Measured Latency Introduced by RFNoC Architecture

GRCON17 - September 11-15, 2017

Josh Sunderlin
joshua.sunderlin@jhuapl.edu

- *This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA).*
- *The views, opinions, and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.*

APL
JOHNS HOPKINS
APPLIED PHYSICS LABORATORY

Background Summary

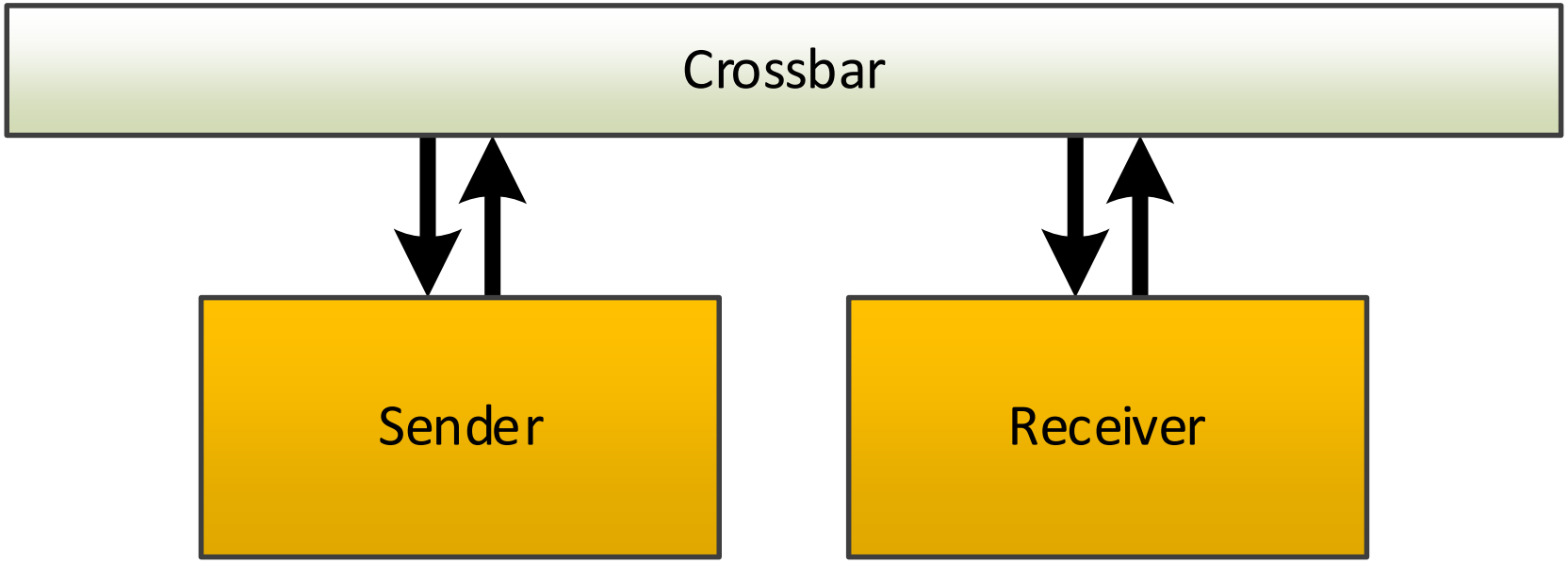
- Radio Frequency Network-on-chip (RFNoC) is an open source framework to develop software-defined radio (SDR) applications that can run on an FPGA-embedded universal software radio peripheral (USRP) transceivers
- To speed up and streamline the development process of signal processing algorithms, RFNoC was developed to modularize and seamlessly integrate the majority of non-processing related tasks into a common paradigm
- While this abstraction is useful for certain applications (i.e. academic, demonstrations, etc.), the automation of these background tasks could have deleterious effects on certain SDR applications that involve more intense computational or timing requirements
- To determine and benchmark the throughput and timing characteristics caused by the RFNoC framework, we sought to test the latency of test signals propagating through an implementation of working DSP blocks in an RFNoC-enabled USRP

Objective and Concept

- **Objective: Measure latency between RFNoC blocks**

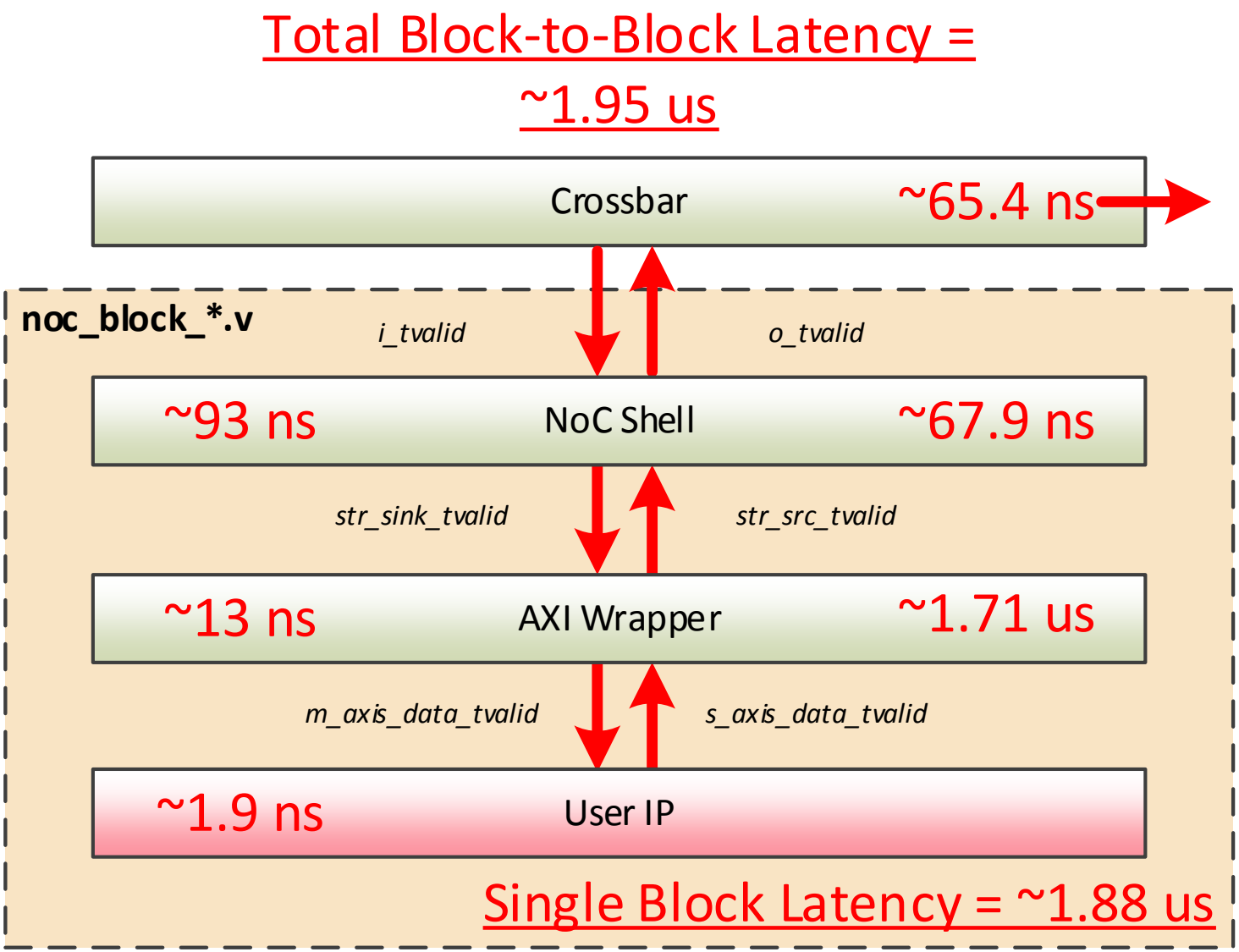
- **Concept: Transmit “tvalid” signals from RFNoC blocks to the USRP’s front panel GPIO**

RFNoC Design

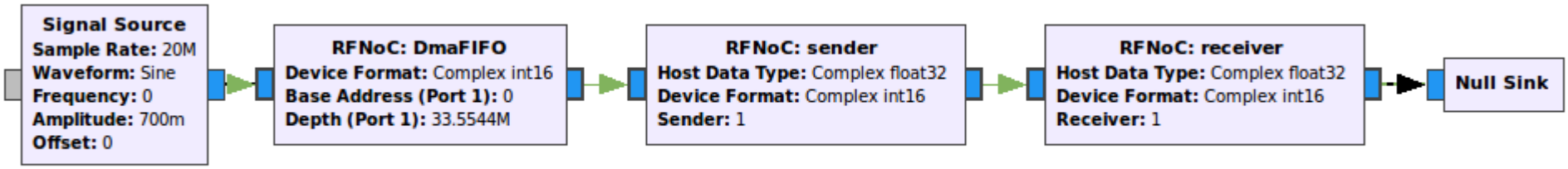


- **Created two Computation Engines (CE), Sender and Receiver**
- **Two types of latency:**
  - **CE block to CE block through the Crossbar**
  - **NoC Shell and AXI Wrapper overhead within the CE block**
  - **32-bit data payload**

