UDP, Ethernet & Implementation in FPGA

Andreas Kugel, ZITI
Content

• Intro
  – Readout + Networks
• Details
  – Protocols + Electronics
• Example
  – DSSC Readout
• Extras
• Summary
Experiment, Simplified ...

Timing Control Data

Some other Magic ...

CERN [1]

Results

Detour

Some Magic ...

Analysis / Algorithms

CERN [2]

CERN [3]
Readout (1)

- **Readout ASIC on sensor**
  - Electrical interface
    - Low power
    - Low material budget
    - Good reliability
    - Radiation hard
    - Optimized, custom protocol
      - „Local“ standards (e.g. GBT)
      - 1Mbit/s .. 1Gbit/s
      - Burst transmission
      - Needs timing + control

- **Near-detector**
  - Concentration, formatting, conversion (e.g. optical)
  - Provides timing + control
  - Add-on functionality
    - Configuration

---

**Custom Electronics**

GBTX [5]
Readout (2)

- Off-“detector“
  - Management
    - Buffer, concentrate, reformat, rearrange, reorder
  - Merge services (timing, control, DAQ)
  - Convert to commodity interface (e.g. PCI, Network)
  - Stand-alone/PC-based
    - PC: NIC + FPGA card
- Network: IP + Ethernet
  - De-facto standard
Network Topology

- Commodity “stuff”
  - Low-cost
  - Scalable
  - Good support HW + SW

- Excess latency
  - Many hops
  - Flow control

- Packet loss
  - Unreliable protocols (UDP)
  - Bit errors (unlikely)
  - Switch blocking, congestion

- Physics setups confined
  - Good chance UDP works
  - Point-to-point / < 70% load
# OSI Network Model

<table>
<thead>
<tr>
<th>Layer</th>
<th>Item (PDU)</th>
<th>Scope</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7: Application</td>
<td>Data</td>
<td>User</td>
<td>SSH</td>
</tr>
<tr>
<td>6: Presentation</td>
<td></td>
<td></td>
<td>Crypo</td>
</tr>
<tr>
<td>5: Session</td>
<td></td>
<td></td>
<td>NFS</td>
</tr>
<tr>
<td>4: Transport</td>
<td>Segment/Datagram</td>
<td>Endpoints</td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>Media</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3: Network</td>
<td>Packet</td>
<td>Nodes, 1 network</td>
<td>IPv4</td>
</tr>
<tr>
<td>2: Data Link</td>
<td>Frame</td>
<td>Node to node</td>
<td>Ethernet</td>
</tr>
<tr>
<td>1: Physical</td>
<td>Bit</td>
<td>Media</td>
<td>Ethernet</td>
</tr>
</tbody>
</table>

- Endpoints connected through OSI layer „stack“ [4]
  - Open Systems Interconnection model
- Only same layers communicate
- Reliability: layer 4 (optional! Not with UDP)
- User data: layer 5..7
Physical Layer

• Media
  - Electrical
    • Low power, low material budget
    • BW ~ up to **1Gbit/s** (twisted-pair), few meters
    • Isolation might be issue (caps, transformers)
  - Optical
    • Power and space requirements
    • BW ~ up to **25Gbit/s** (fiber), some 100 meters
    • Aging

• Ethernet (currently)
  - **1000BASE-T**: 1Gbit/s, RJ45, 4 twisted-pairs
  - **1000BASE-SX**: 1Gbit/s, SFP, dual-fiber
  - **10GBASE-SR**: 10Gbit/s, SFP+, dual-fiber
  - **100GBASE-SR4**: 100Gbit/s, QSFP+, 8 fibers
1000BASE-T

- **OSI Layer 1: PHY ASIC**
  - GMII: interface to layer 2
    - CLK, 8 bit data, 2 bit control
    - TTL level
    - RGMII option (fewer lines)
  - MDI/X: media dependent interface
    - Word clock recovery
    - Scrambling (spread spectrum)
    - Trellis encoding / viterbi decoding
    - PAM5 modulation over 4 lines
      - $5^4 = 625$ symbols
      - 8-bit data + parity + control
  - Echo cancellation
    - Pulse transformer interface

![PHY chip diagram](wikipedia.org/wiki/Trellis_modulation)

![PHY chip diagram](Belfuse[8])
Optical: xGBASE-S...

