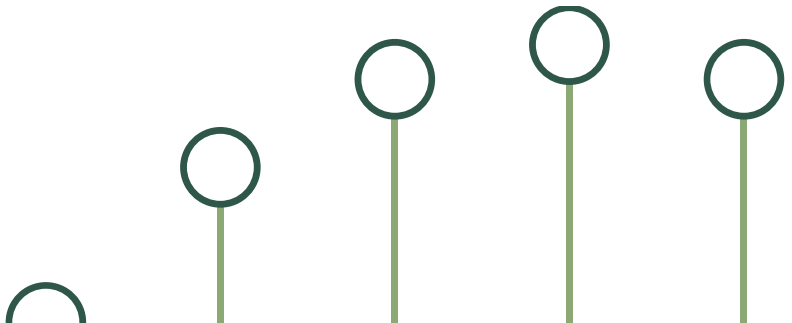
Ettus Research™
A National Instruments Company

RFNoC™ Deep Dive: FPGA Side

Jonathon Pendlum

5/28/2014



Outline

Ettus Research™
A National Instruments Company

- Getting started with RFNoC – FPGA
- Anatomy of a Computation Engine
- FFT RFNoC Block walkthrough
- Design Tips

Hardware / Software Prerequisites

Ettus Research™ A National Instruments Company

- Hardware:
  - PC or Laptop, decent amount of RAM (4+ GB)
  - USRP X300, X310, or E310
  - Xilinx JTAG debugger module (optional)
    - Only needed for E310, X3x0 series has built in USB JTAG

- Software:
  - Linux based OS
  - Xilinx Vivado 2014.4
    - E310 can use free Webpack version
  - Modelsim (optional)
  - UHD
  - GNU Radio (optional, but highly recommended)
    - gr-ettus

FPGA Directory Structure

Ettus Research™ A National Instruments Company

- **usrp3** -- Third gen devices (X300, E310, B200)
  - **tools** -- Build infrastructure software
  - **sim** -- Simulations
  - **lib** -- HDL shared between devices
    - **ip** -- IP cores shared between devices
    - **rfnoc** -- NoC Shell, AXI Wrapper, NoC blocks, basic blocks
  - **top** -- Device specific files & toplevel
    - **x300**
      - **ip** -- Device specific IP cores
      - x300.v
      - rfnoc_ce_auto_inst_x300.v
      - Makefile

Building RFNoC FPGA Images

Ettus Research™ A National Instruments Company

- Make sure Vivado 2014.4 installed
  - Helpful if installed in default directory (in /opt/Xilinx)
- Go to toplevel directory
  - usrp3/top/x300, e300
- source setupenv.sh
  - Sets up Xilinx tools
- make X310_HGS_RFNOC (or E310_RFNOC)
  - To launch Vivado GUI: make GUI=1 X310_HGS_RFNOC
- Build takes about an hour

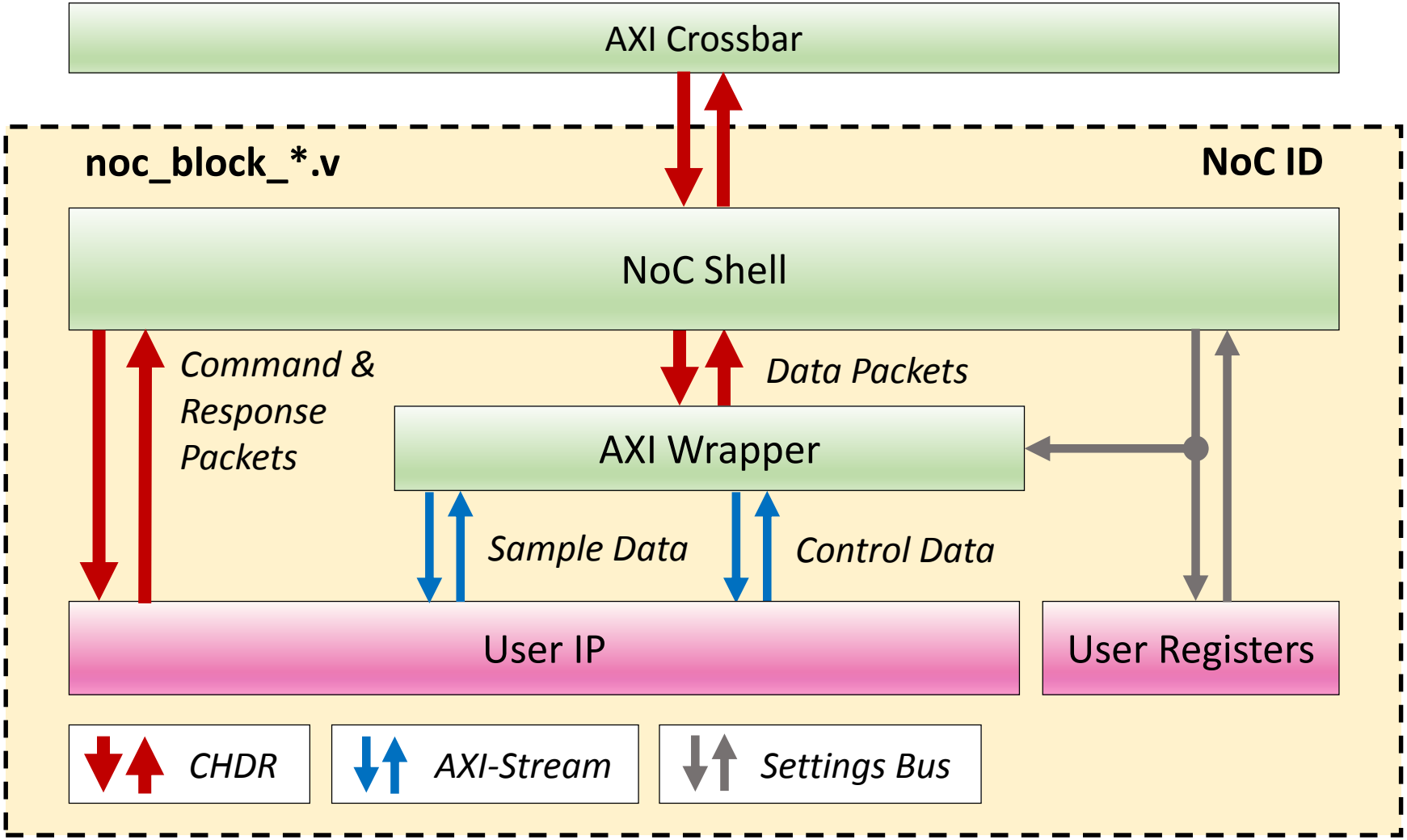
Building RFNoC FPGA Images

Ettus Research™
A National Instruments Company

- Output in build directory
  - usrp_x310_fpga_HGS_RFNOC.bit
  - Includes a report file with utilization / timing info
- Program bitfile with usrp_x3xx_fpga_burner
- make clean
- make cleanall -- also removes build-ip