Latency Measurements

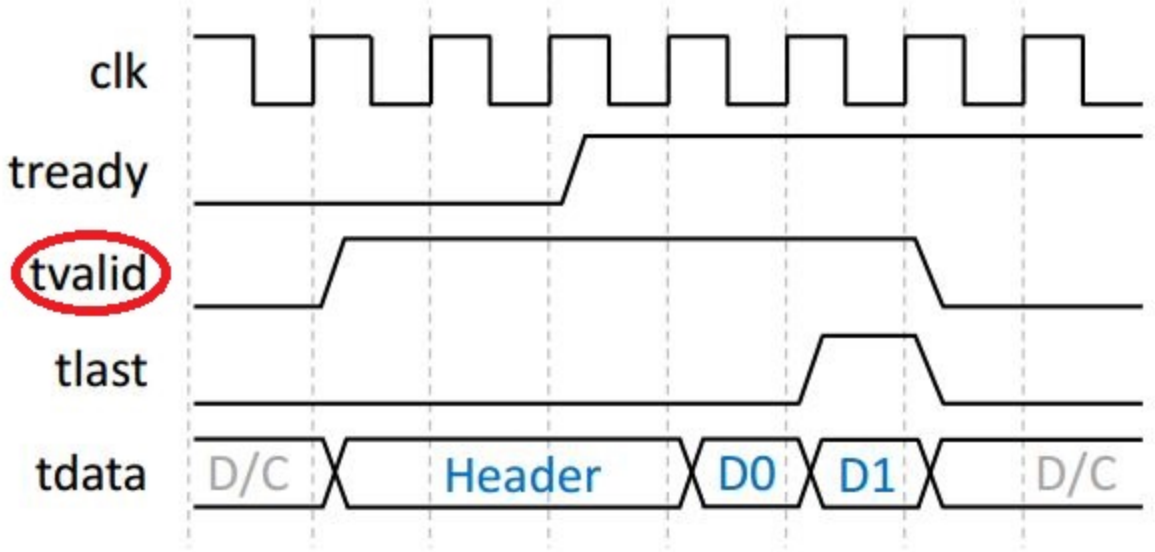


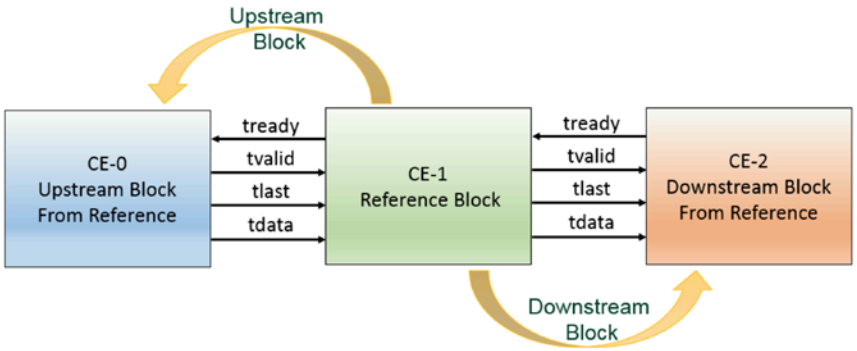
GRC Design and Test Setup



- **Test platform consists of:**
  - **Host Computer -> Ethernet -> NI USRP-2953R -> Oscilloscope**
  - **Signal source from host computer, driving a waveform**
  - **The DmaFIFO block helps to prevent underruns in GRC**
  - **The Null Sink block completes the chain back to the host computer**