• Bit-serial data transmission
  – Ser/Des in PHY/FPGA
  – Symbol coding (balanced)
    • 8B/10B (1Gbit/s)
    • 64/66B (>=10Gbit/s)
  – Clock-data-recovery (CDR)
    • PLL + VCO
    • Regenerate bit clock
    • Synchronization option

• Optical transceivers
  – Dual fibers (one per direction)
  – Options (short/long range, single-mode, multi-mode)
  – 1, 4, 12 channels, hot pluggable
Data-Link-Layer

- By spec has two sub-layers
  - Logical-Link-Control (LLC)
    - Interface to network layer
    - Protocol mux (IP, ICMP, ...)
    - Flow control (pause frames)
    - Error detection (via FCS)
  - Media Access (MAC)
    - Interface to PHY (GMII)
    - Physical addressing
      - Not routable
    - Framing
      - PHY control signals
      - 14 byte header
      - FCS, gaps, idles

Frequently referred to as MAC-Layer
FPGA compatible

MTU
1500 byte regular
9000 byte jumbo

6 byte dest addr
6 byte src addr
Network Layer

• In this case: IPv4
  – Internet Protocol

• Many features
  – Host configuration (DHCP)
  – Address resolution (ARP)
    • Local and global (world-wide) addressing
  – Routing across networks
  – Various protocols like TCP, UDP

• Packet format
  – 20 byte header (plus options, if any)
  – Almost 64kB payload, 8 byte units
  – Fragmentable

• Connectionless node-to-node datagram delivery

At this layer, as in the lower ones, errors are only detected. Transfers are not reliable.
UDP (User Datagram Protocol)

- Application level connectionless datagram
- 8 byte header
  - Port numbers (src, dest)
  - Datagram byte length (just below 64kB)
  - Checksum (incl “pseudo-header” with IP addresses)
- Still not reliable
- No handshaking overhead
- Application API via sockets, read(), write()
  - IP and Ethernet handled by OS
TCP (Transport Control Protocol)

- Application level reliable transmission
  - ACK messages
  - Timeouts
  - Retransmission
  - Buffering
- 20 byte header
  - Port number (src, dest)
  - Seq. / Ack. numbers
  - Flags, Window, Checksum
- MTU size fragments
- Connection oriented API
  - Sockets with init phase
  - Connect(), listen(), accept()
FPGA

• Programmable HW: implements any digital logic
• Basic element: logic cell. Millions of (well, up to)
• Used in virtually any experiment ...
FPGA Special Features

- Special functions: Hard IP
  - Clock management
  - I/O SerDes (fabric/hard)
  - Memory (BRAM, FIFO, memory controller)
  - DSP blocks
  - Protocol/processing (PCIe, Ethernet, …)
  - Processors
- Plus Soft IP libraries
  - All kind of stuff …
    - Processing, Networking,…

Largest FPGA ~ 20B transistors. > GPU/CPU
Ethernet

- Ethernet MAC
  - Soft-IP, any modern FPGA (resource consuming). 1G, 10G, (100G)
  - Hard-IP, selected FPGAs (1..few / FPGA)
- Ethernet PHY (serial only/ no CAT5)
MGT (GTP/H/X)

- Multi-Gigabit-Transceivers
- Serial bitrates up to 33Gbit/s
- Many protocols
  - FC, GE, SATA, PCIe
  - Custom (e.g. for low latency)
- Channel bonding
- Clock correction
- Clock recovery
High-Performance SoC

- Soft/old-style CPUs are FPGA slaves
- Newer hard CPUs are stand-alone on-chip blocks: Cyclone V, Stratix 10, Zynq/Zynq+
- Performance approaching high-end mobile CPUs – 4-core, 64 bit, 1.5GHz, GPU
- High-bandwidth fabric interface for efficient co-processing
- Integrated debugging

FPGA-centric or CPU-centric
**Boards**

- Apart from power, all additional components are application specific

- Flexible, multi-purpose systems with standard interfaces/peripherals (network, memory, etc)
  - FPGA adapts to application
    - E.g. µTCA FPGA
  - Reconfigurable System-on-Chip
    - E.g. FPGA SoM
  - Open hardware (e.g. netfpga.org)
More Boards

Custom

Commercial

MPRACE2 (Heidelberg)

DSSC PPT (Heidelberg)