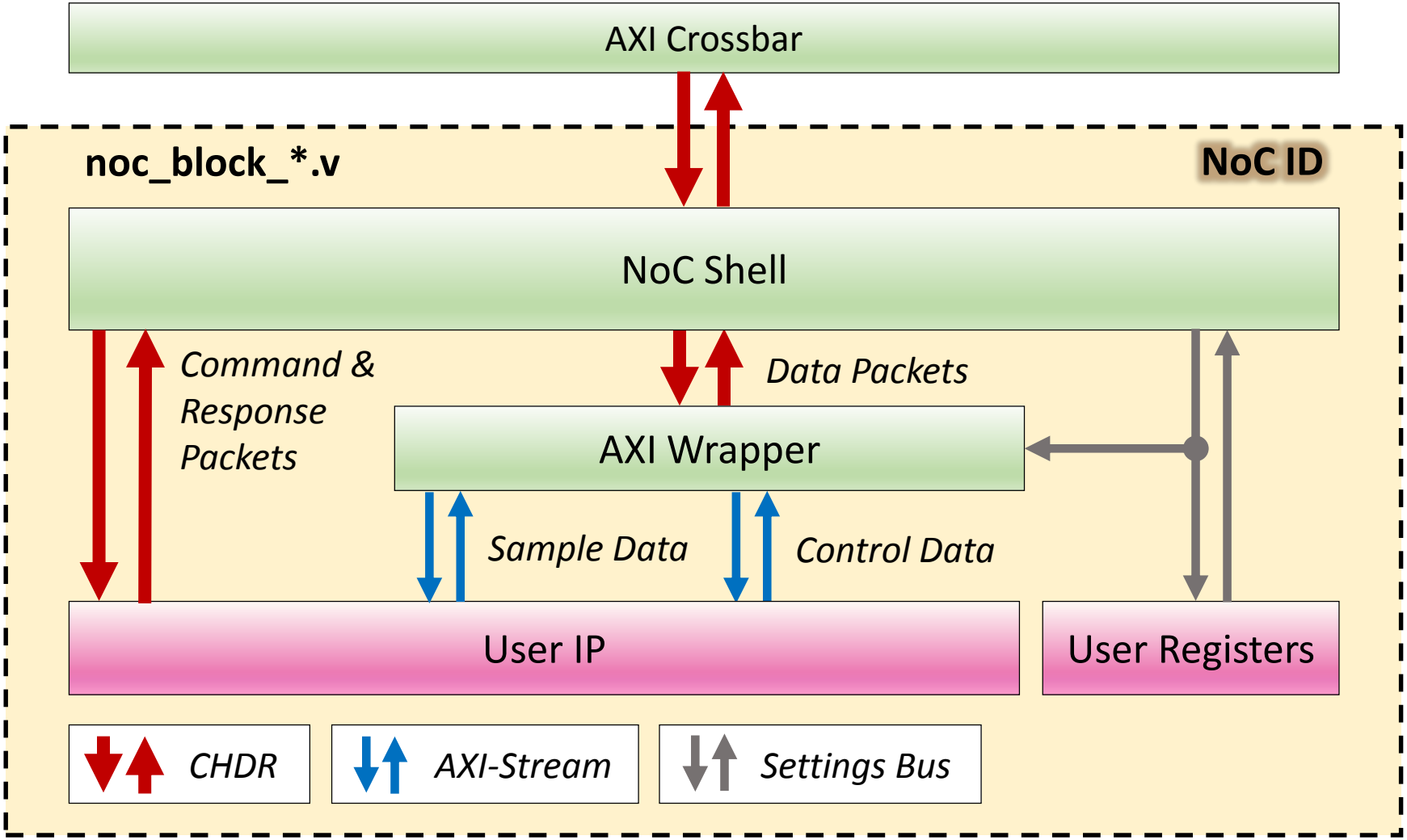
Anatomy of a Computation Engine

Ettus Research™
A National Instruments Company



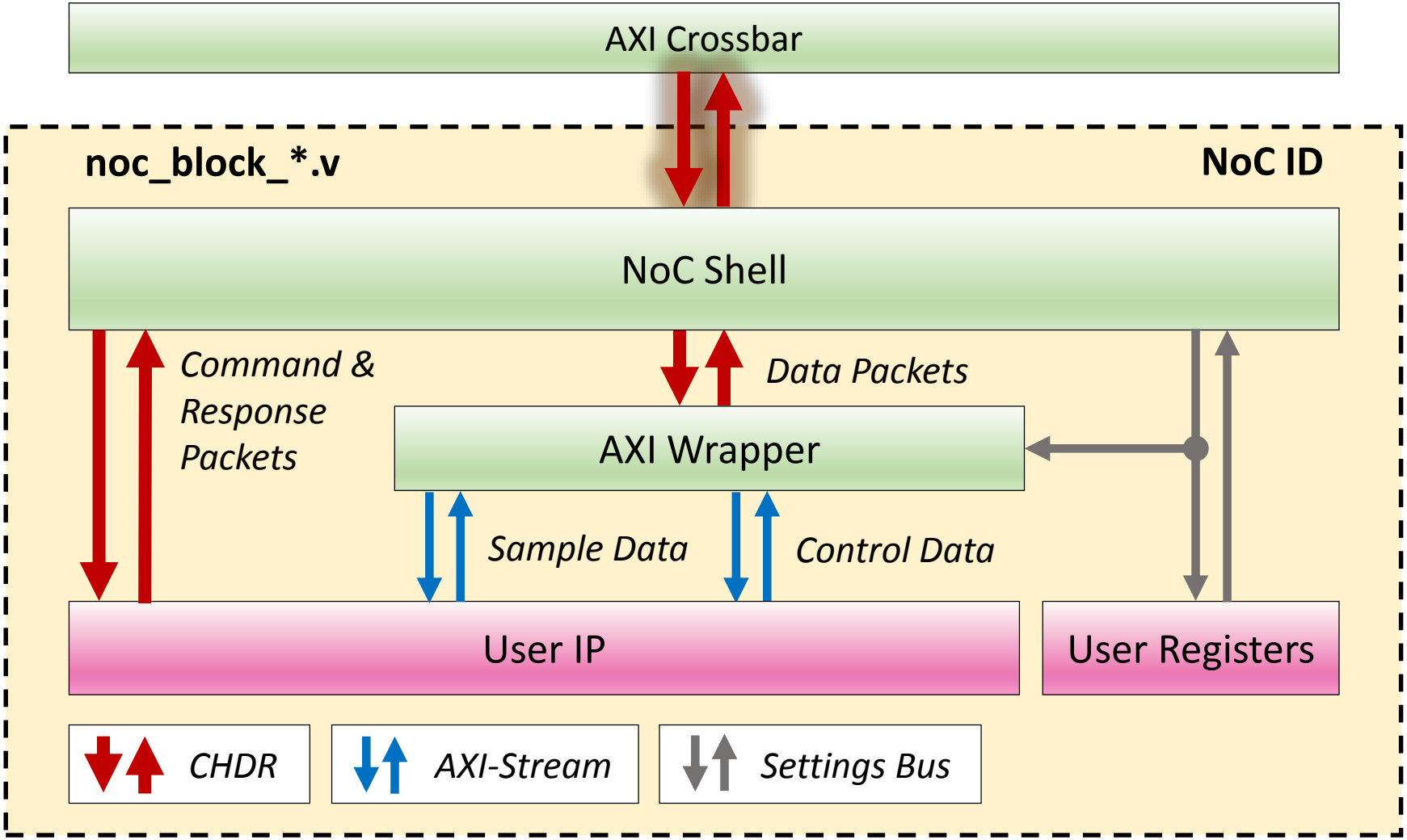
Anatomy of a Computation Engine

Ettus Research™
A National Instruments Company



Anatomy of a Computation Engine

Ettus Research™
A National Instruments Company



CHDR Packet Protocol

Ettus Research™ A National Instruments Company

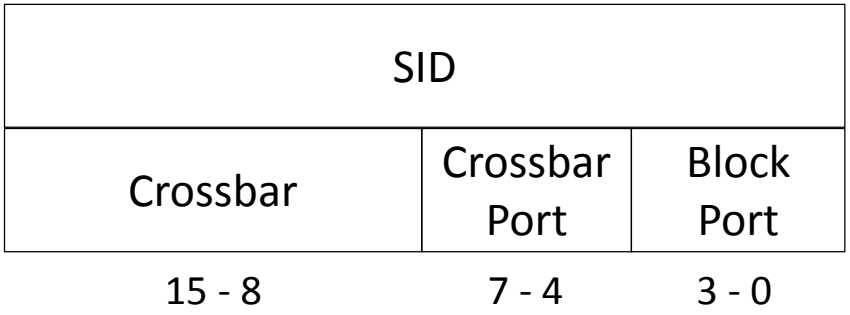
Pkt Type	Has Time	EOB	Seq #	Length (in bytes)	SRC SID	DST SID
63 - 62	61	60	59 - 48	47 - 32	31 - 16	15 - 0
Fractional Time (Optional, Has Time = 1)						
Payload						
…						

- Packet type based on bits 63, 62, & 60

63	62	60	Packet Type
0	0	0	Data
0	0	1	Data (End of Burst)
0	1	0	Flow Control
1	0	0	Command
1	1	0	Response
1	1	1	Response (Error)

Stream IDs

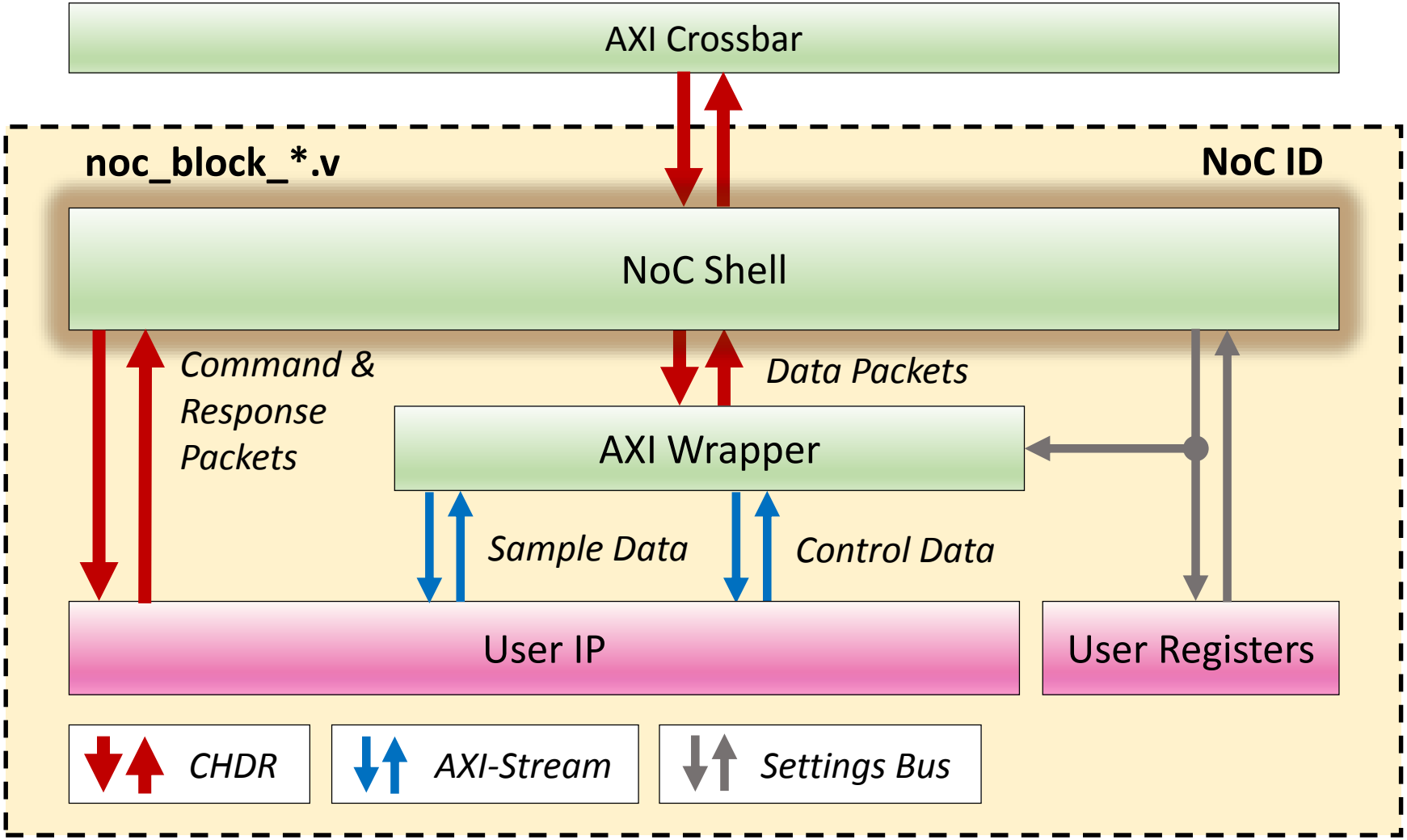
Ettus Research™
A National Instruments Company



- 16 bit Stream ID
  - 256 unique crossbar (or device) IDs
  - 16 ports per crossbar
  - 16 ports per block
- Example Crossbar ID: 2, Port: 1, Block Port: 0
  SID: 2.1.0