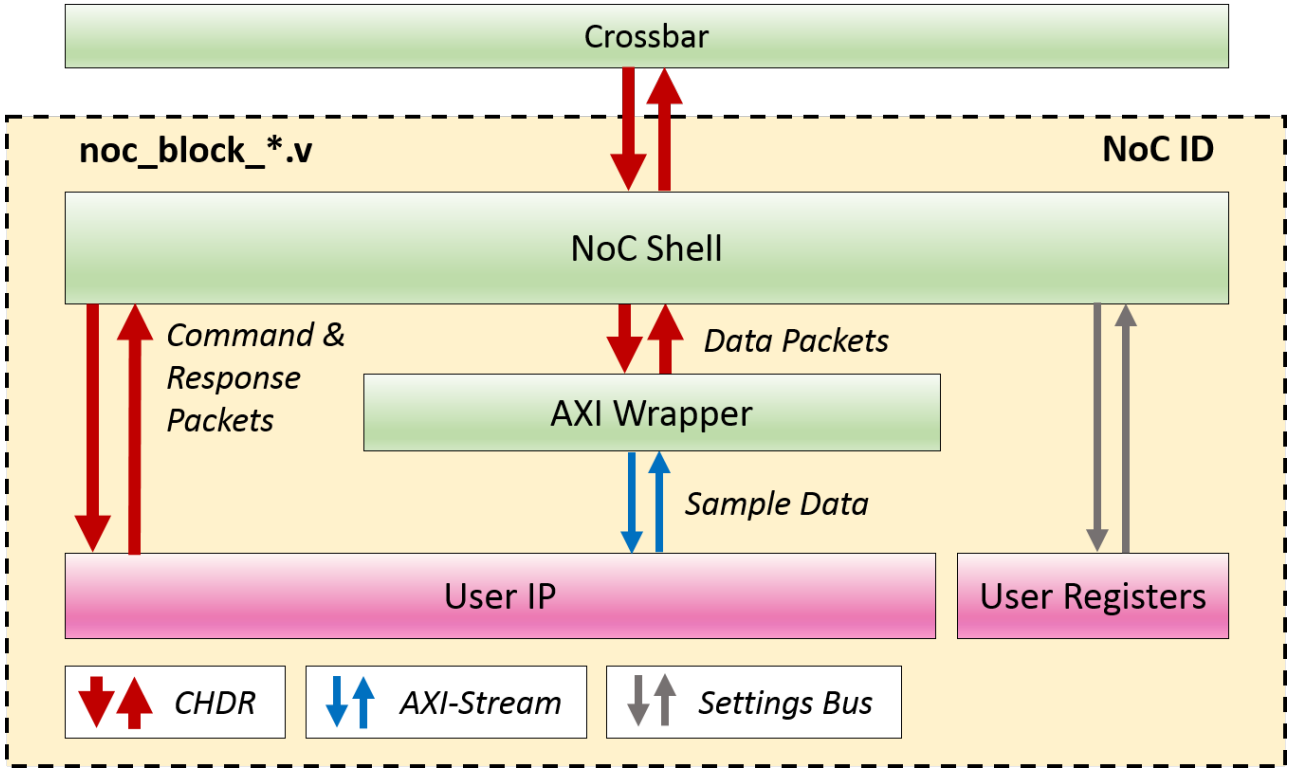
RFNoC Data Stream Structure





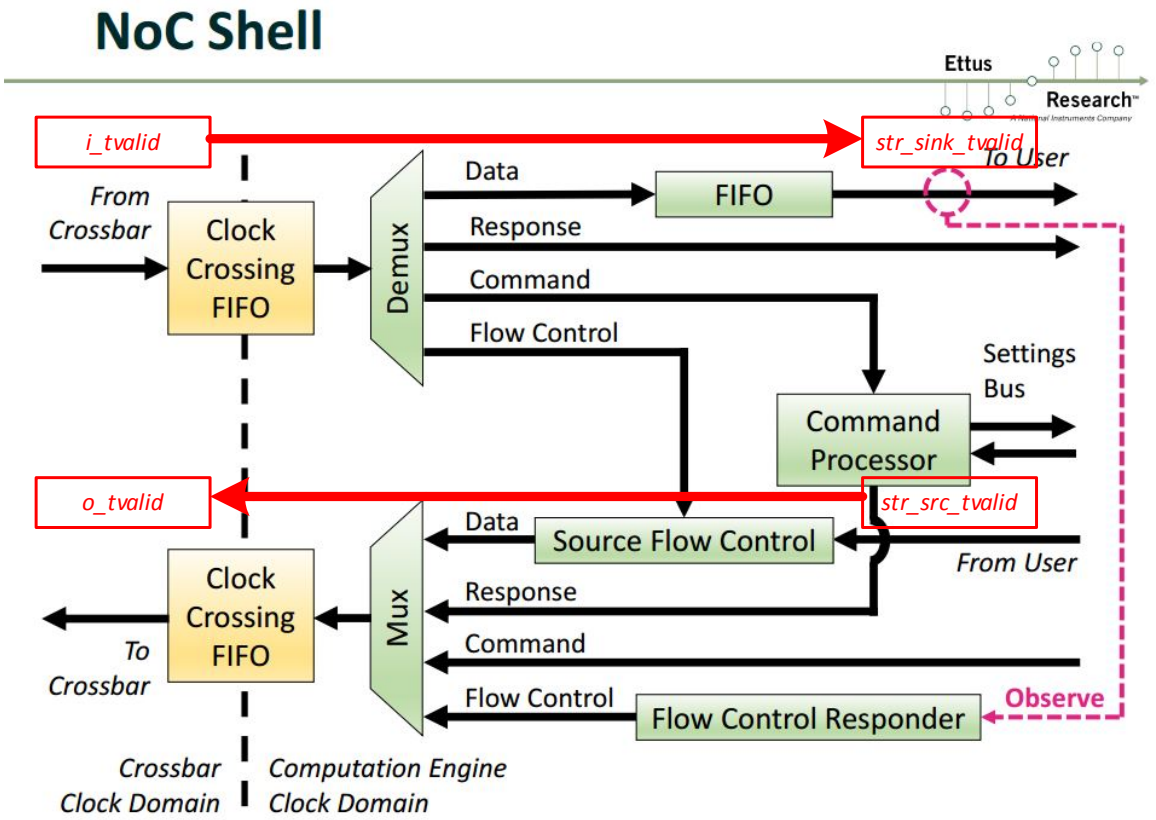
- **The rising edge of the tvalid AXI stream signal shows beginning of valid data coming into the CE block from the Crossbar**

RFNoC Block Internal components



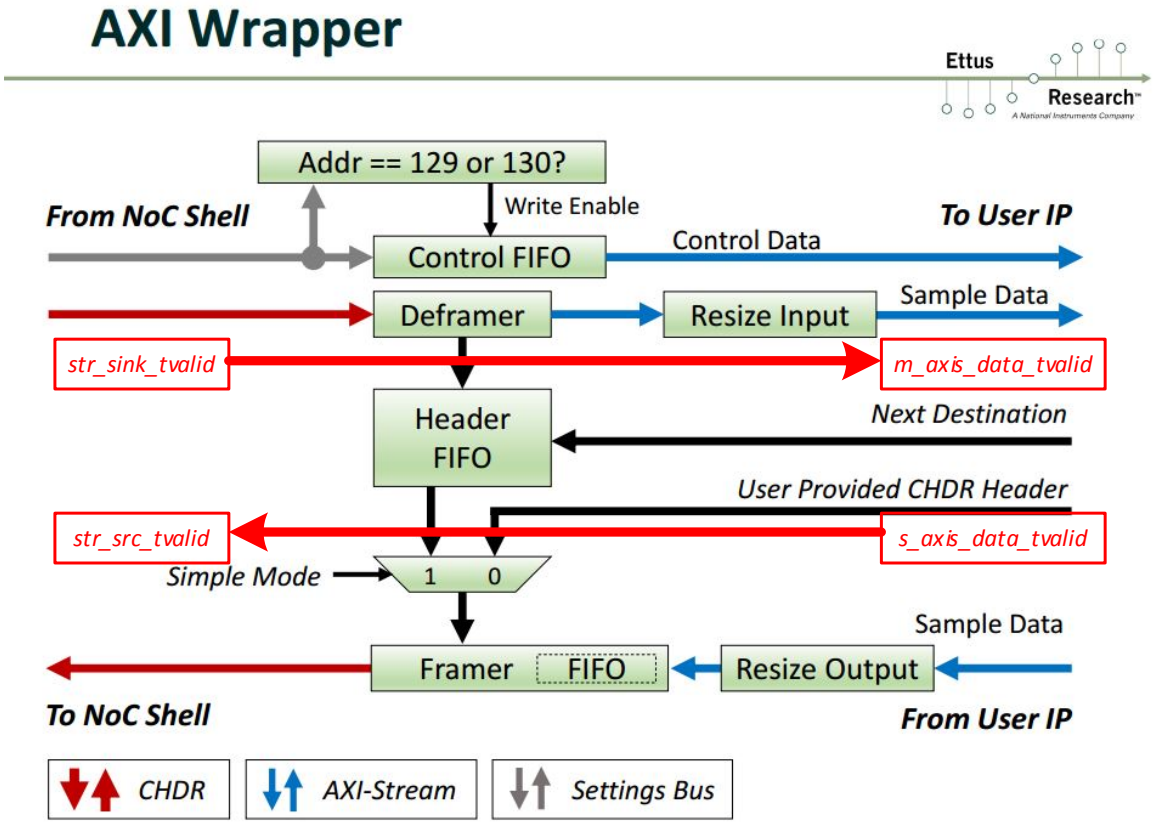
- **Signals of interest off the Crossbar and into the NoC block are "i_tvalid" and "o_tvalid"**
- **NoC Shell and AXI Wrapper are provided by RFNoC**
- **Our User IP is a "wired" data passthrough (virtually no latency)**

RFNoC NoC Shell Components



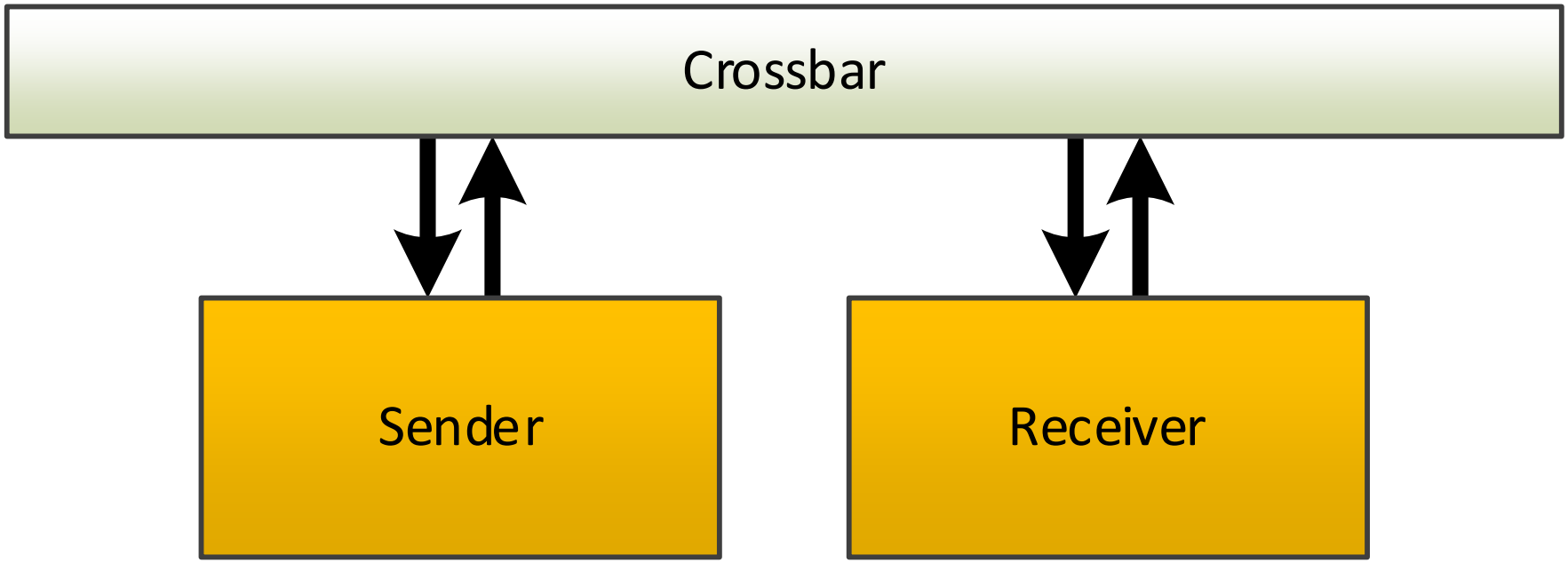
- **Conversion at NoC Shell:**
  - **“i_tvalid” becomes “str_sink_tvalid”**
  - **“str_src_tvalid” becomes “o_tvalid”**