DE0 (Altera/Terasic)

Atlas IBL BOC (Heidelberg)

Atlas RobinNP (CERN)

Zybo (Xilinx/Digilent)
Detector Readout

- Interface to optimized detector electronics
  - Custom signal levels, protocol, timing
- On-detector (close to) data processing
  - Compression, histograms, filtering, FEC
- Challenging data transfer requirements
  - Latency, bandwidth, channels, buffering
- Complex functionality
  - Custom SoC

*Any of above + small volume: no escape from FPGA*
Tool Flow (1): Specification

- **Basic**
  - HDL (Verilog, VHDL)
  - IP Libraries

- **Graphical tools**
  - Matlab
  - FSM GUIs
  - Vendor GUIs (block diagrams)

- **High-level entry**
  - OpenCL (=> application kernels)
  - Vivado HLS (“C” => IP blocks)

```process
Begin
Wait until rising_edge(clk);
C <= A + B;
End process;
```
Tool Flow (2): Simulate

• Simulator „runs“ the HDL description
  – Waveform output
  – Breakpoints
  – Checks
    • Text-I/O
    • Assertions
    • API (FLI) + testprog.
  – Incl. processor

Fast turnaround
Slow execution
Tool Flow (3): Synthesis + Mapping

- Synthesize HDL into **RTL** netlist (gate level)
  - Vendor/third-party tools
- Map RTL onto technology
  - LUT content + connections
- Place LUTs + routing
- Generate configuration “bitstream”
  - Equivalent to SW binary
  - Loading bitstream to FPGA starts “execution”

*Can take **hours**!*
Tool Flow (4): Configuration

- **Boot options**
  - Download bitstream via JTAG cable
  - Download bitstream from on/off-chip CPU
  - Auto-boot from FLASH/SD-card

- **Runtime reconfiguration**
  - Load/modify (section of) device at runtime
  - Internal partial reconfiguration (caveats)
Tool Flow (5): Test

- ChipScope “debugger”
  - Access FPGA internal state + memory content
  - Trigger + Control
  - Waveform display
  - Incl. processor

Slow turnaround
Fast execution
High-Level-Synthesis

- Tool extracts DFG + CFG from “C”
- Loop pipelining
- Operator optimisation
- C-code entry (approx):

\[
\text{Do } \{ \\
\text{zx}' = zx\cdot zx - zy\cdot zy + cx; \\
\text{zy}' = 2.0\cdot zx\cdot zy + cy; \\
\text{While (zx\cdot zx + zy\cdot zy < 4.0)}
\}\]

- C-based simulation
- Can also do pixel S-curve fit ...
Case Study: XFEL

- DESY-HH, Germany (2016)
- e⁻ linear accelerator (1.7km) undulator
  - X-ray "flashes" (<100fs)
    - 600µs bunch, 99.4ms break
  - High intensity
    - $10^{12}$ photons/bunch
  - 250eV..25keV/photon
- Bio-molecules, nano-structures, chem. processes,...
  - Single or multi-shot

XFEL.EU

Lutz, NIMA 2010
Numbers updated
DSSC Readout

- 1M pixel detector
  - 4 quadrants
    - 4 ladders @64k pix
- 5MHz readout * 800 frames * 10Hz
  - 75Gbit/s raw BW
- Tasks
  - Configuration
  - Readout control/timing
  - Data transfer
    - ASIC => UDP
    - 16*10GE

DSSC: [9], [10]
System View

C&C Ctrl/Time Generator
Control Network 1G
DAQ Network 10G

FPGA2
X4

FPGA1
X4
X16

ASIC
SENSOR

Quadrant
Ladder

Lab ladder setup

FPGA1: Spartan-6
FPGA2: Kintex-7
Software

- QT GUIs on host for data taking / lab tests
- Socket communication
- Embedded Linux
  - RPC-like task set ("device")
Configuration Path

- Ethernet 1G <=> **FPGA2**
  - Firmware download (dynamic + static)
  - Calibration support
  - Run control
  - Monitoring

- Embedded System
  - Microblaze CPU, 100MHz
  - Linux + simple I/O to fabric

- Simple HDL
Control/Timing Path

- CLK + CTL telegrams
- CTL decoder, master-FSM, CLK mult, veto memory
- Slave-FSM, PWR/CLR driver, CLK-Buf

