! Axi4-Lite Signals
axi4lite_wdata   : in std_logic_vector(31 downto 0); --! Axi4-Lite Signals
axi4lite_wstrb   : in std_logic_vector(3 downto 0);  --! Axi4-Lite Signals
axi4lite_wvalid  : in std_logic;                     --! Axi4-Lite Signals
axi4lite_arready : out std_logic;                    --! Axi4-Lite Signals
axi4lite_awready : out std_logic;                    --! Axi4-Lite Signals
axi4lite_bresp   : out std_logic_vector(1 downto 0); --! Axi4-Lite Signals
axi4lite_bvalid  : out std_logic;                    --! Axi4-Lite Signals
axi4lite_rdata   : out std_logic_vector(31 downto 0);--! Axi4-Lite Signals
axi4lite_rresp   : out std_logic_vector(1 downto 0); --! Axi4-Lite Signals
axi4lite_rvalid  : out std_logic;                    --! Axi4-Lite Signals
axi4lite_wready  : out std_logic;                    --! Axi4-Lite Signals
```

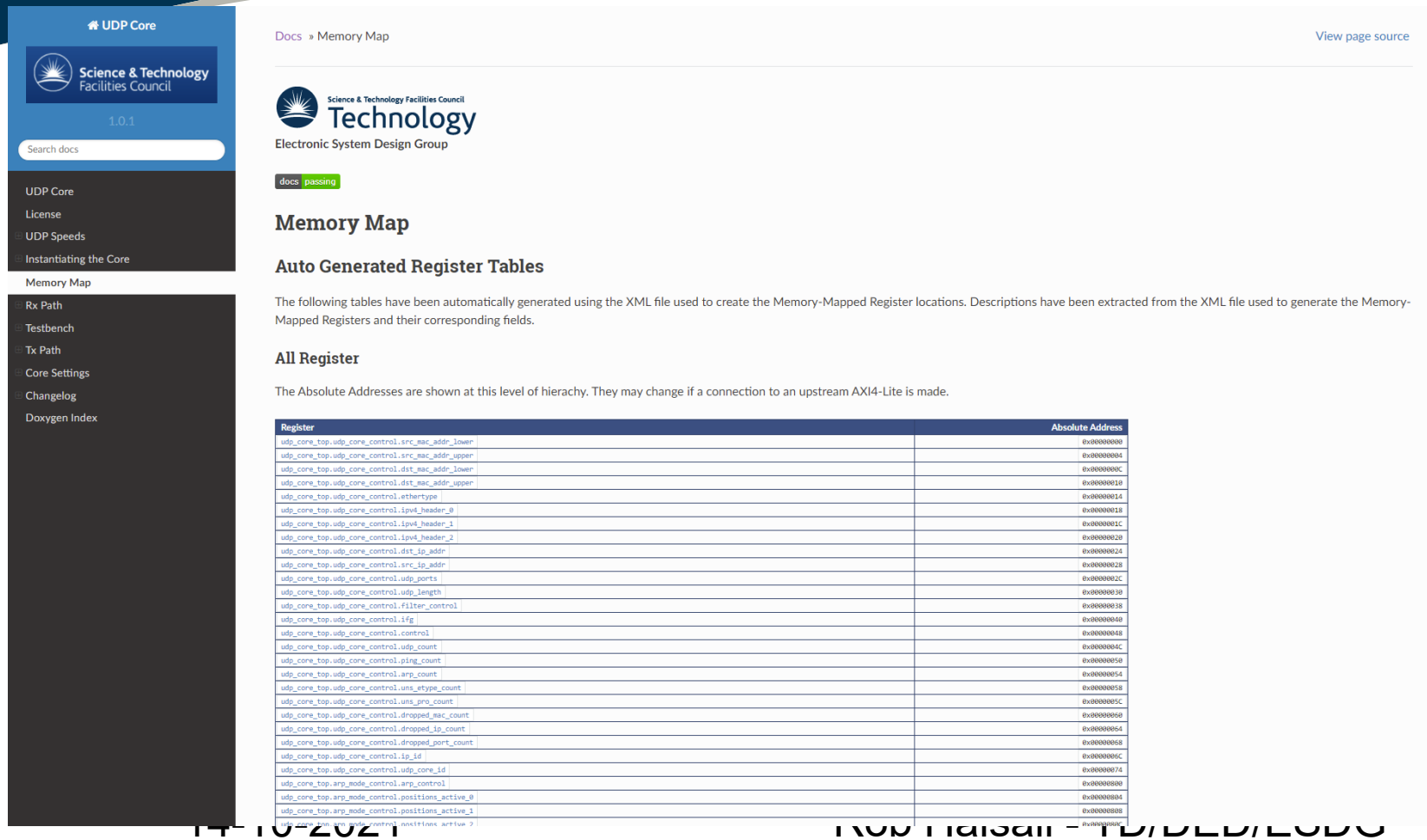
- Non-record structure AXI4L Wrapper – required for IP Packaging/Vitis

100G UDP Core



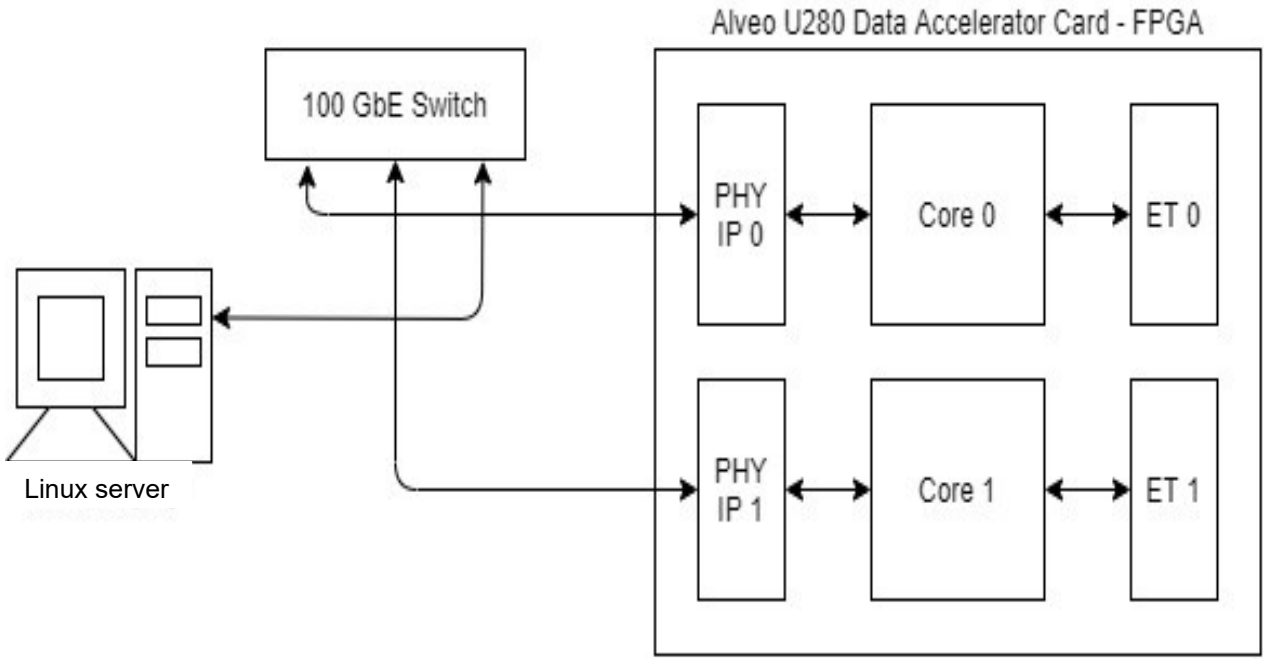
- Hosted on STFC Gitlab
- Automatic documentation
- Sphinx/Read the Docs/doxygen
- Mem Map Tables from XML
- Helper Scripts
- VUNIT Simulation
- BSD 3-Clause License
- packaged