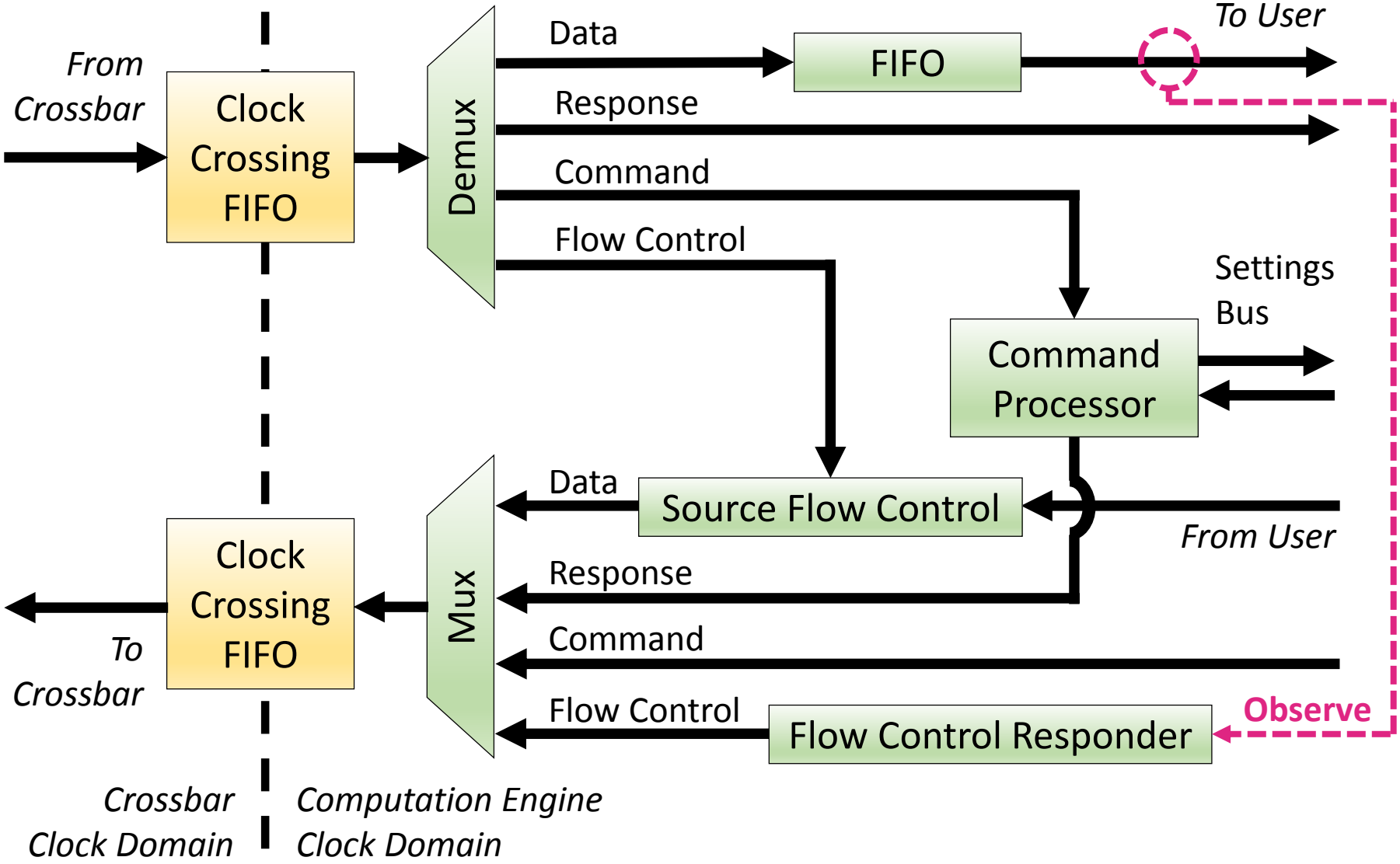
Anatomy of a Computation Engine

Ettus Research™
A National Instruments Company



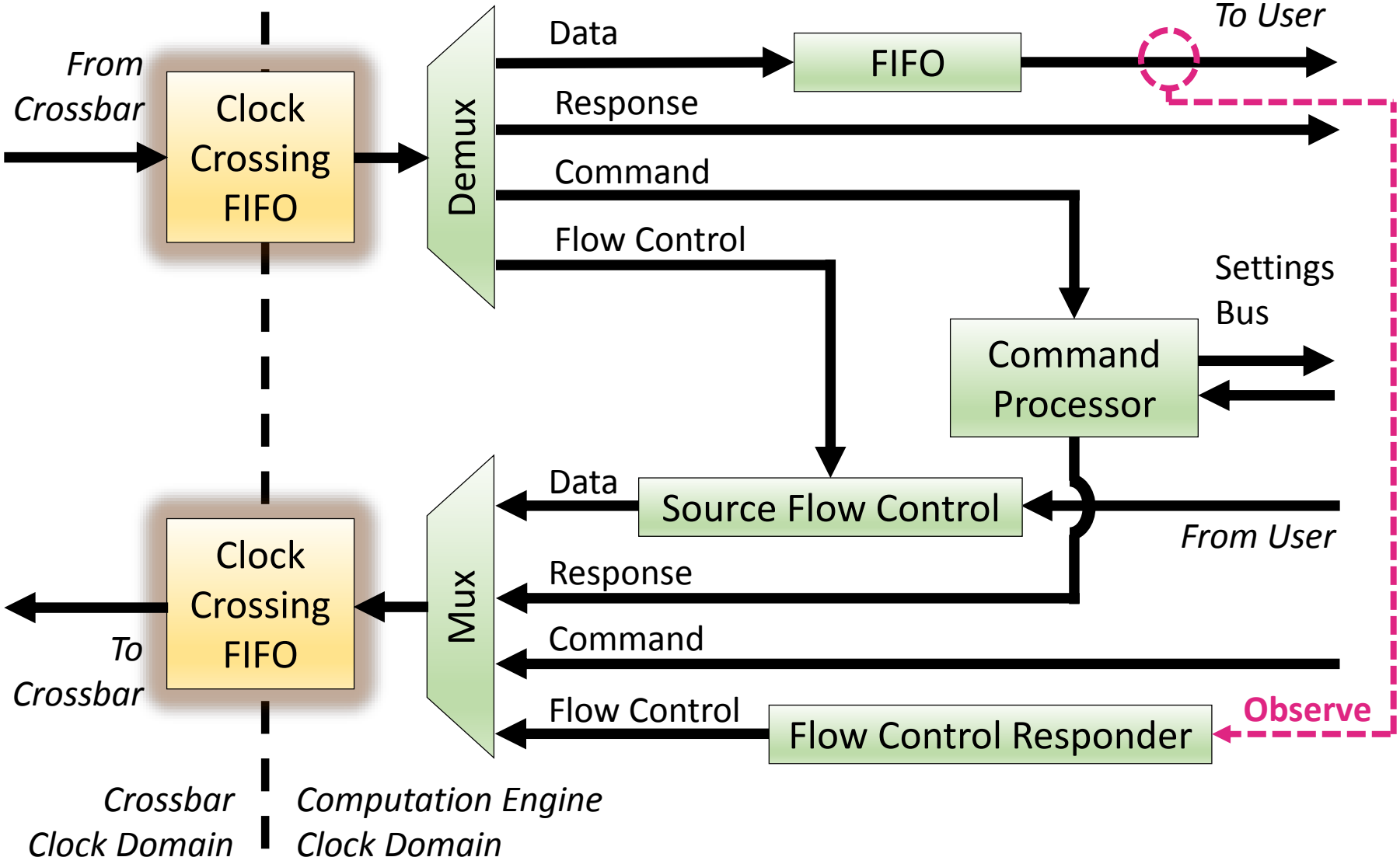
NoC Shell

Ettus Research™ A National Instruments Company

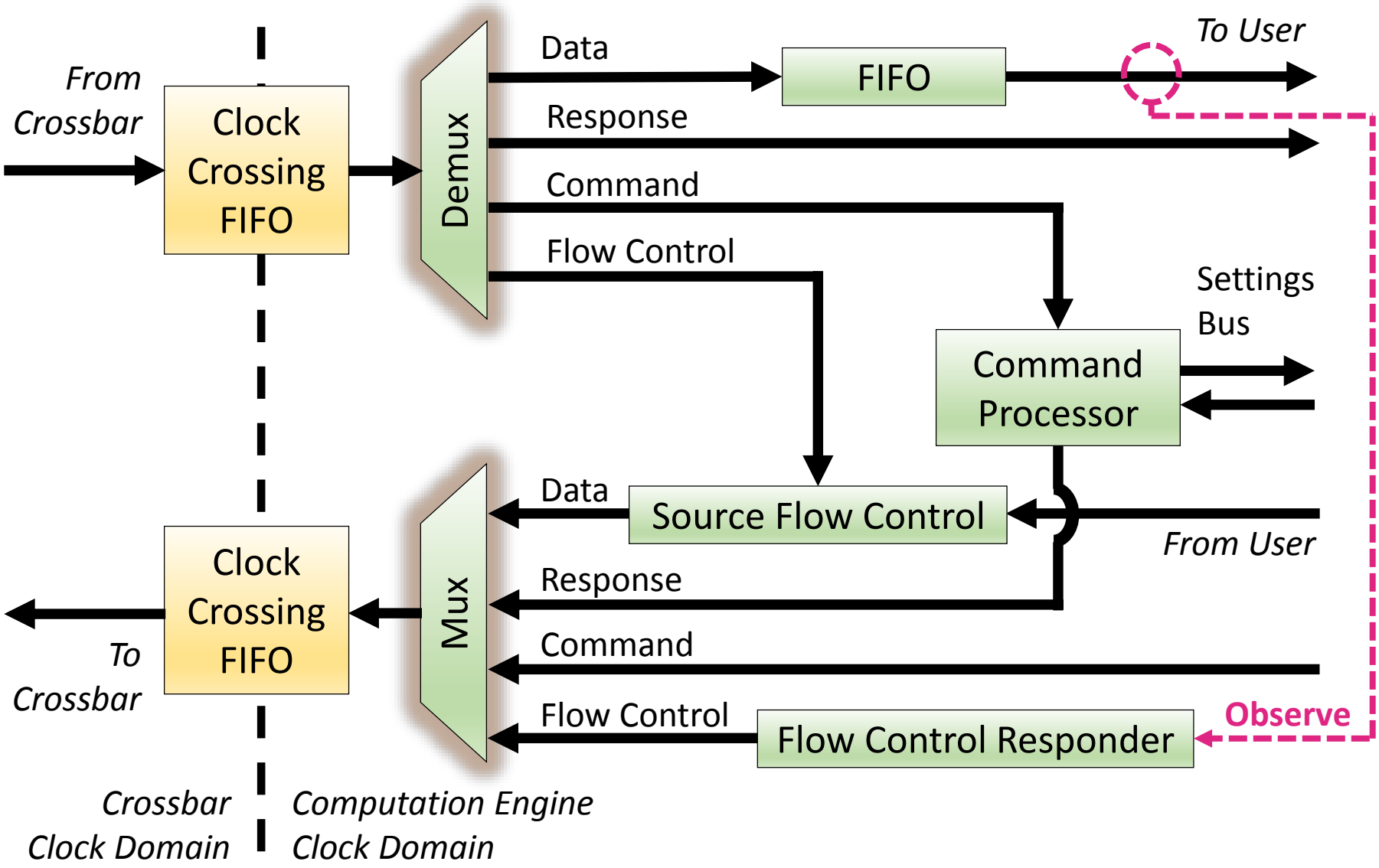


NoC Shell

Ettus Research™ A National Instruments Company

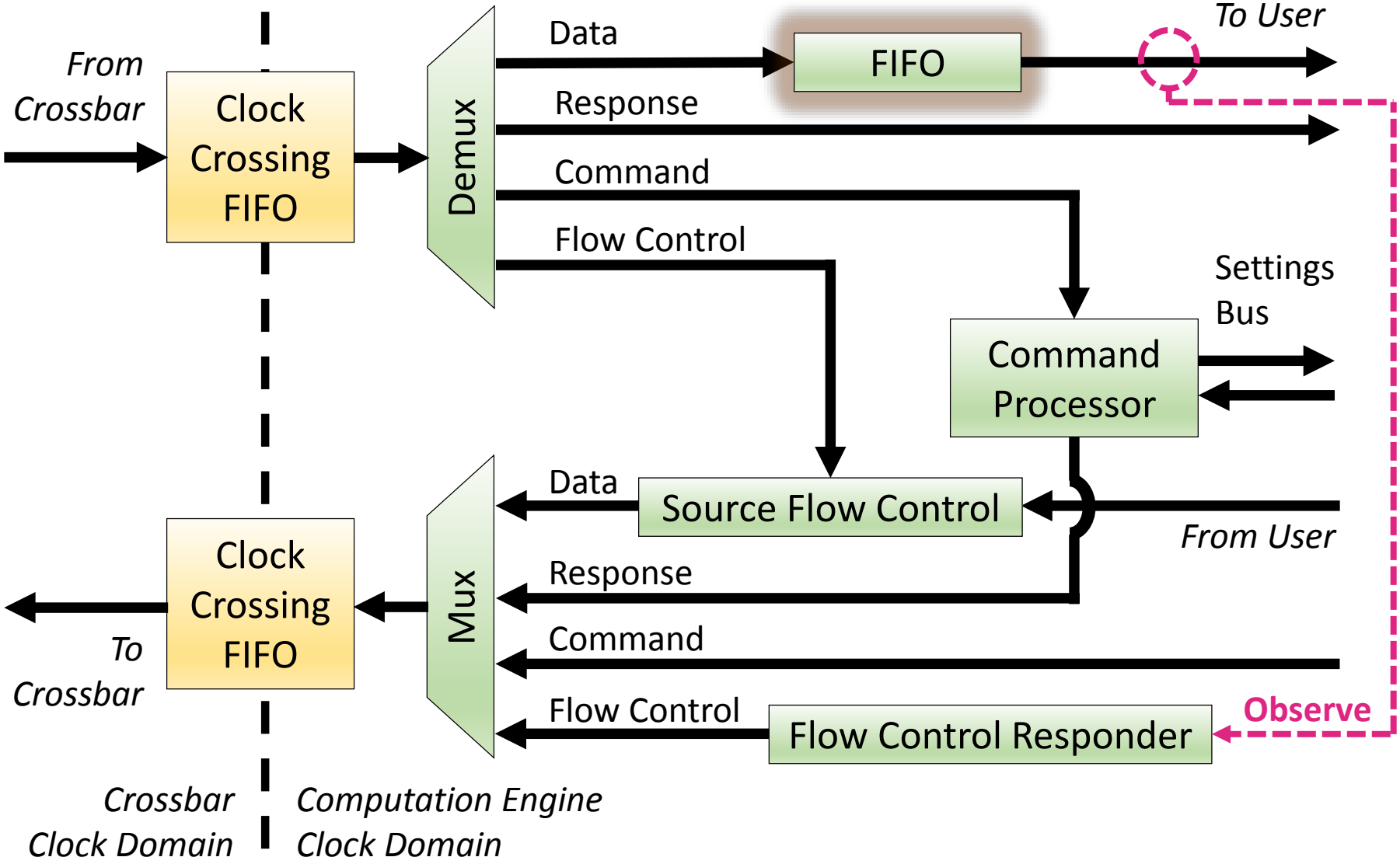


NoC Shell


From Crossbar
Clock Crossing FIFO
Demux
Data
FIFO
To User
Response
Command
Flow Control
Settings Bus
Command Processor
Data
Source Flow Control
From User
Clock Crossing FIFO
Mux
Response
To Crossbar
Command
Flow Control
Flow Control Responder
Observe
Crossbar Clock Domain
Computation Engine Clock Domain