- Plain digital logic, HDL + FSM GUI tools
Data Transmission

- Digital logic
  - FSMs
  - FIFOs
- Embedded software
  - Register/memory interface to fabric
  - Track RO status
  - UDP header initialization
  - Reordering
PPT (FPGA-2) Design
**Data Path**

**Kintex 7 FPGA**

- **Data Receiver**
  - Rx Aurora
  - 3 Lanes x4
  - 9/10 Bit Input: 20.16 GBit/s

- **16 Fifos**
  - 4 Fifos
  - 4 Fifos
  - 4 Fifos
  - 4 Fifos

- **256 to 128 Bit FIFO**
  - x4

- **DDR3 800MHz buffer**
  - cntrl
  - Max 102.4 GBit/s

- ** Ethernet Channels**
  - x4
  - UDP (+TTP)
  - Fifos
  - IP
  - MAC
  - PHY

- **Microblaze**
  - Linux - Control
  - @100MHz

**Calculations**

- **16Bit Data Words:**
  - \(35.84 \text{ GBit/s}\)

- **Incl Header Bytes:**
  - \(9.02 \times 4 = 36.08 \text{ GBit/s}\)
  - @ 8KB UDP Packetsize

**9/10 Bit Input:**
- 20.16 GBit/s

**16Bit Data Words:**
- 35.84 GBit/s

**Incl Header Bytes:**
- 9.02x4 = 36.08 GBit/s @ 8KB UDP Packetsize
# UDP Packet Format

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>14 Bytes</td>
</tr>
<tr>
<td>IP</td>
<td>24 Bytes</td>
</tr>
<tr>
<td><strong>Receiver MAC 48Bit</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Sender MAC 48 Bit</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Ethernet Type (0x8870)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>IP Packet Length (1062/1078)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Identification</strong></td>
<td>Packet ID 16Bit</td>
</tr>
<tr>
<td><strong>ttl (0xFF)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Proto (0x11=UDP)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>IP Header CRC</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Address 32Bit</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Source IP</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Destination IP</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Options (0x00)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Padding (0x00)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Source Port</strong></td>
<td>16Bit</td>
</tr>
<tr>
<td><strong>Length (8208)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>DATA</strong></td>
<td>8192 Bytes</td>
</tr>
<tr>
<td><strong>CRC (0x0000) not used</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word a 0 16Bit</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word b 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word c 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word d 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word e 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word f 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word g 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word h 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word i 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word j 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word k 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word l 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word m 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word n 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word o 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word p 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word q 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word r 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word s 0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word a 1</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Word b 1</strong></td>
<td></td>
</tr>
<tr>
<td><strong>... total 8192 Bytes Payload</strong></td>
<td></td>
</tr>
<tr>
<td><strong>TTP – Trailer</strong></td>
<td>8 Bytes</td>
</tr>
<tr>
<td><strong>SOH, EOF, Reserved</strong></td>
<td></td>
</tr>
<tr>
<td><strong>MSB</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Train ID (32 Bit)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>LSB</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Packet ID</strong></td>
<td></td>
</tr>
<tr>
<td><strong>LSB</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Note:**
- DMA 8 kB datagrams
- jumbo frames
- CPU
- 8192 bytes payload
- 8 Bytes

**References:**
- HighRR, FPGA Networking, A. Kugel
Results

- System works well at full speed
- FPGA1: straightforward design (HDL)
- FPGA2: complex HDL
  - Bit shuffling
  - Much FIFO buffering
  - Multiple FSMs
  - CPU + fabric interface slow
  - Almost full

- If to do it again …
  - Faster links between FPGA + FEC
  - Hard CPU + faster fabric I/F
    - UDP DMA driver?
    - Buffer access
  - HLS blocks
    - Converters
    - UDP DMA
Recipe

• Analyze requirements
  – Bandwidth, distance, number of channels
  – Extra features (latency, timing, control, ...)
  – Constraints (radiation, power, ...)