100G UDP Core



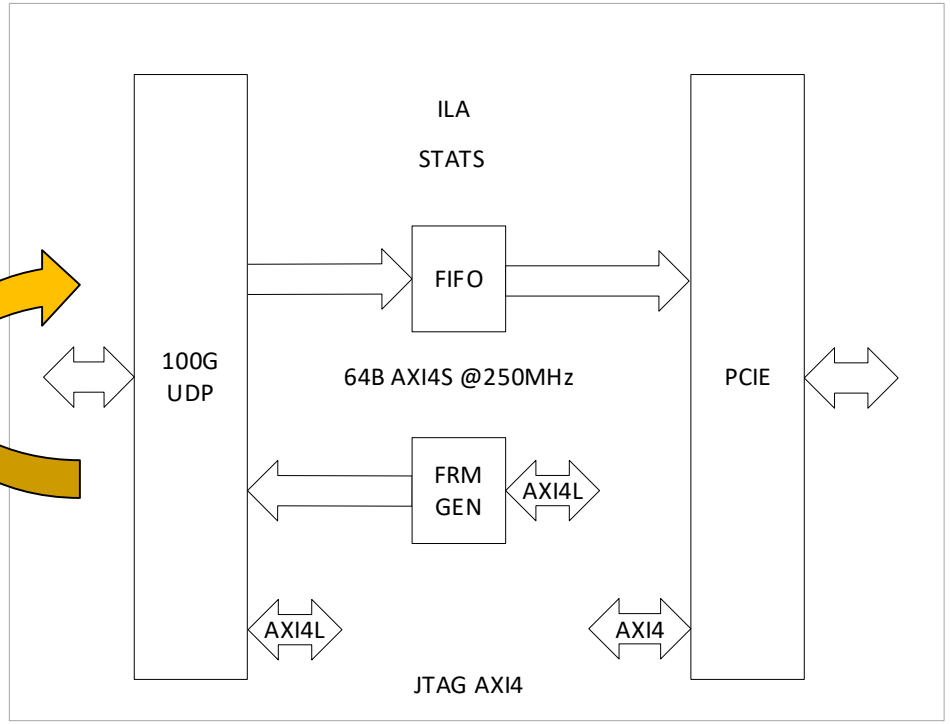
- Hosted on STFC Gitlab
- Automatic documentation
- Sphinx/Read the Docs/doxygen
- Mem Map Tables from XML
- Helper Scripts
- VUNIT Simulation
- BSD 3-Clause License
- packaged

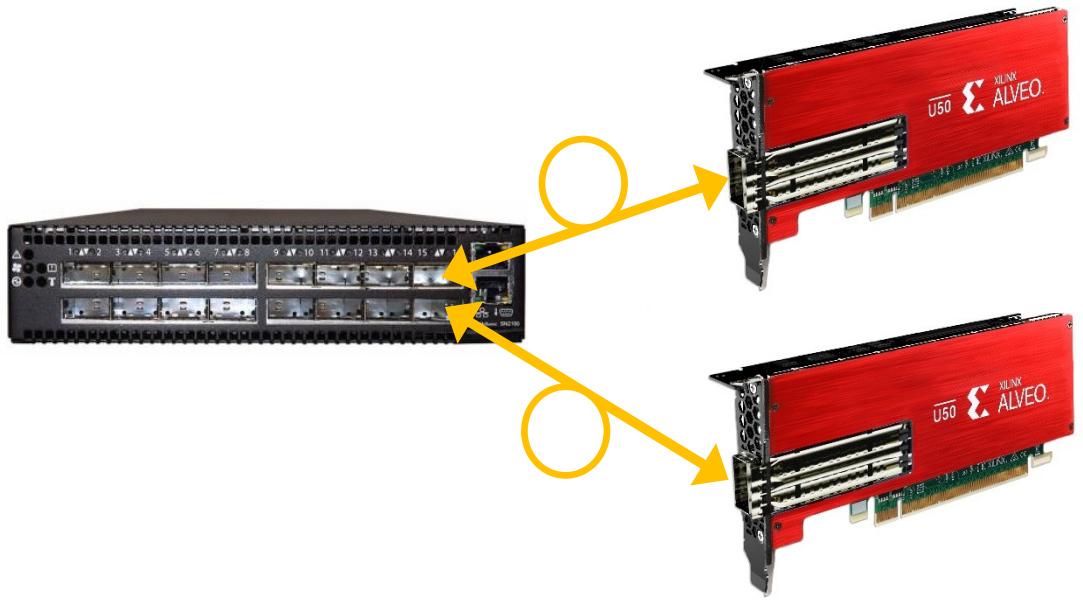
100G UDP Core – H/W Test



- Alveo U280 & HTG RFSoC
- FPGA <-> FPGA via switch
- Ran Embedded Test overnight, no errors or dropped packets
- Script reads switch counters over time period to give a measured speed – 99.8 Gb/s
- JTAG AXI4L core & xtcl setup
- ILA

100G UDP-PCIe Test Top Level





Full Processing FPGA Top Level