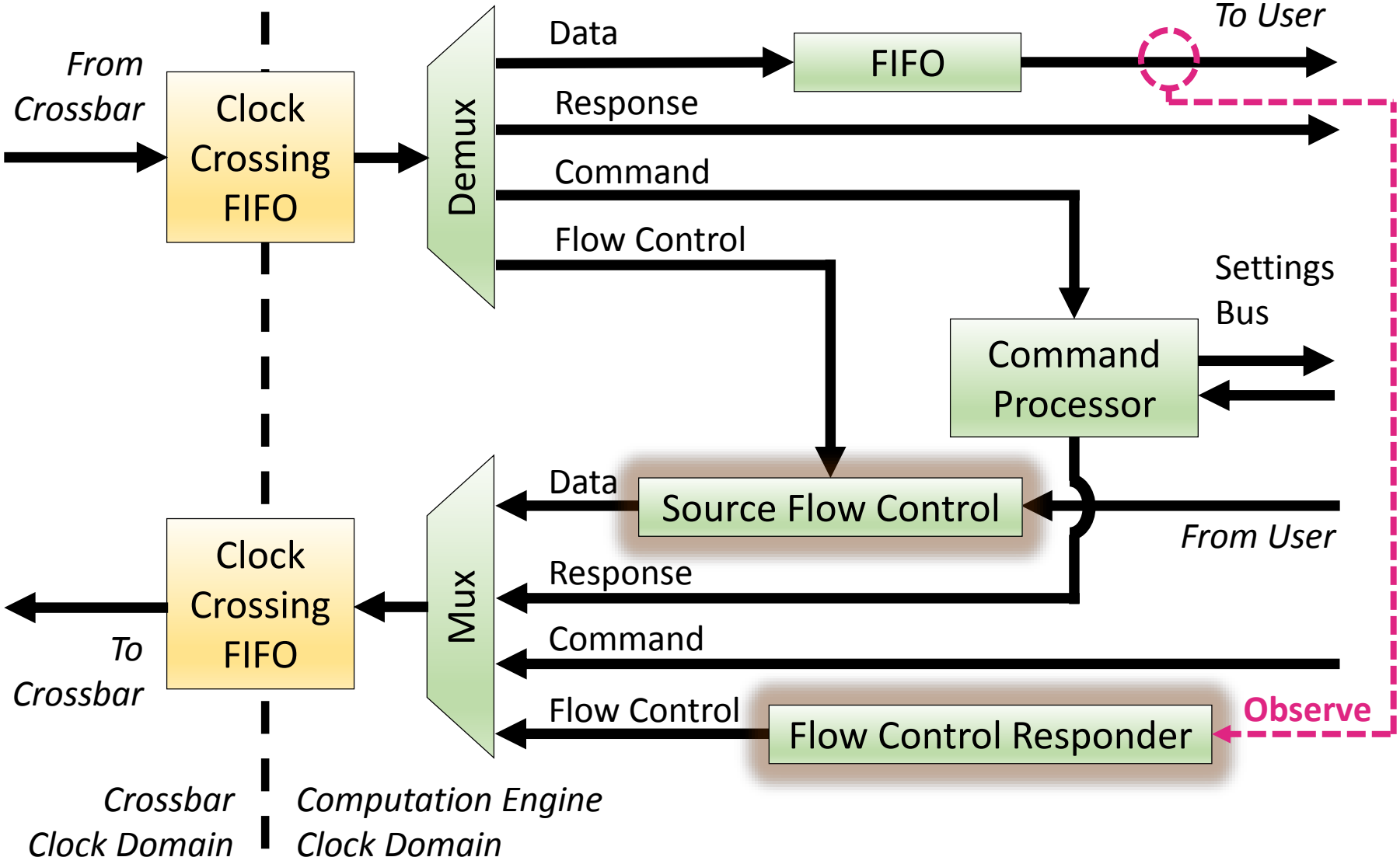
NoC Shell

Ettus Research™ A National Instruments Company



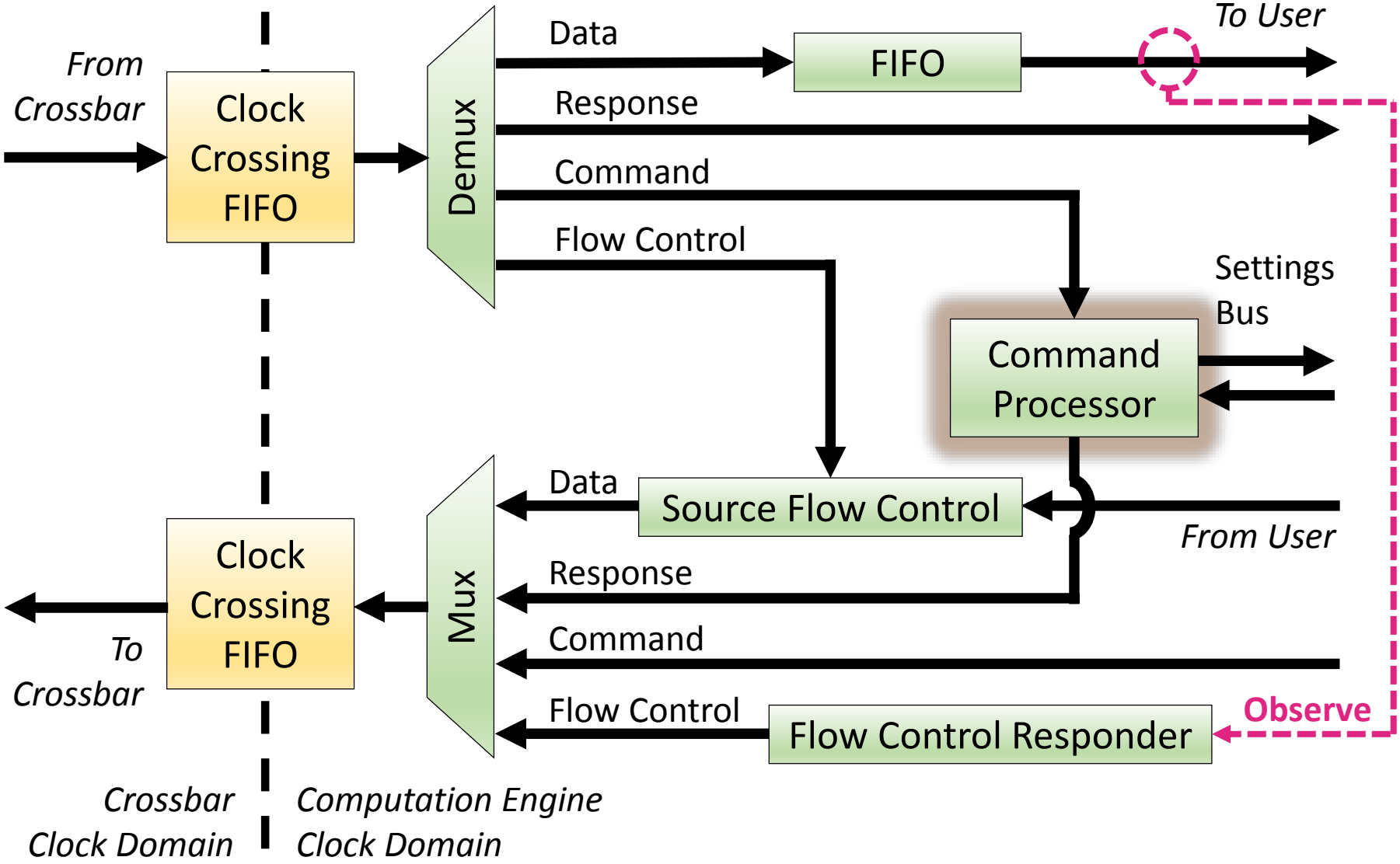
NoC Shell

Ettus Research™ A National Instruments Company



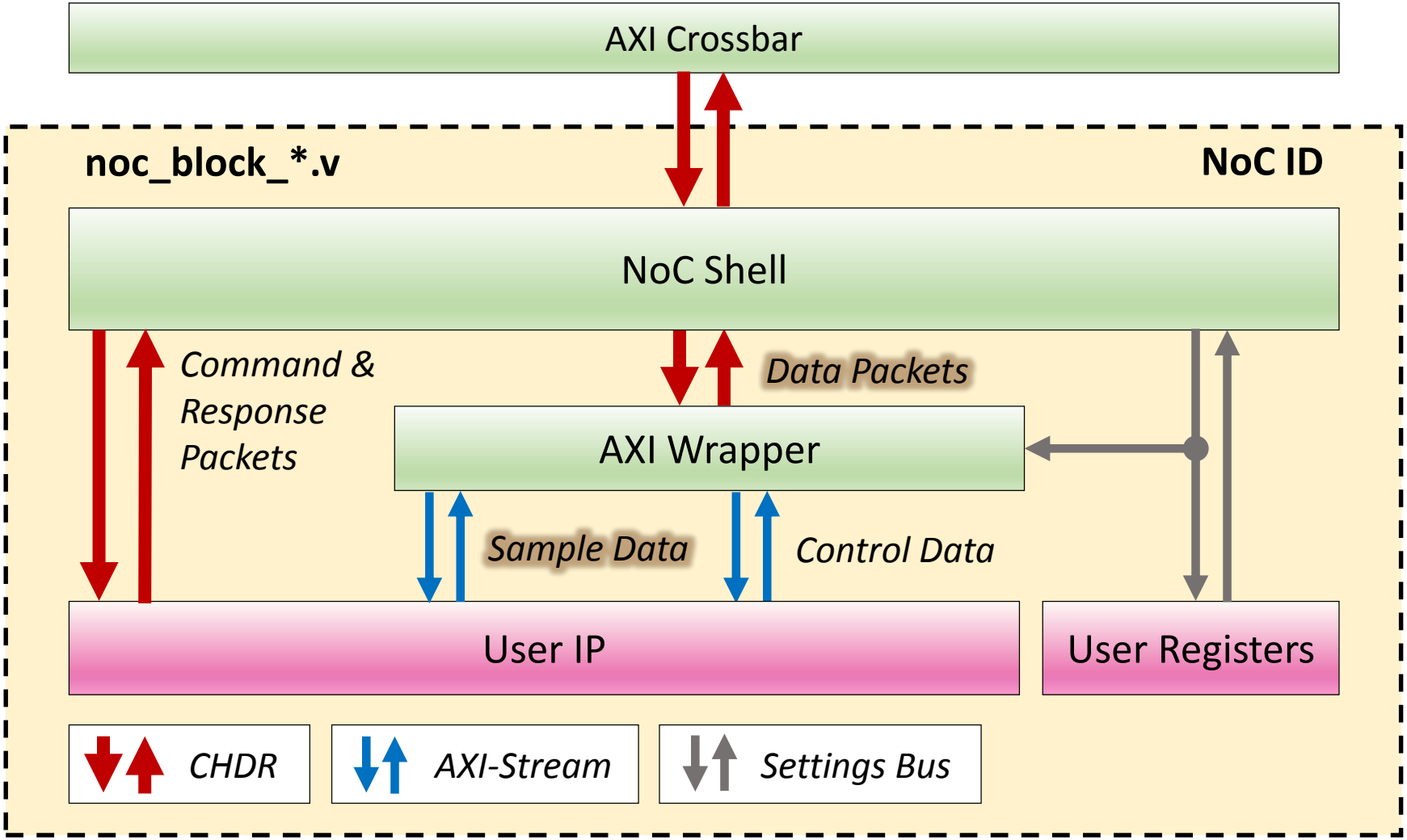
NoC Shell

Ettus Research™
A National Instruments Company



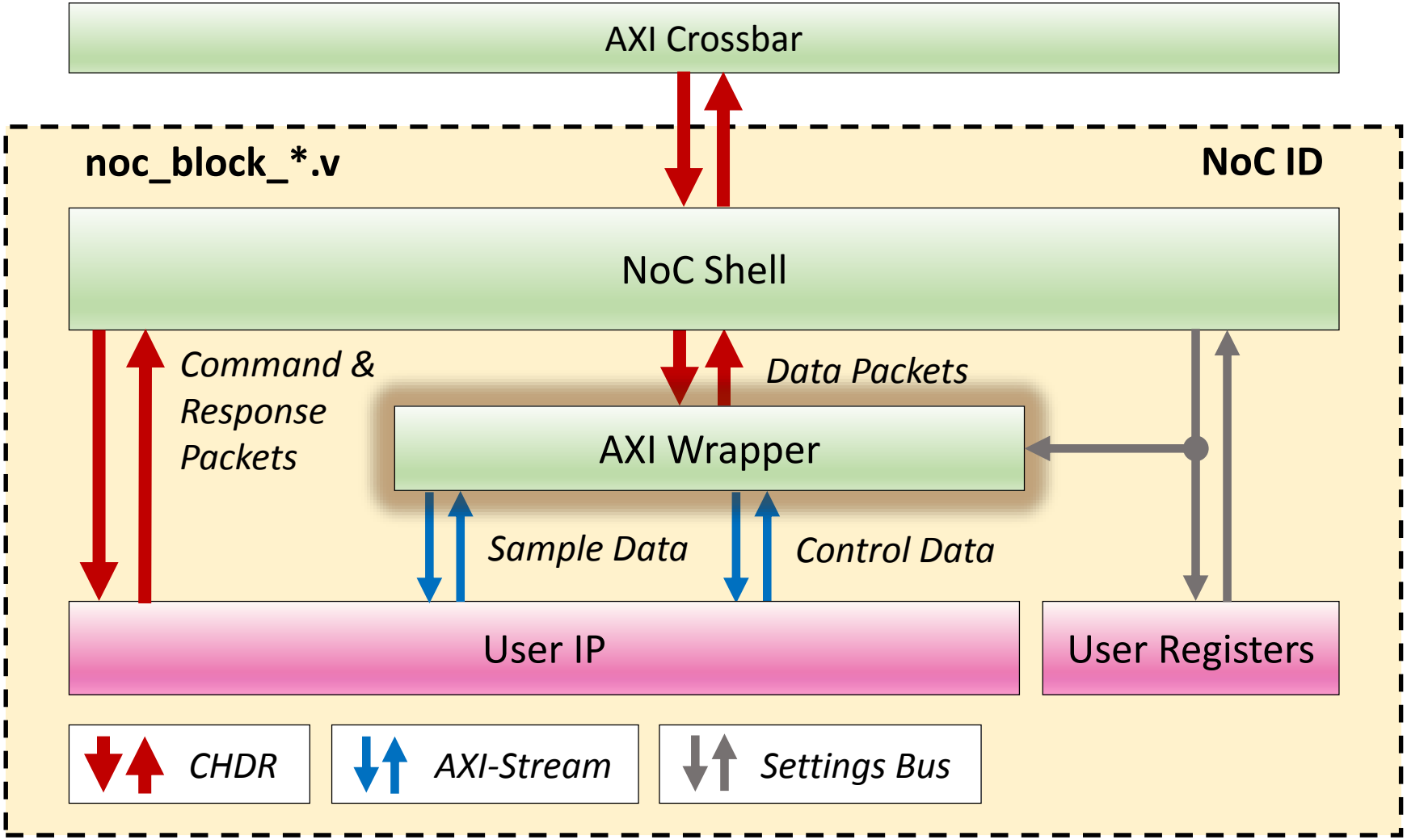
Anatomy of a Computation Engine