• Select Architecture
  – Overall topology
  – Unit design
    • PC-based, crate-based, embedded system (SoC)
    • Bandwidth, channel aggregation
    • Extra features + constraints

• Implementation Options
  – Devices (FPGA family, hard/soft CPU, ...)
  – Media (electrical, optical, ...)
  – Tools and IP

• Choose components

Optimize

Look what others are doing. And why!
Extra: TCP in FPGA

• Reliability
  – Significant more effort to implement wrt. UDP

• FPGA SoC
  – CPU TCP sufficiently performant?
    • Might work for 1G if hard-CPU (ARM)
  – 10G++ need HW TCP
  – Couple of IP around
    • Opencores: http://opencores.org/project,tcp_socket
      – Performance unclear
    • QuickTCP: www.plda.com/products/fpga-ip/xilinx/fpga-ip-tcpip/quicktcp-xilinx
      – Commercial

• “C”-based FPGA design (Vivado-HLS):
TCP/IP over Ethernet

*Work in progress ...*

**Reliable Internet**
- TCP Byte Stream
- "Any" size

**Unreliable Internet**
- IP Packet = TCP Segment
- 64kB max

**Client**
- Send Buffer
- TX-APP
- free
- valid
- sent

**Server**
- Rcv Buffer
- RX-APP
- rcv'ed
- WINDOW

**Unreliable Local network**
- Ethernet Packet = IP Fragment
- 1.5kB max MTU
**TX Straw Man Architecture**

- **Buffer**
- **Addr Gen**
- **TCP FSM**
- **IP/ETH HDR**
- **ETH Decoder**
- **ETH MAC**
- **ETH PHY**

**Data**

**Addr Gen**

**Buffer Mgmt**

**TX FSM**

**TCP FSM**

**IP Segment Size = 1 MTU**

**.init/term Ping, ARP, Slow data**

**Software**

**SW = SW or HW or shared HW**

---

... includes "SW" for network + connection functions

---

**2016-04-11**

HighRR, FPGA Networking, A. Kugel
TCP: Standard

- Linux kernel: many files, all tcp ~ 15k lines of code, all IPV4 57k lines
- LwIP: several files, tcp.c ~ 1k lines, 17k all

Just run HLS. No Way!
TCP: Custom SW Model

4 „loops“, 1 shared data structure (SCB)

<table>
<thead>
<tr>
<th>Language</th>
<th>files</th>
<th>blank</th>
<th>comment</th>
<th>code</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>7</td>
<td>254</td>
<td>428</td>
<td>1333</td>
</tr>
<tr>
<td>C/C++ Header</td>
<td>2</td>
<td>74</td>
<td>87</td>
<td>233</td>
</tr>
</tbody>
</table>

SUM: 9 328 515 1566
Model Features

Linux, Client + Server (emulator) Programs, Pipes

TX, RX, ACK segments

Timeouts

ERTT update

Buffer + management (SCB) occupancy

Window update

Model Features:
- TX, RX, ACK segments
- Timeouts
- ERTT update
- Window update

Details:
- TX, RX, ACK segments graph
- Timeouts graph
- ERTT update graph
- Window update graph

Linux, Client + Server (emulator) Programs, Pipes

2016-04-11
HighRR, FPGA Networking, A. Kugel
Targeting PPT

Product family: kintex7
Target device: xc7k325tffg900-2

Performance Estimates

- Timing
  - Summary
    | Clock | Target | Estimated | Uncertainty |
    |-------|--------|-----------|-------------|
    | default | 3.00 | 3.81 | 0.38 |

- Latency (clock)

Utilization Estimates

- Summary
<table>
<thead>
<tr>
<th>Name</th>
<th>BRAM_18K</th>
<th>DSP48E</th>
<th>FF</th>
<th>LUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression</td>
<td></td>
<td></td>
<td>0</td>
<td>163</td>
</tr>
<tr>
<td>FIFO</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instance</td>
<td>32</td>
<td>8</td>
<td>8157</td>
<td>12564</td>
</tr>
<tr>
<td>Memory</td>
<td>2</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Multiplexer</td>
<td></td>
<td></td>
<td></td>
<td>1149</td>
</tr>
<tr>
<td>Register</td>
<td></td>
<td></td>
<td>1157</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>34</td>
<td>8</td>
<td>9314</td>
<td>13876</td>
</tr>
<tr>
<td>Available</td>
<td>890</td>
<td>840</td>
<td>407600</td>
<td>203800</td>
</tr>
<tr>
<td>Utilization (%)</td>
<td>3</td>
<td>~0</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

General Information

- Report date: Fri May 15 09:09:41 EDT 2015
- Device target: xc7k325tffg900-2
- Implementation tool: Xilinx Vivado v2014.4