RFNoC AXI Wrapper Components

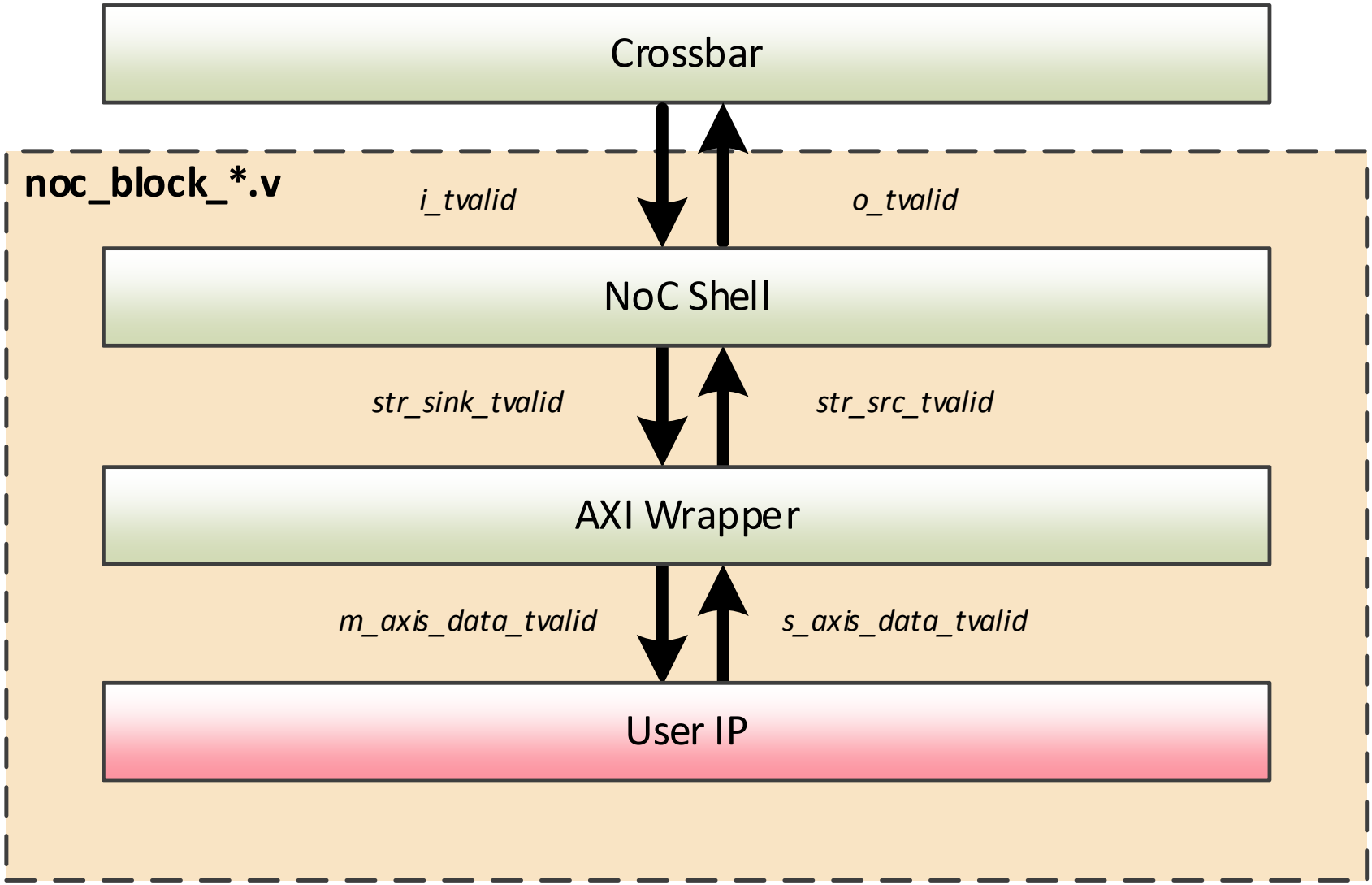


- **Conversion at AXI Wrapper:**
  - **“str_sink_tvalid” becomes “m_axis_data_tvalid”**
  - **“s_axis_data_tvalid” becomes “str_src_tvalid”**