Ettus Research™
A National Instruments Company



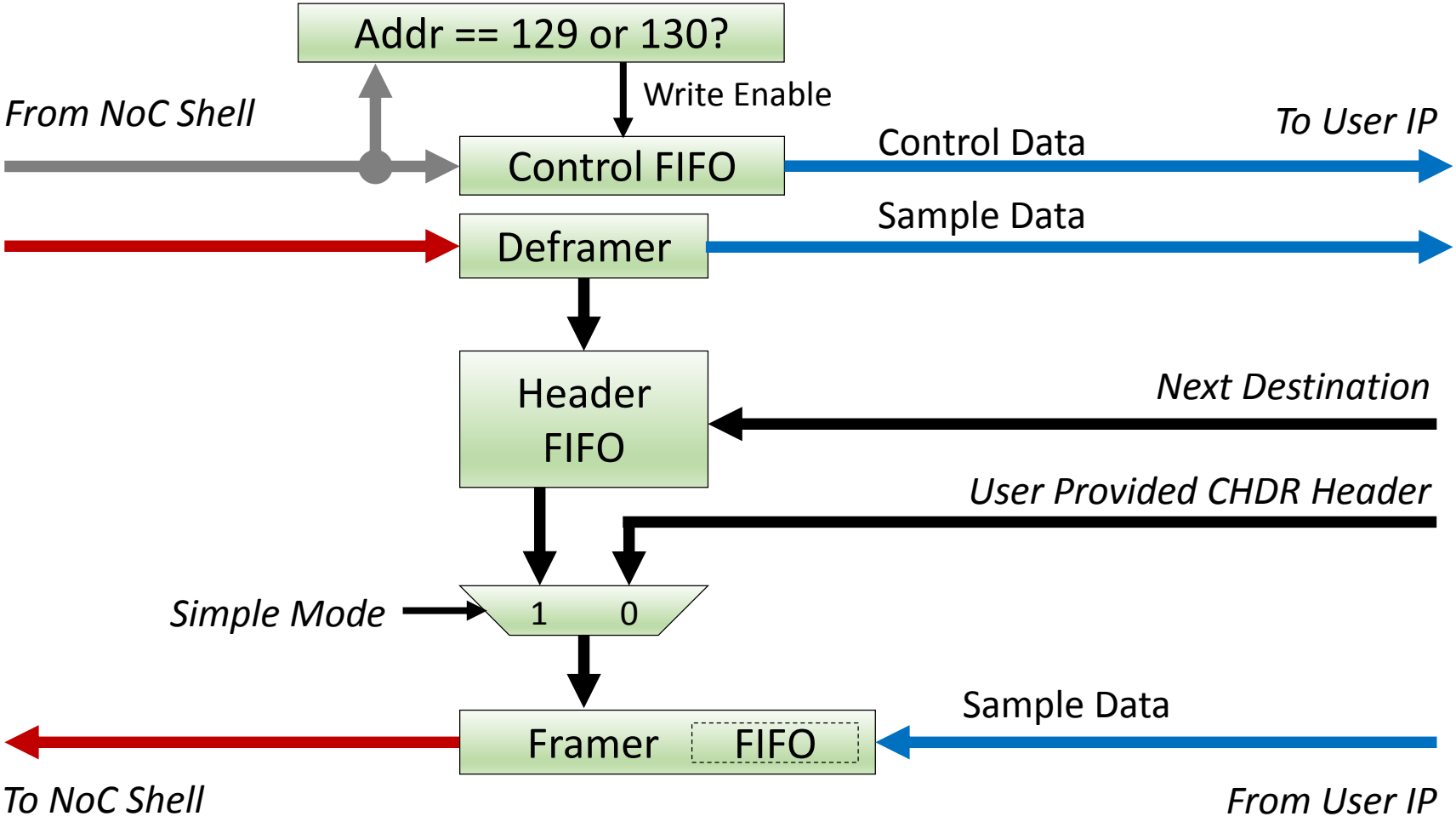
Anatomy of a Computation Engine

Ettus Research™
A National Instruments Company



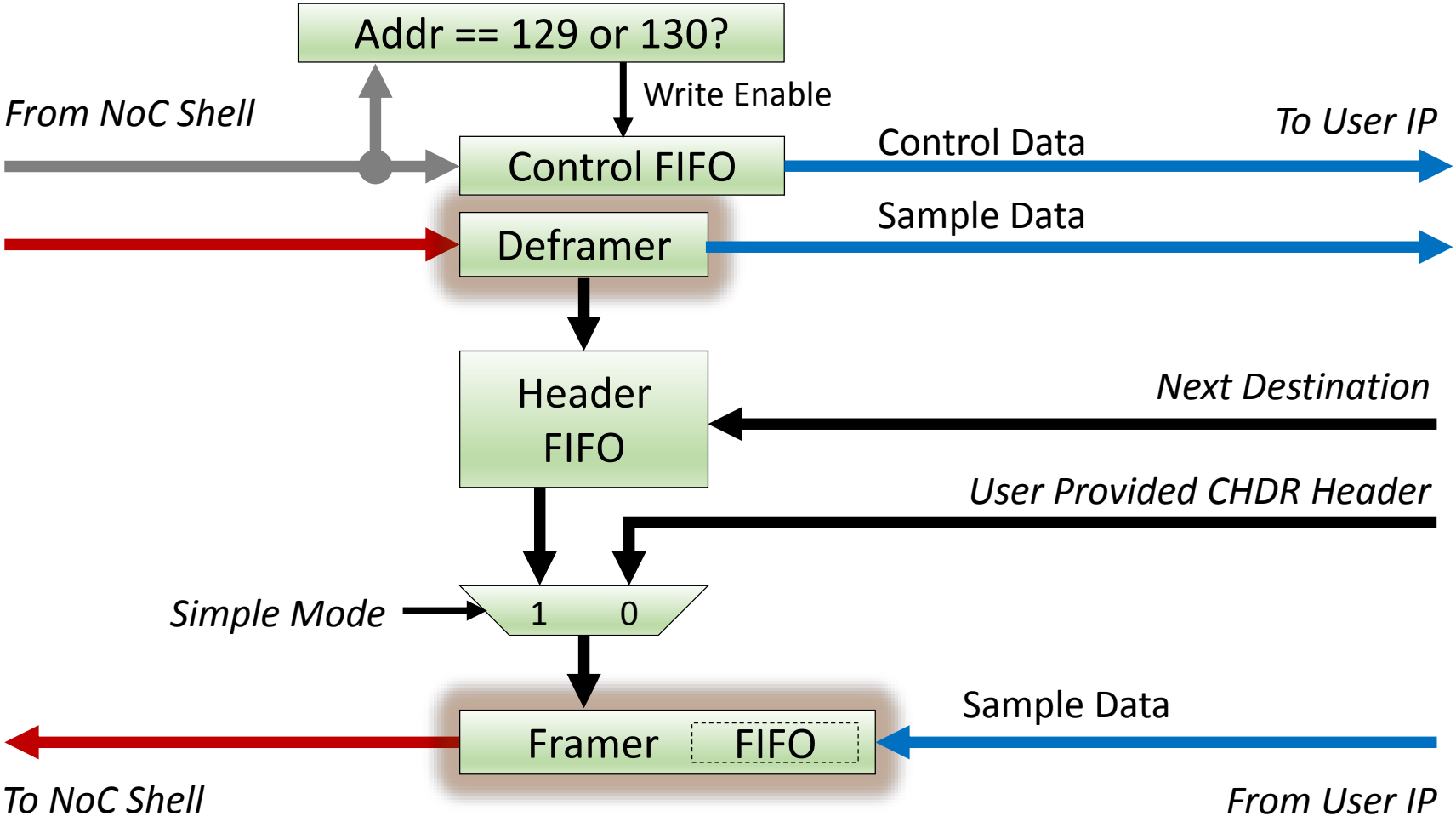
AXI Wrapper

Ettus Research™
A National Instruments Company



AXI Wrapper

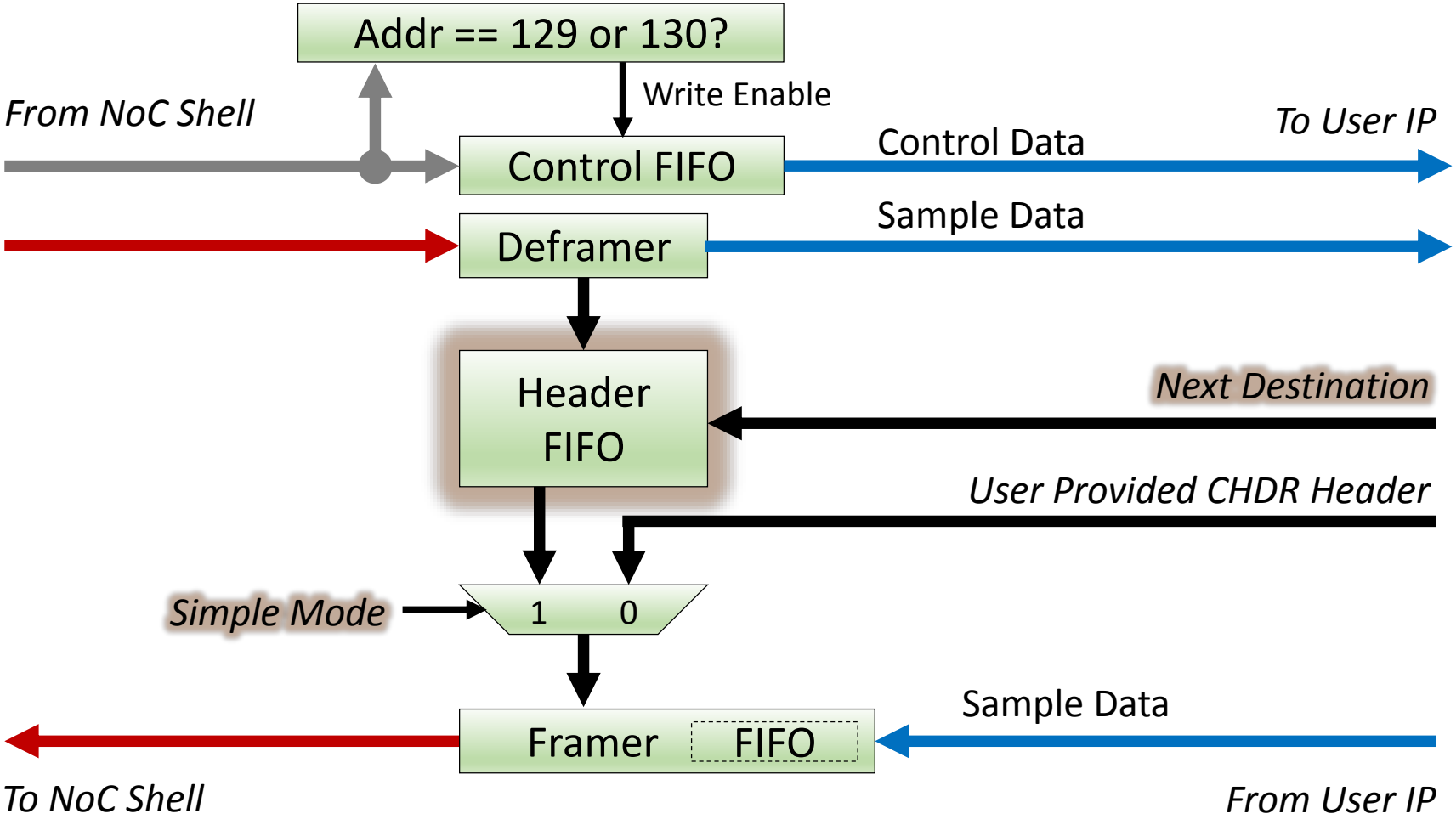
Ettus Research™
A National Instruments Company