Resource Usage

<table>
<thead>
<tr>
<th></th>
<th>VHDL</th>
<th>SLICE</th>
<th>LUT</th>
<th>FF</th>
<th>DSP</th>
<th>BRAM</th>
<th>SRL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2004</td>
<td>5690</td>
<td>5590</td>
<td>3</td>
<td>2</td>
<td>72</td>
<td></td>
</tr>
</tbody>
</table>

Final Timing

<table>
<thead>
<tr>
<th></th>
<th>VHDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP required</td>
<td>3.000</td>
</tr>
<tr>
<td>CP achieved</td>
<td>3.567</td>
</tr>
</tbody>
</table>

Timing not met

Export the report(.html) using the Export Wizard

- 312.5 MHz speed for 10G @ 32 bit: almost met
- Low 6% utilisation
Extra: Timing and Control

*White Rabbit Project [12]*

- Ethernet based
- Synchronization
- Deterministic latency
- Data and Control
White Rabbit Details [13][14]

- **Precision Time Protocol (PTP)**
- **PHY Clock Recovery**
- **Dual-Mixer Time Difference**

- **PTP ~ 1us**
- **DMTD < 1ns**
Ethernet Prospects

- **State-of-the-art**
  - 1G electrical or optical (single fiber)
  - 10G optical (electrical in progress), single fiber
  - 40G optical = 4*10G with channel bonding
  - 100G optical = 4*25G with channel bonding

- **FPGA**
  - Transceivers (PHY)
    - 10G pretty standard (Xilinx series-7, speedgrade -2)
    - 25G in newer devices (up to 32Gbit/s)
  - MAC-IP
    - 10G core well established (University programme)
    - 40G/100G soft IP for series-7
    - 100G hard IP for newer devices
Summary

- IP/Ethernet ubiquitous
- FPGA ubiquitous
- Can use FPGA for IP networking close to detector
  - UDP fairly easy
  - SoC-based for “low”-bandwidth requirements?
- PC-based networking might be more sensible
  - Depends on overall system architecture
- Advanced implementations possible
  - TCP
  - Timing synchronization
  - Deterministic latency control messages
- Keep an eye on Eth/IP for your next experiment!
refs

[4] OSI model: http://dx.doi.org/10.1109/TCOM.1980.1094702
[15]
Backup
Clocking

- DCM
  - Phase shift
  - Clock alignment
  - Frequency synthesis
  - Multi-phase clocks
- PLL
  - Frequency synthesis
  - Jitter filter
- Variety of clock buffers
- Limited resources
I/O Cells

- **IN/OUT buffer**
  - Tri-state option
  - Differential option
  - Termination
  - Levels: I/O "bank" supply

- **Programmable delay**
  - ~ 100ps resolution
  - Few ns max delay

- **SerDes**
  - 1:2, 1:4 ratio
  - Multiple clocks
Memories

- LUT memory
- Hard-IP memory blocks (18kB)
  - FIFO
  - Dual ported RAM
  - Configurable (A, D)
- Memory Controllers
  - Soft-IP
    - DDR1,2,3 + IO-Delay
  - Hard-IP
    - Multi-port, high speed
Processors

- **Soft-IP**
  - “Full custom“ µC
  - 32 bit RISC (µBlaze)
    - Linux
    - 100 .. 200MHz

- **Hard-IP (depends)**
  - Arm A9, dual core
    - 800 Mhz
    - FPU
    - Peripherals
Latest Processors

- Altera Stratix-10
- Quad ARM Cortex-A53

- Xilinx Zynq+
- Quad ARM Cortex-A53
- Dual ARM Cortex-R5
- Mail GPU

Latest Processors

- Altera Stratix-10
- Quad ARM Cortex-A53

- Xilinx Zynq+
- Quad ARM Cortex-A53
- Dual ARM Cortex-R5
- Mail GPU
DSP

- 18x18 (24x24) ADD – MUL – ADD
- Cascadable
- Filters, Processing, barrel shifters, ...

Xilinx: TOPS/s
DSP Soft-IPs

- **CORDIC**
  - Vector operations
  - Sin and Cos
  - Sinh and Cosh
  - Atan and Atanh
  - Square root
- **DFT, DFT^{-1}**
- **FIR**
- **DDS**
- ... via CoreGen