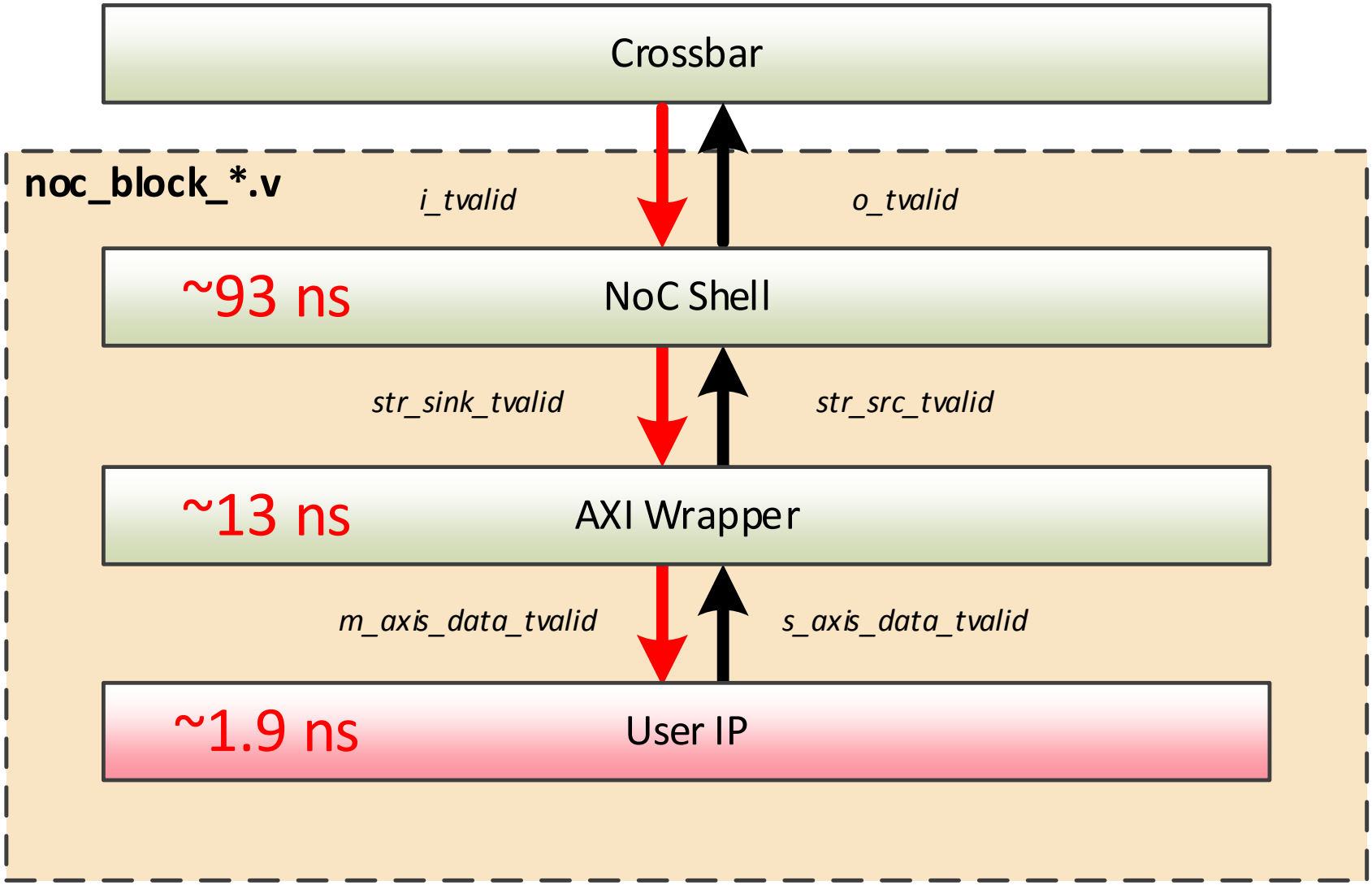
Block-to-block Diagram



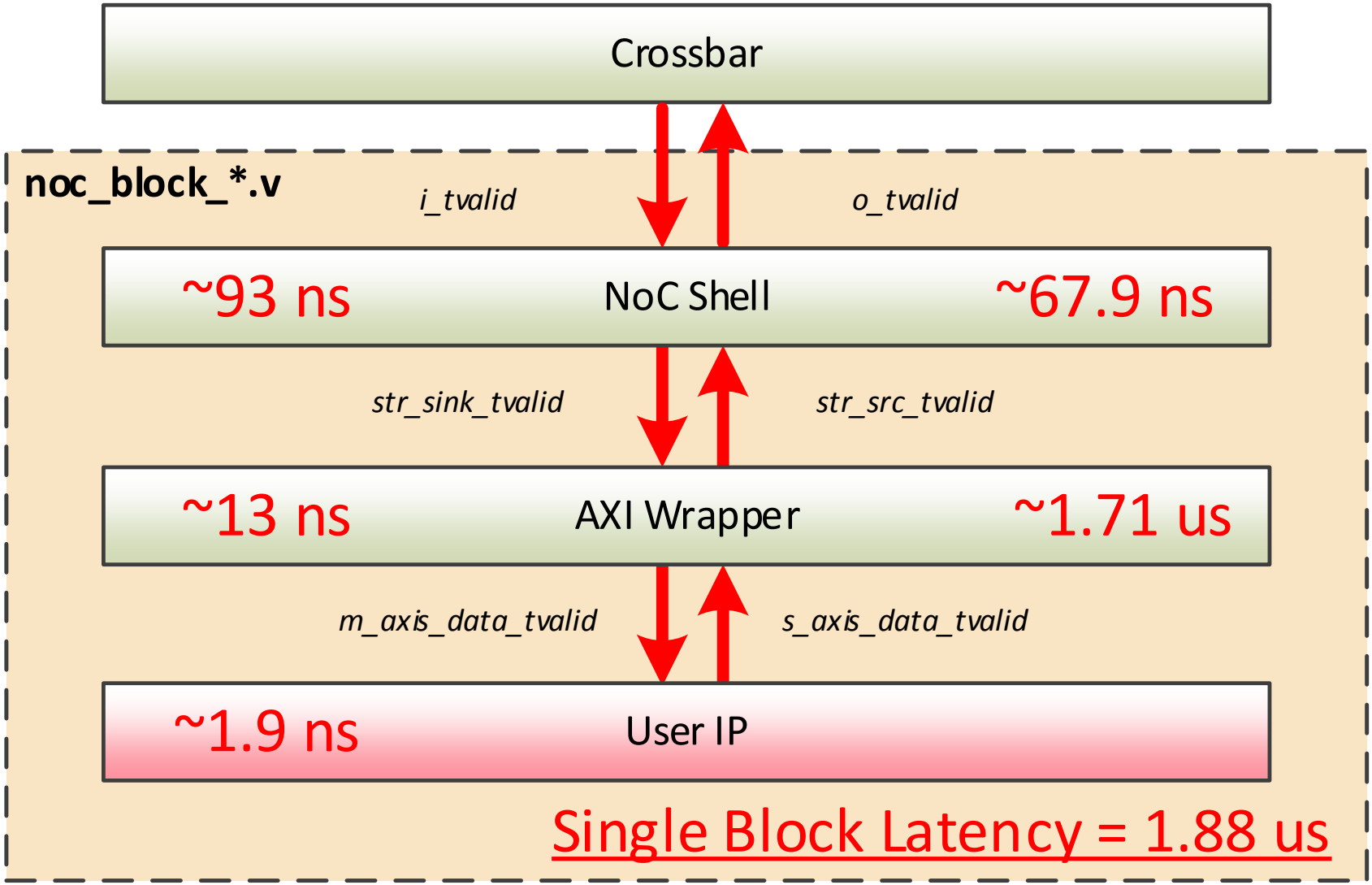
Internal Block Latency



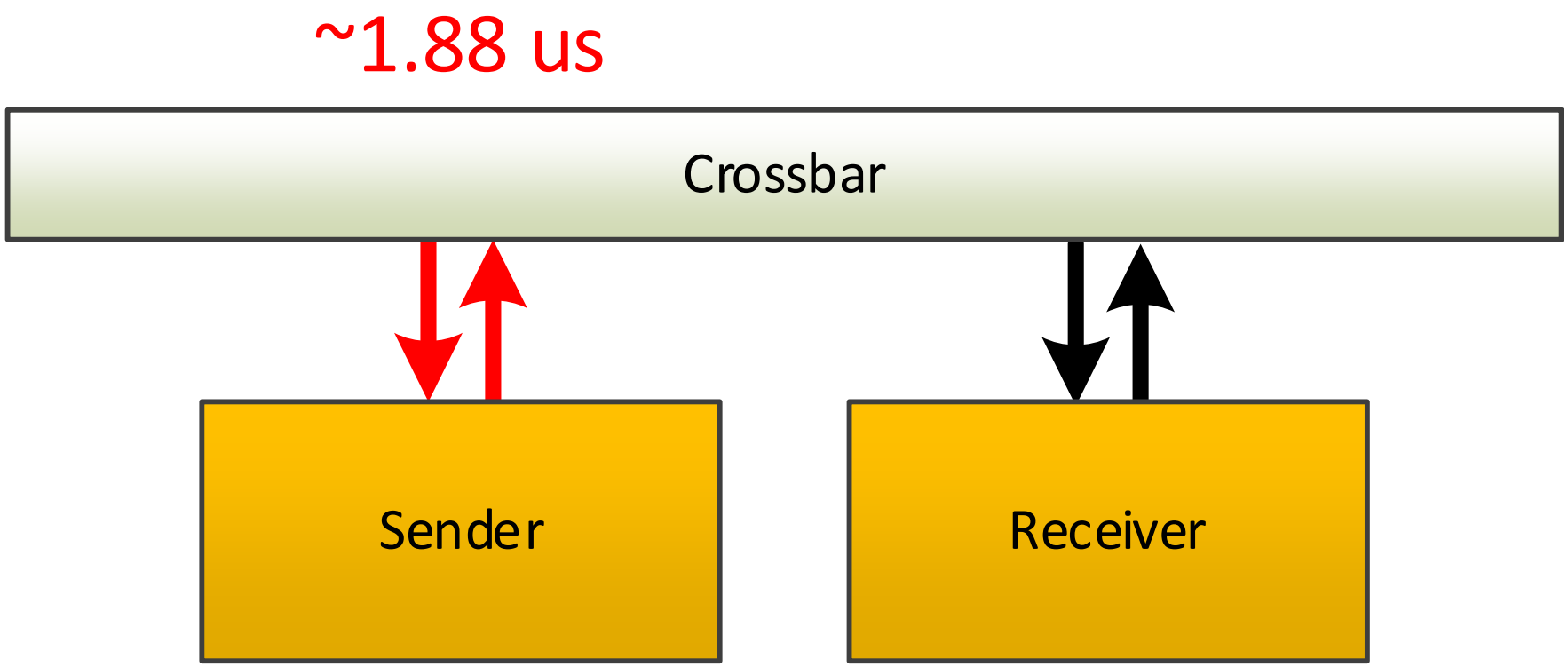
Internal Block Latency: Incoming Data



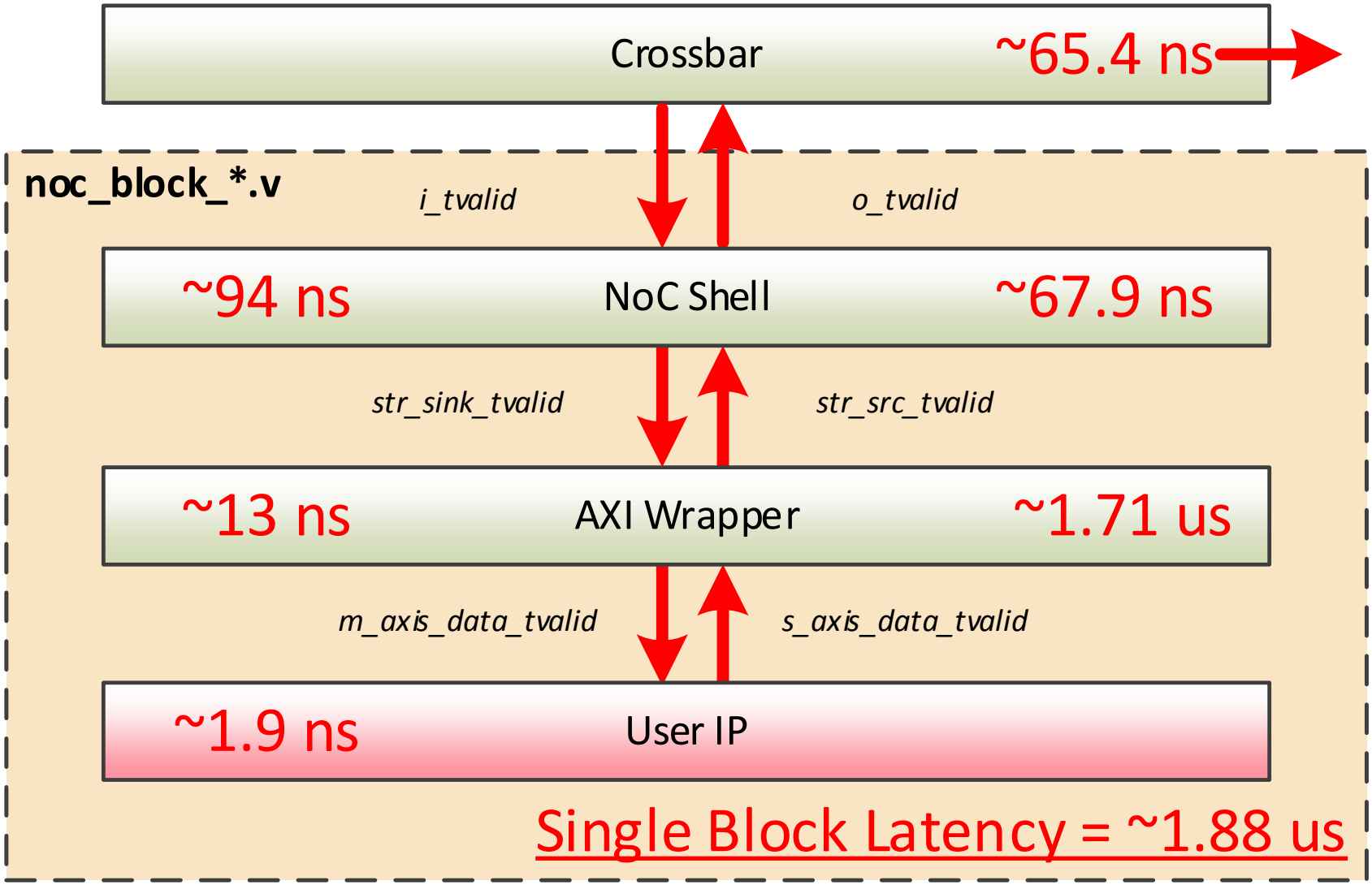
Internal Block Latency: Outgoing Data



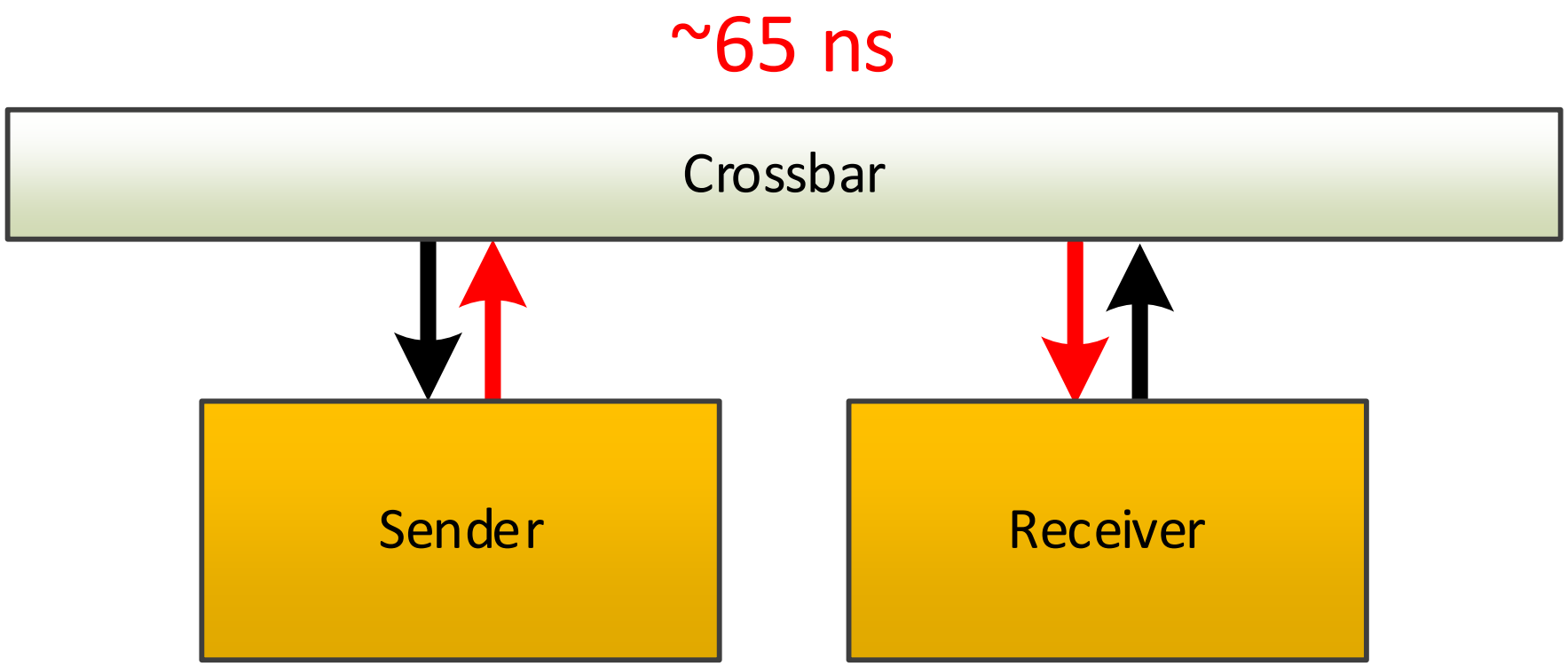
Single Block Latency



Internal Block Latency: Outgoing Data

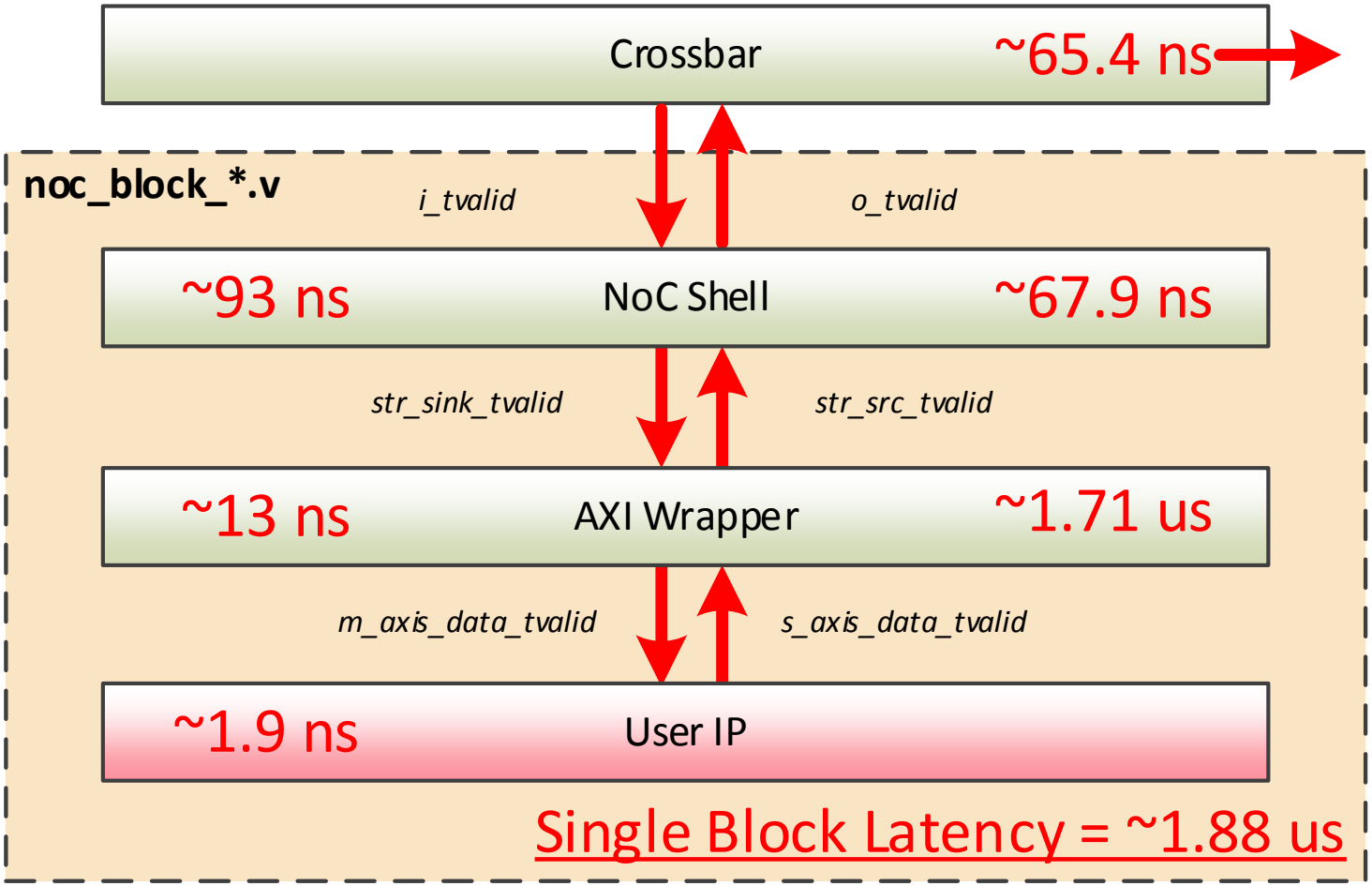


Crossbar Latency

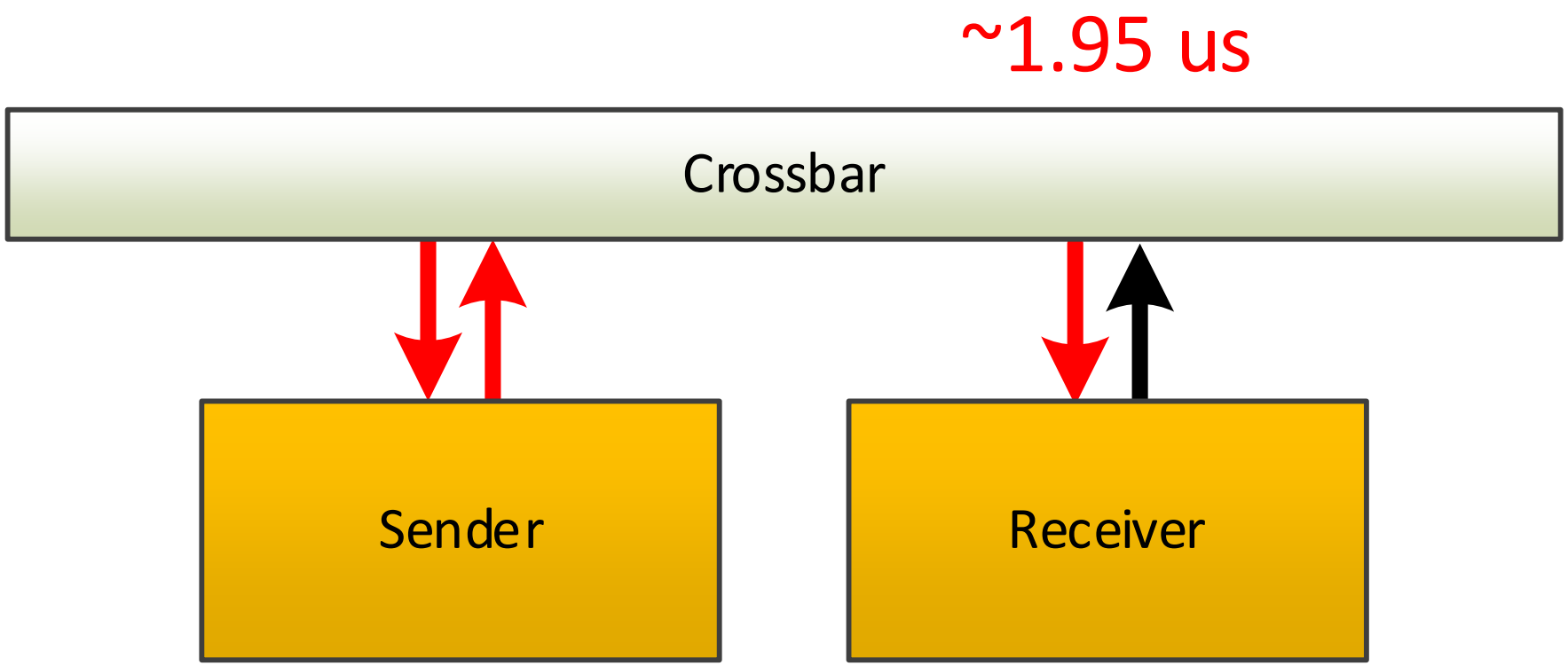


Latency Measurements

Total Block-to-Block Latency = ~1.95 us



Sender-to-Receiver Latency

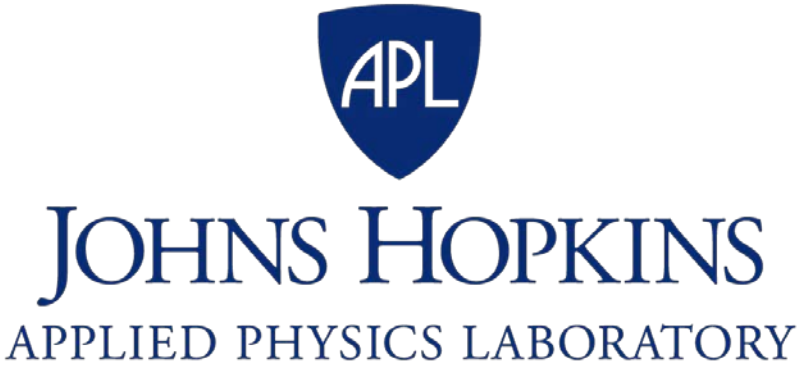