CHDR | AXI-Stream | Settings Bus

AXI Wrapper

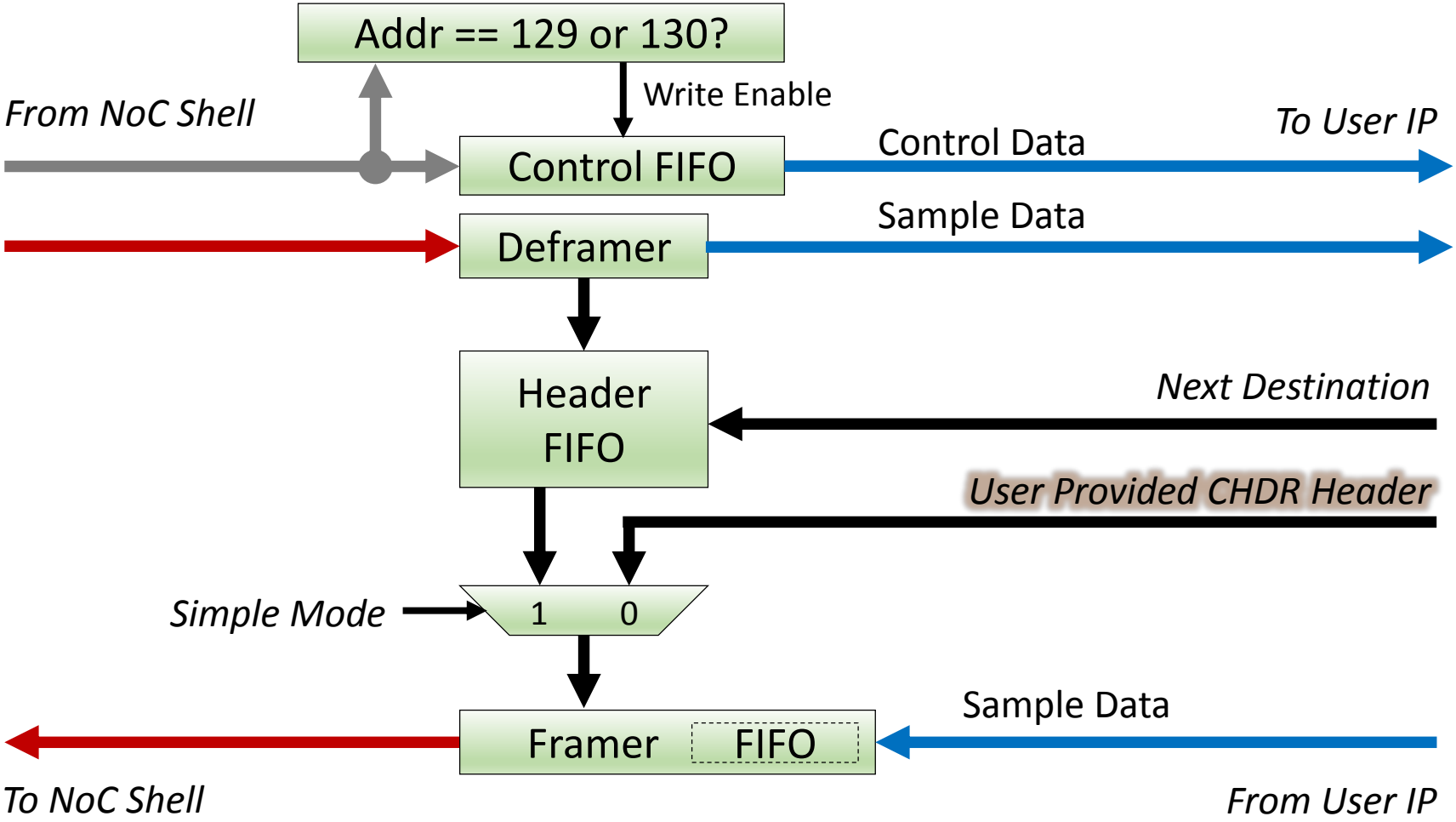
Ettus Research™
A National Instruments Company



CHDR | AXI-Stream | Settings Bus

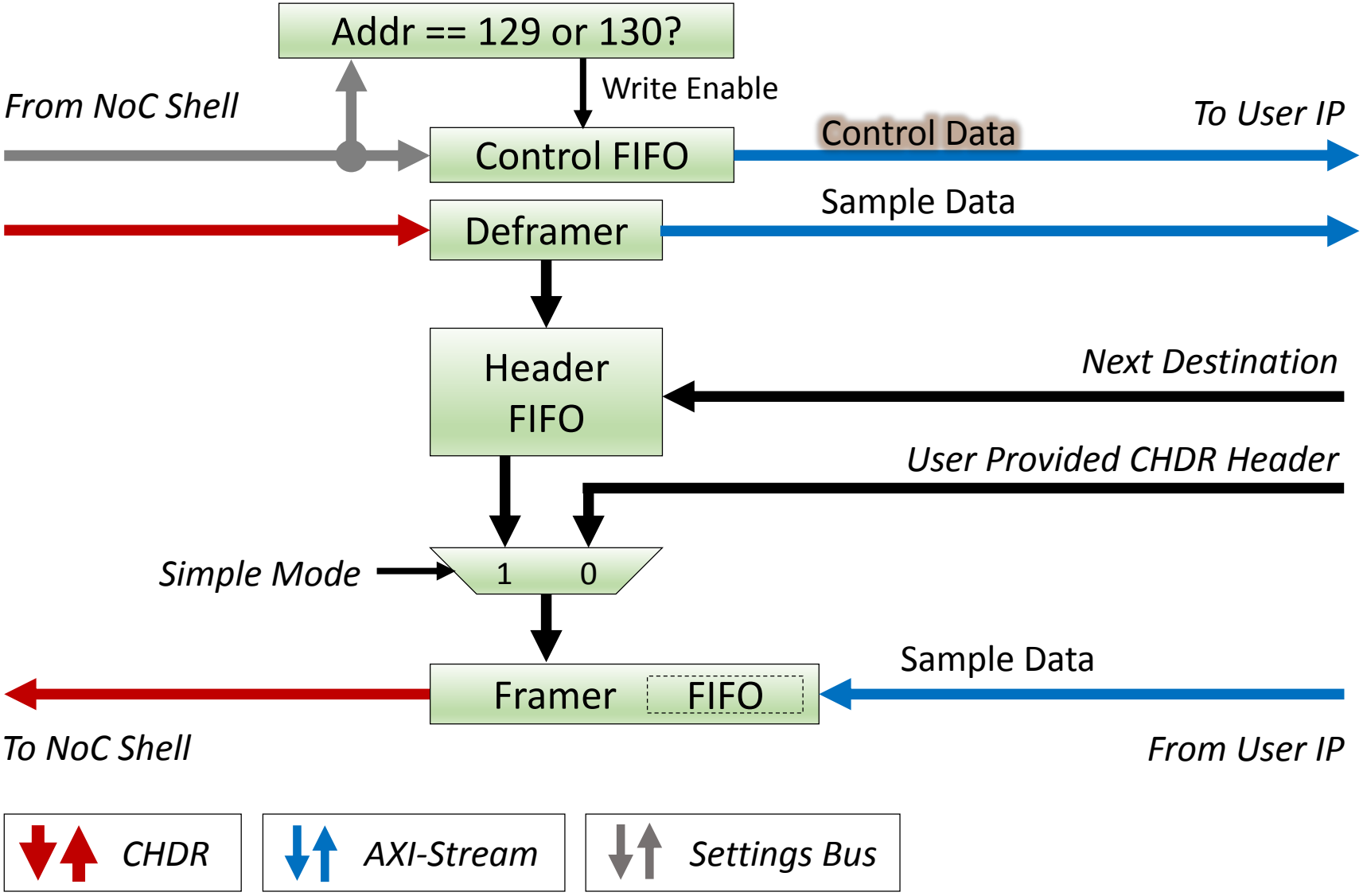
AXI Wrapper

Ettus Research™
A National Instruments Company



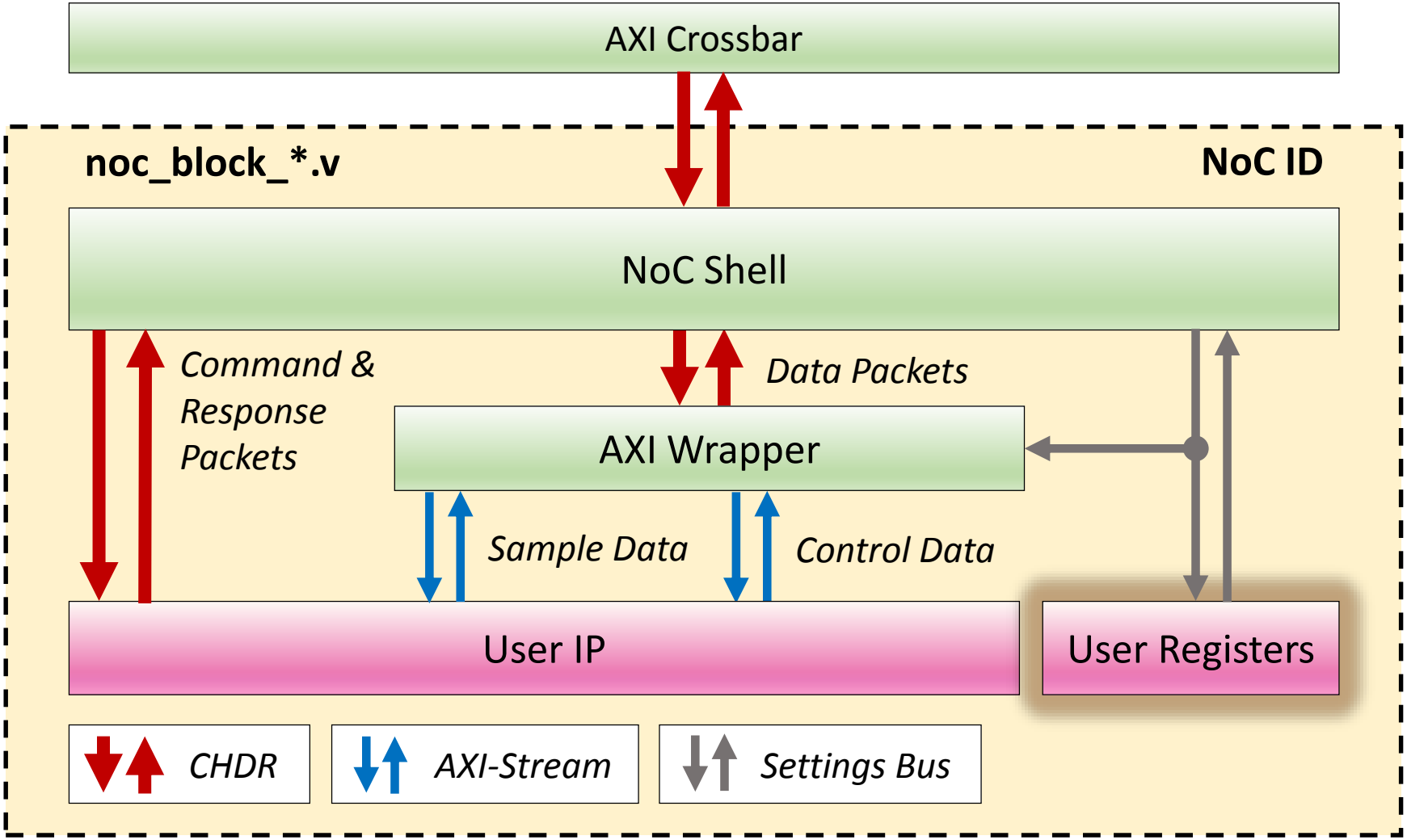
AXI Wrapper

Ettus Research™
A National Instruments Company



Anatomy of a Computation Engine

Ettus Research™
A National Instruments Company



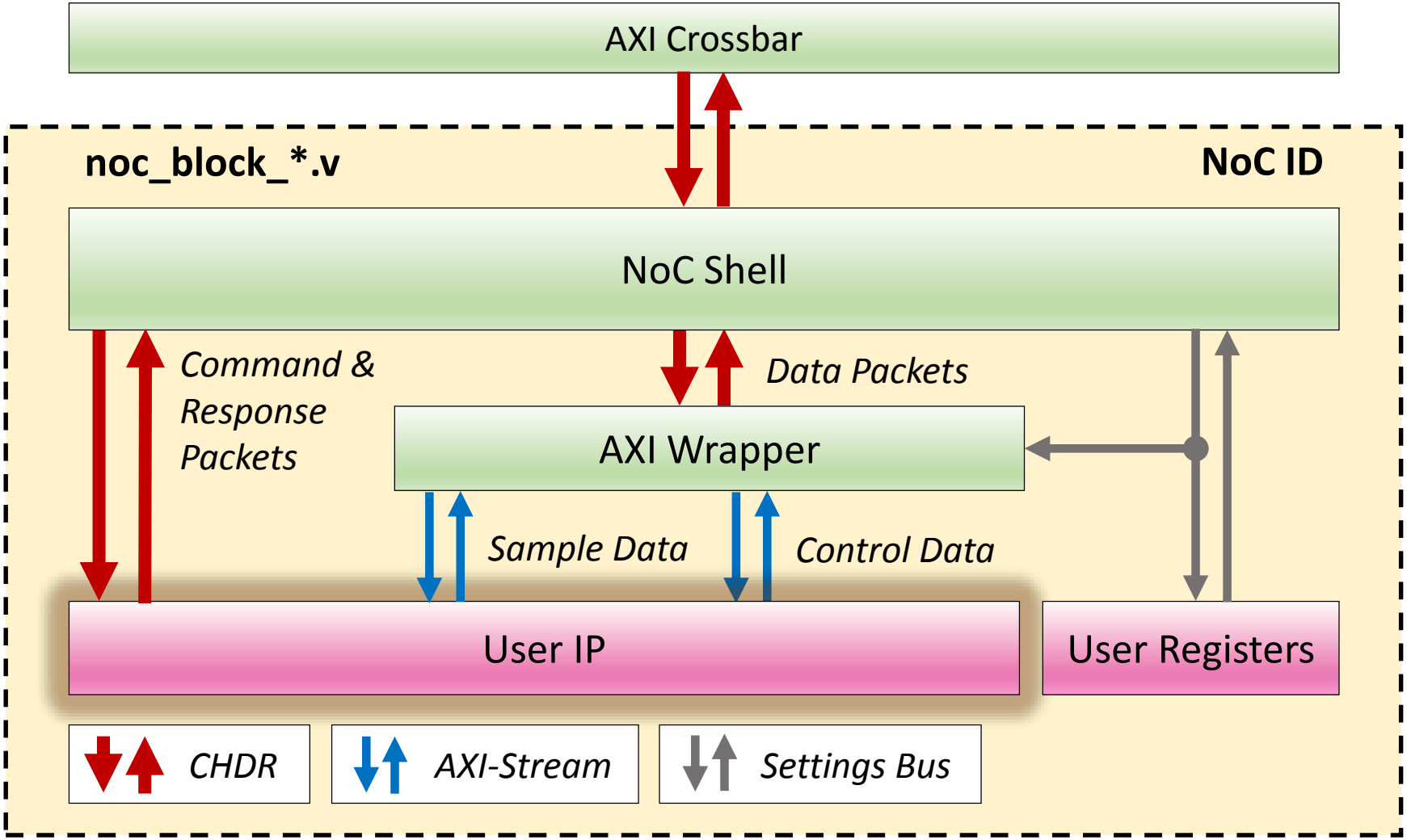
Register Space

Ettus Research™ A National Instruments Company

- Settings Bus 8-bit address = 256 regs
- NoC Shell Regs: 0 – 127
  - 0 – 63: Flow control (4 per block port, x16)
  - 127: NoC Shell readback address
- NoC Shell Readback Regs:
  - 0: NoC ID
  - 1: Window Size
- User Regs: 128 – 255
  - 128: Next destination (AXI Wrapper, set by software)
  - 129: Control data (tlast not asserted)
  - 130: Control data (tlast asserted)
  - 255: User readback address

Anatomy of a Computation Engine

Ettus Research™
A National Instruments Company



Basic Building Blocks

Ettus Research™ A National Instruments Company

- Many basic functions already written
  - All located in usrp3/lib/rfnoc
- AXI-Stream interfaces
- Notable blocks:
  - Muliplication: mult, mult_add, cmult, mult_rc
  - Add: cadd
  - Clipping / Rounding: axi_round, axi_clip
  - FIFO: axi_fifo
  - Delay: delay, delay_type2, delay_type3
  - Utility: split_stream, axi_join, packet_resizer

Existing RFNoC blocks

Ettus Research™
A National Instruments Company

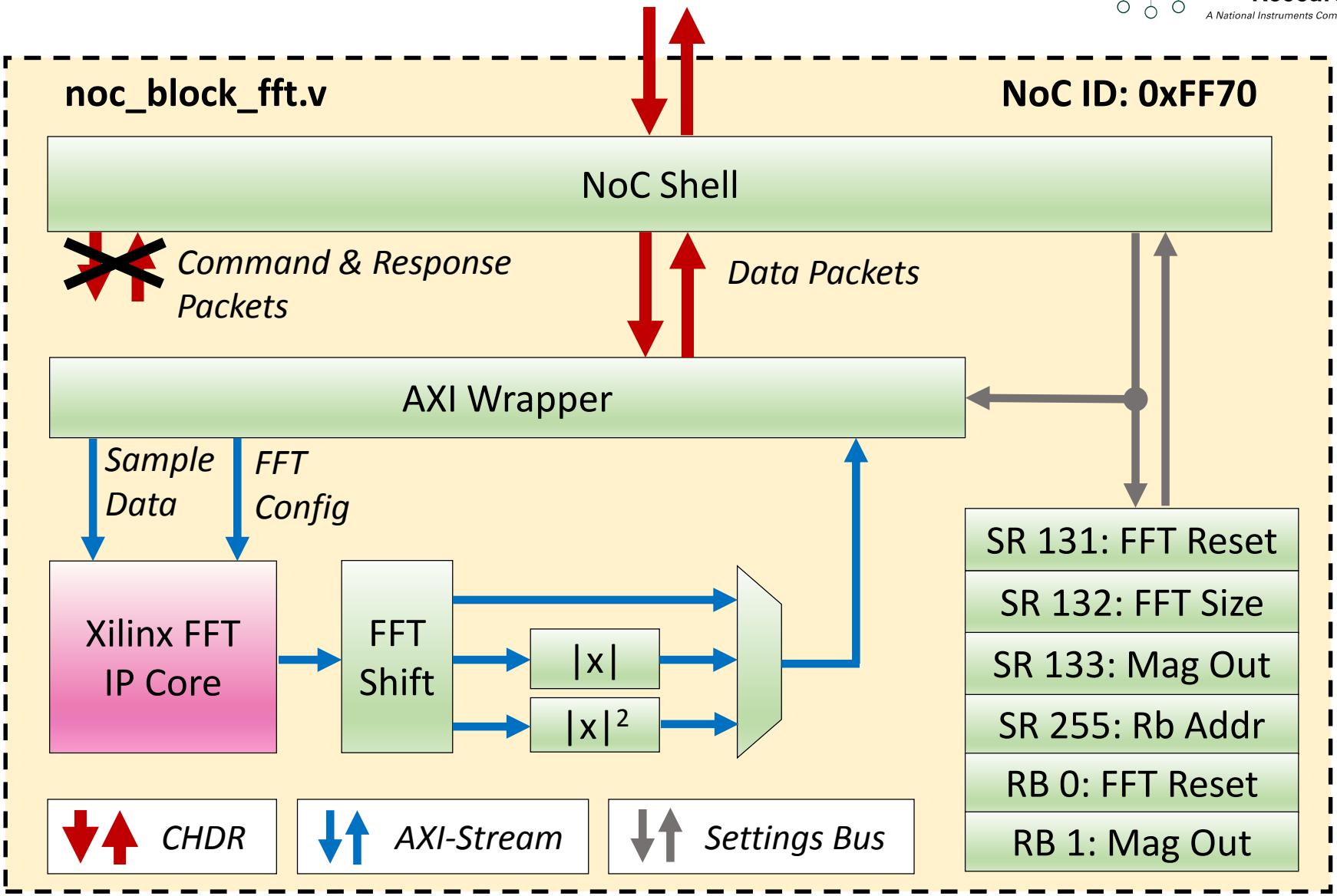
- Function:
  - FFT, Windowing, FIR, Vector Averaging
  - AddSub – Block ports
- Utility:
  - Packet resizer, Split stream, Keep one in N, FIFO
  - Null source and sink
- In development:
  - OFDM receiver & transmitter, Polyphase filter bank
  - Radio lite, separate DSP block

NoC Block FFT Walkthrough

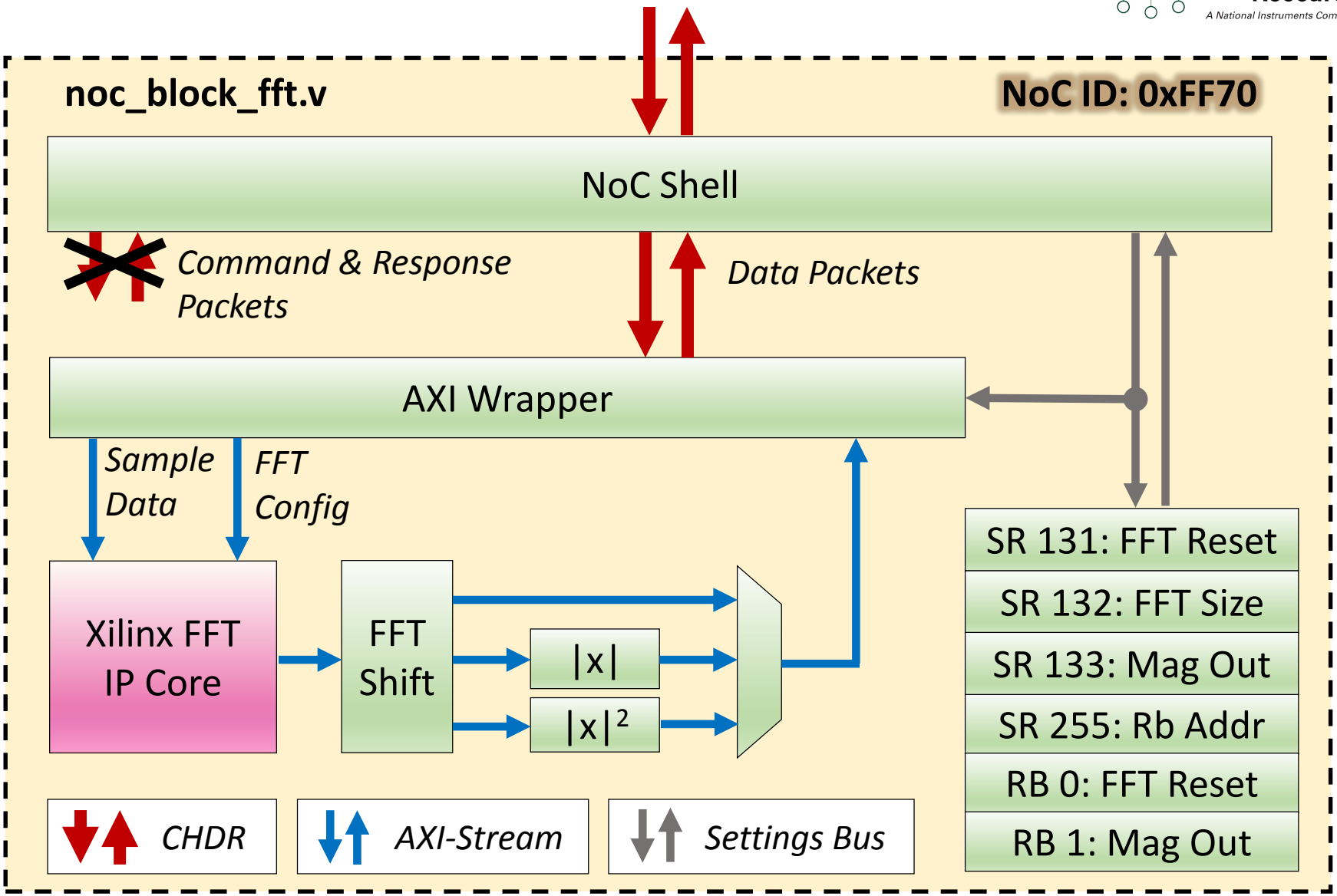
Ettus Research™ A National Instruments Company