Conclusions and Future Work

- **Conclusions**
  - **RFNoC architecture overhead and reliance on a host computer may preclude applicability to problem sets that require small latencies**
    - **This is determined by the 2us latency seen here, which is equivalent to 390 FPGA clock cycles, to move data block-to-block without complex processing**
  - **Standalone "headless" operation/architecture configurations that don't require a host computer may facilitate wider applicability**
  - **RFNoC CE block capacity may also limit scalability and integration of complex designs for some applications**
- **Future Work**
  - **Evaluate longer (extended) AXI-stream packet sizes**
  - **Extend analysis to include more complex DSP algorithms**

References

- Braun, Martin and Cuervo, Nicolas. “Getting Started with RFNoC Development” (2017). https://kb.ettus.com/Getting_Started_with_RFNoC_Development
- Braun, Martin and Pendlum, Jonathon. “RFNoC: RF Network on Chip” (2014). https://www.ettus.com/content/files/RFNoC_Wireless_at_VT_Intro.pdf
- Braun, Martin, Pendlum, Jonathon, and Ettus, Matt. “RFNoC – RF Network-on-Chip” (2016). Proceedings GNU Radio Conference, 2016.
- Pendlum, Jonathon. “RFNoC Deep Dive: FPGA Side” (2014). https://www.ettus.com/content/files/RFNoC_Wireless_at_VT_FPGA.pdf.


APL
JOHNS HOPKINS
APPLIED PHYSICS LABORATORY

RFNoC Performance Review Additional Slides & Discussion

GRCON17

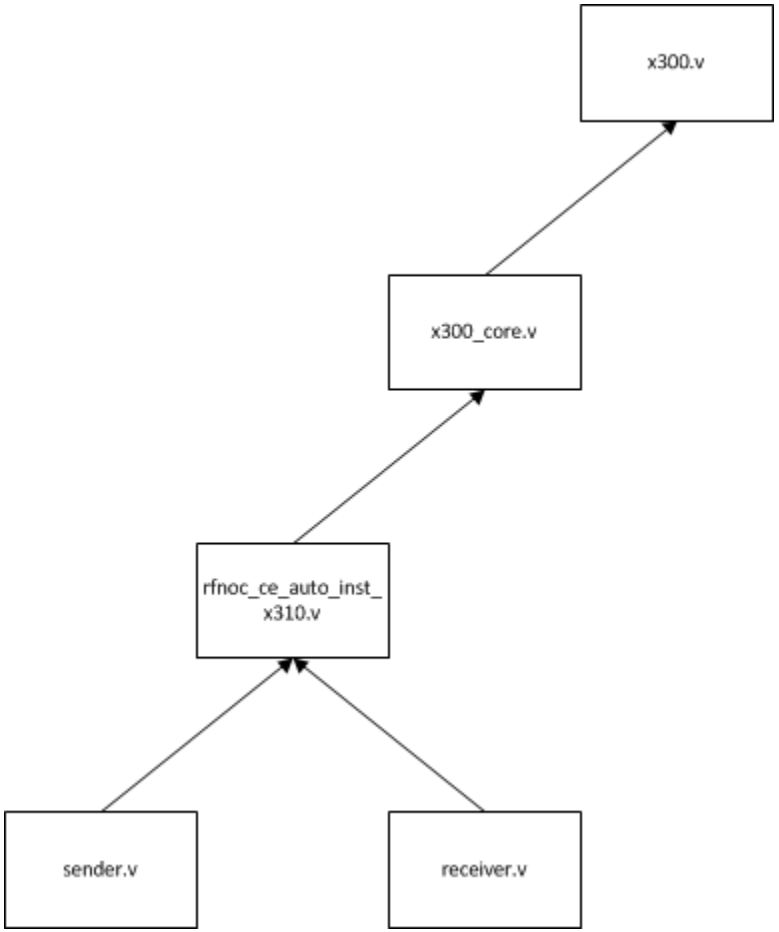
September 11-15, 2017

Josh Sunderlin

joshua.sunderlin@jhuapl.edu

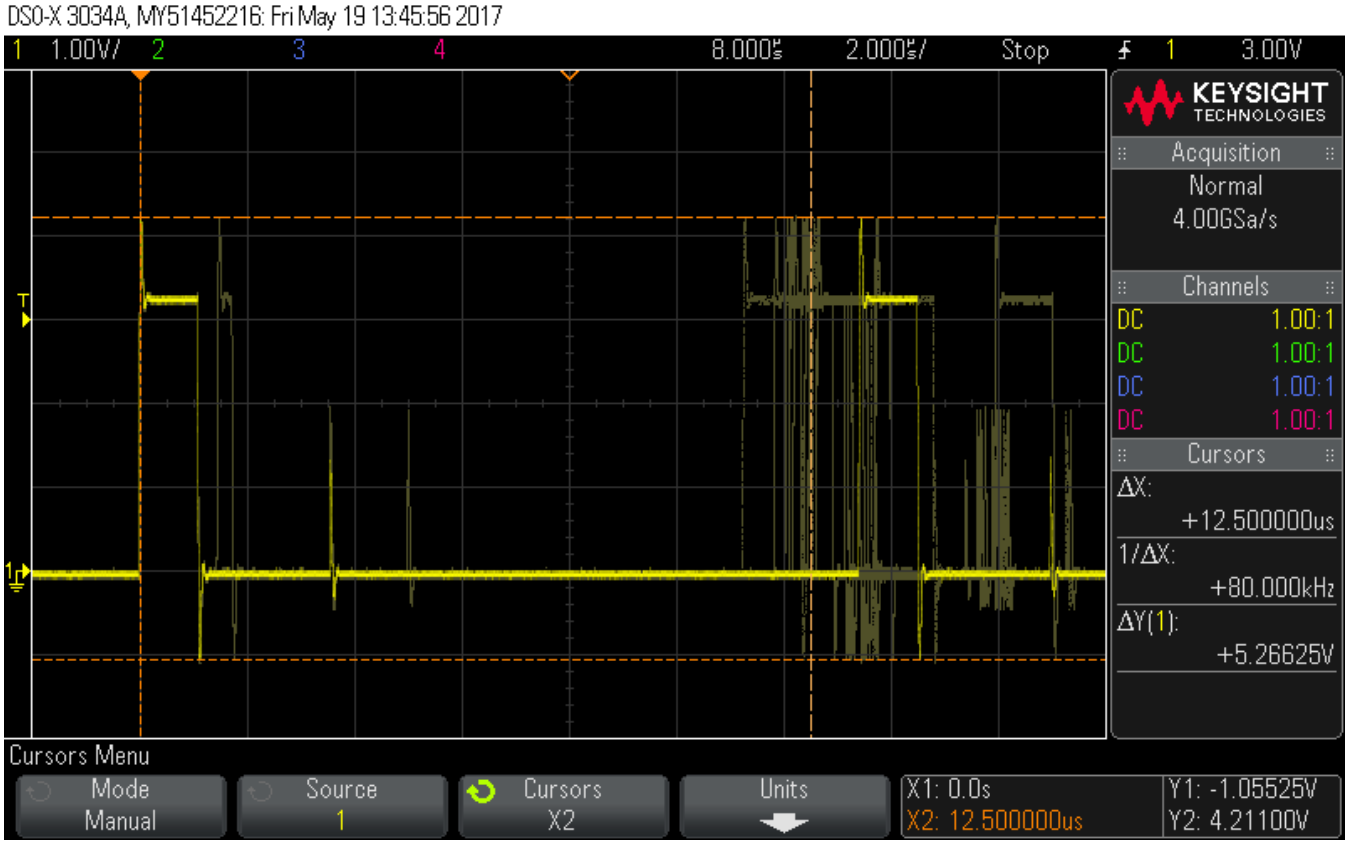
JOHNS HOPKINS
APPLIED PHYSICS LABORATORY

RFNoC Module Topology



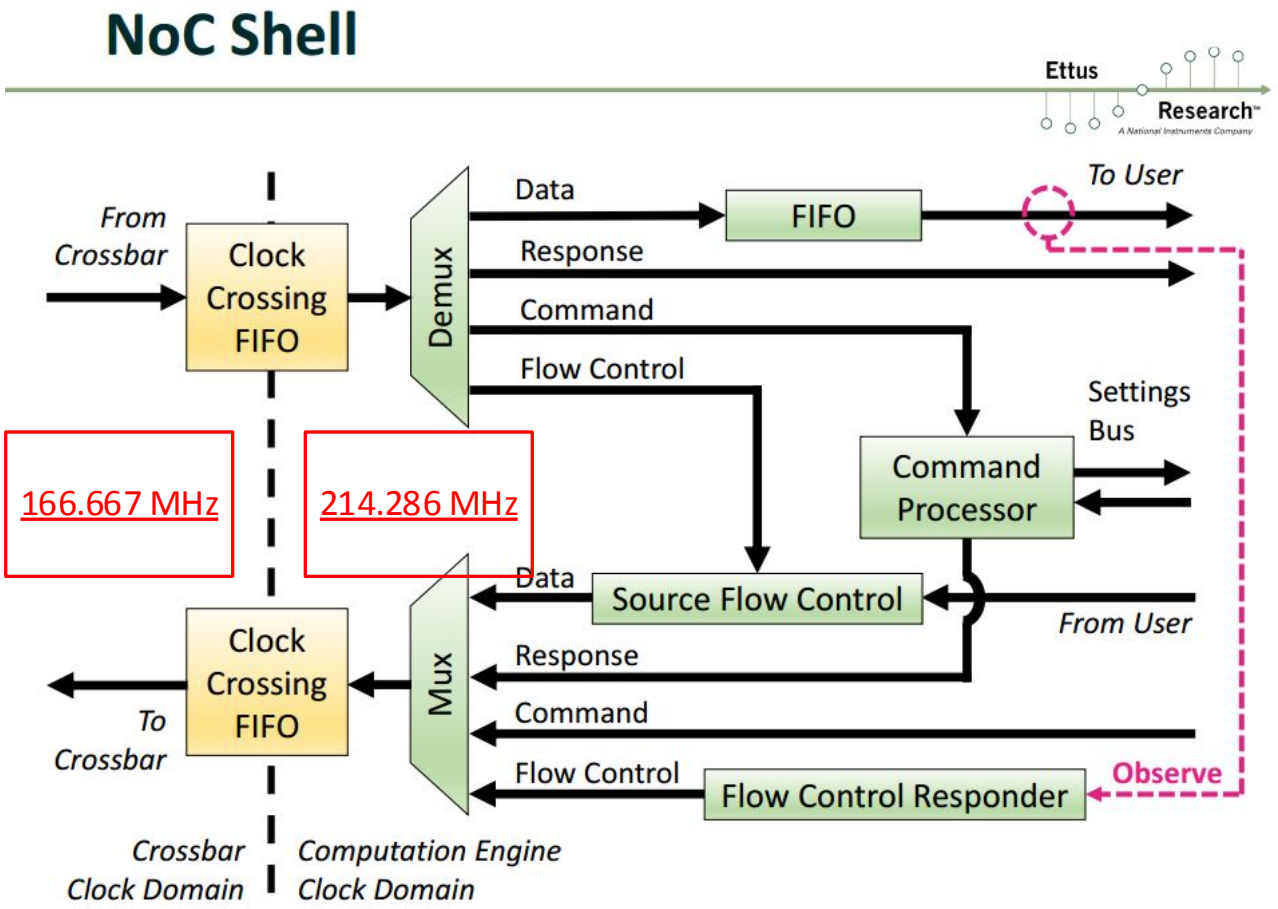
- GPIO Pins are accessed as output port out of x300.v module
- RFNoC block instantiation occurs in rfnoc_ce_auto_inst_x310.v

Data Stream from Host Computer



- **Mean Data Stream Periodicity: 12.706 us**
- **Standard Deviation (Jitter) Data Stream Periodicity: 1.2646 us**
- **Min/Max Data Stream Periodicity: 10.9044 us / 17.1844 us**

RFNoC Clock Frequencies



- Crossbar clock is 166.667 MHz, CE block clock is 214.286 MHz
- Radio (FPGA) clock is 200 MHz, ZPU/SC clock is 83.333 MHz