- usrp3/lib/rfnoc/noc_block_fft.v
- Based on Xilinx AXI-Stream FFT core
- FFT size 32 – 4096
  - Up to 64K
- Configurable output
  - Complex, Magnitude, Magnitude Squared
- Built in FFT shift
- No bubble states, i.e. throughput = clock rate

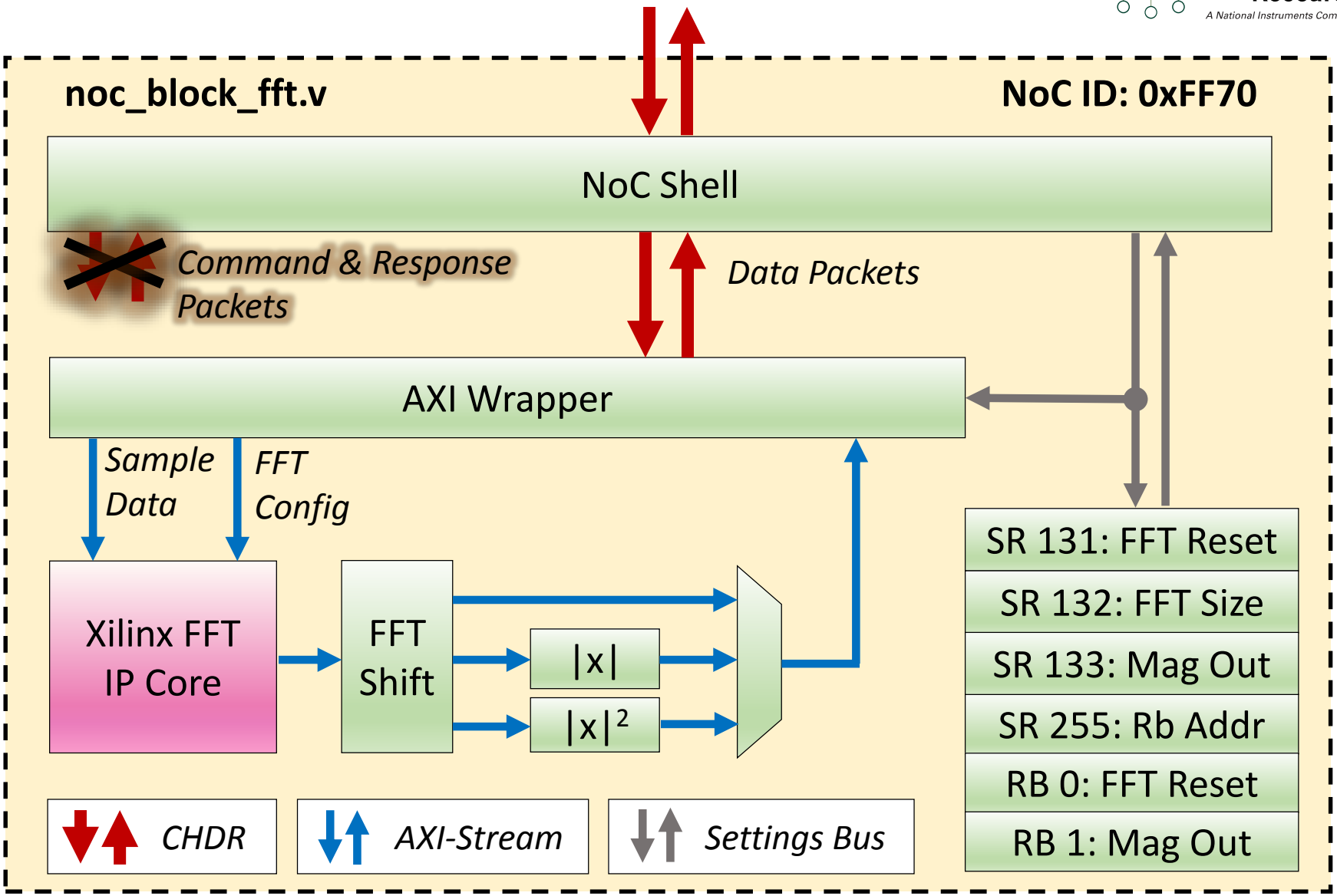
NoC Block FFT



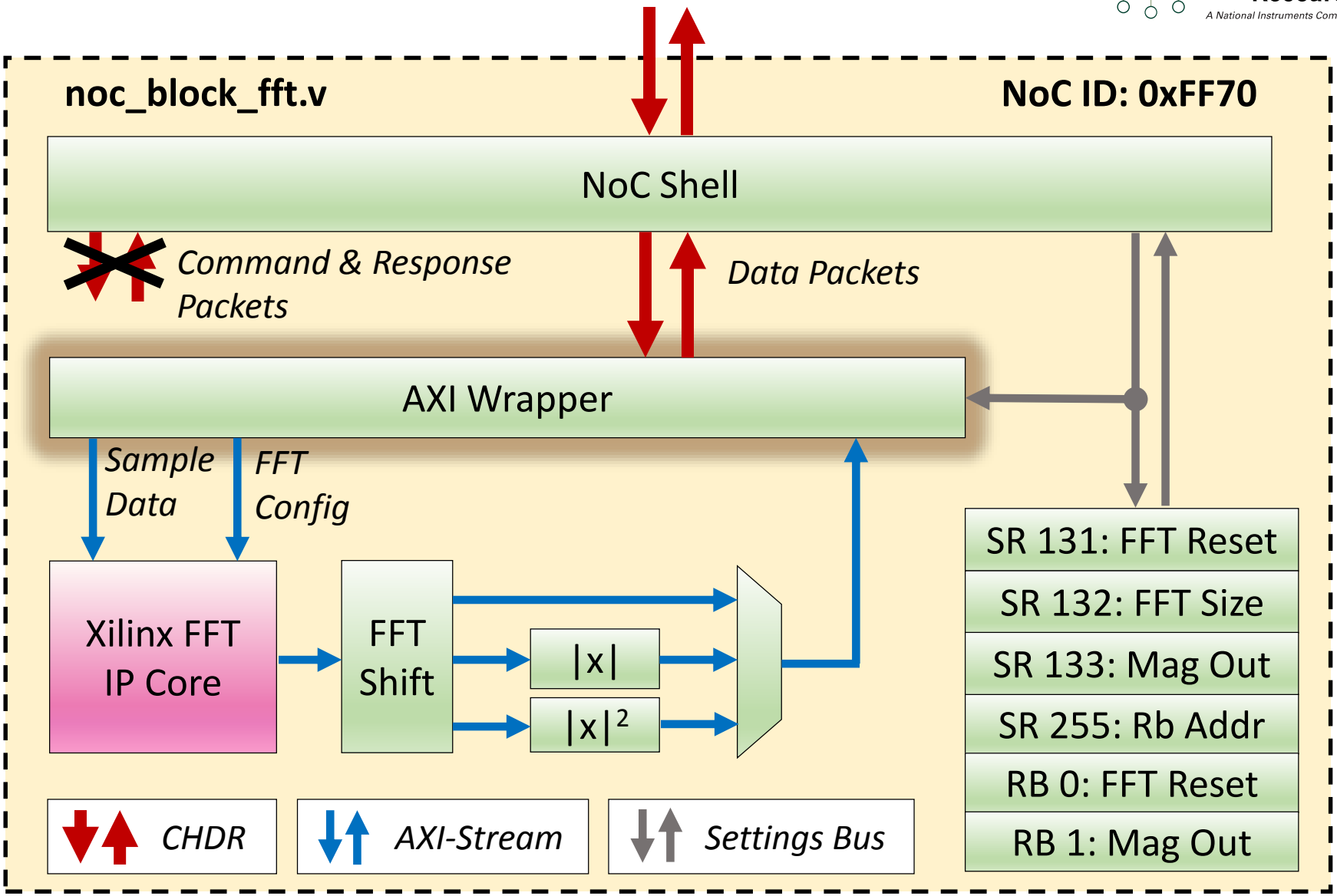
NoC Block FFT



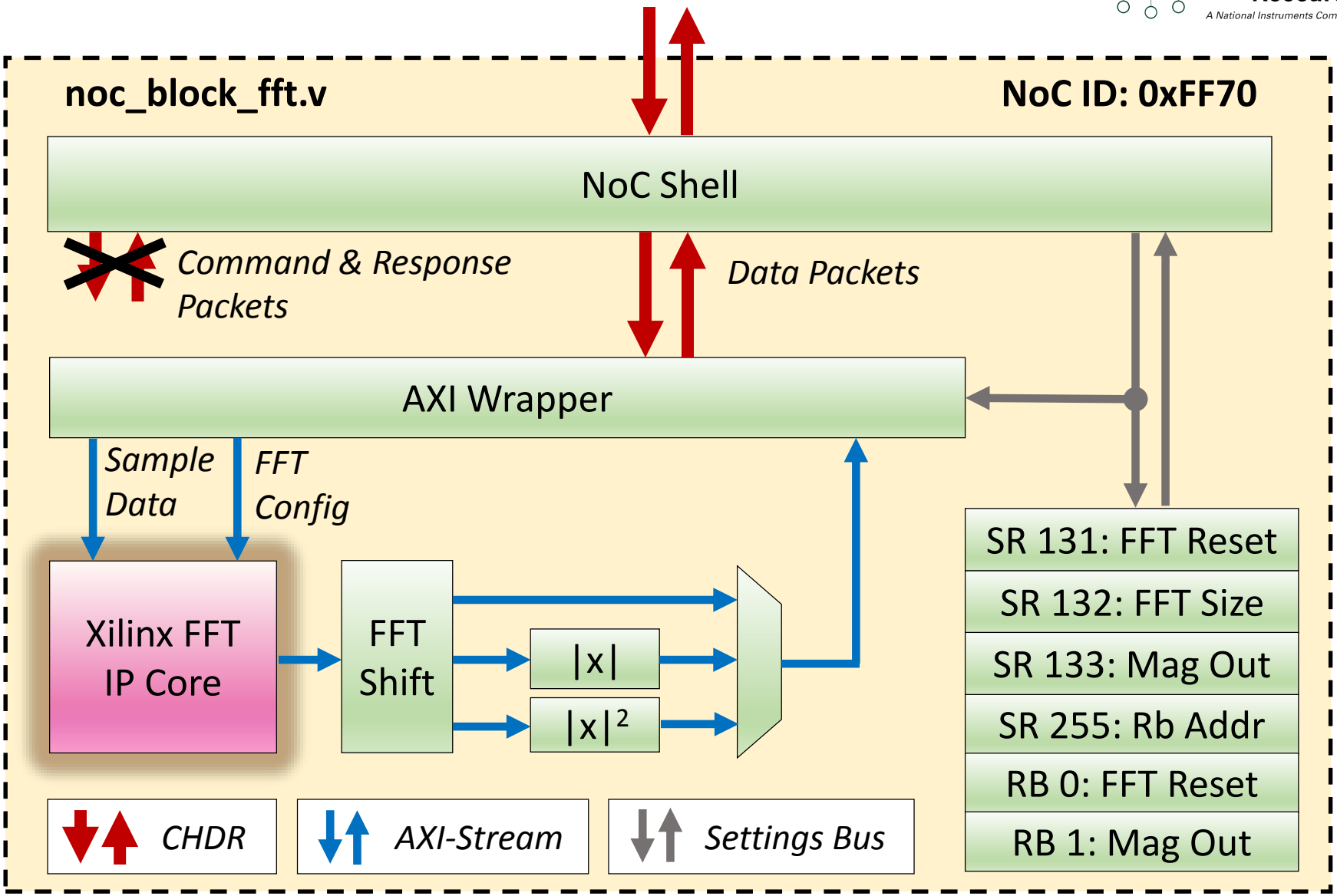
NoC Block FFT



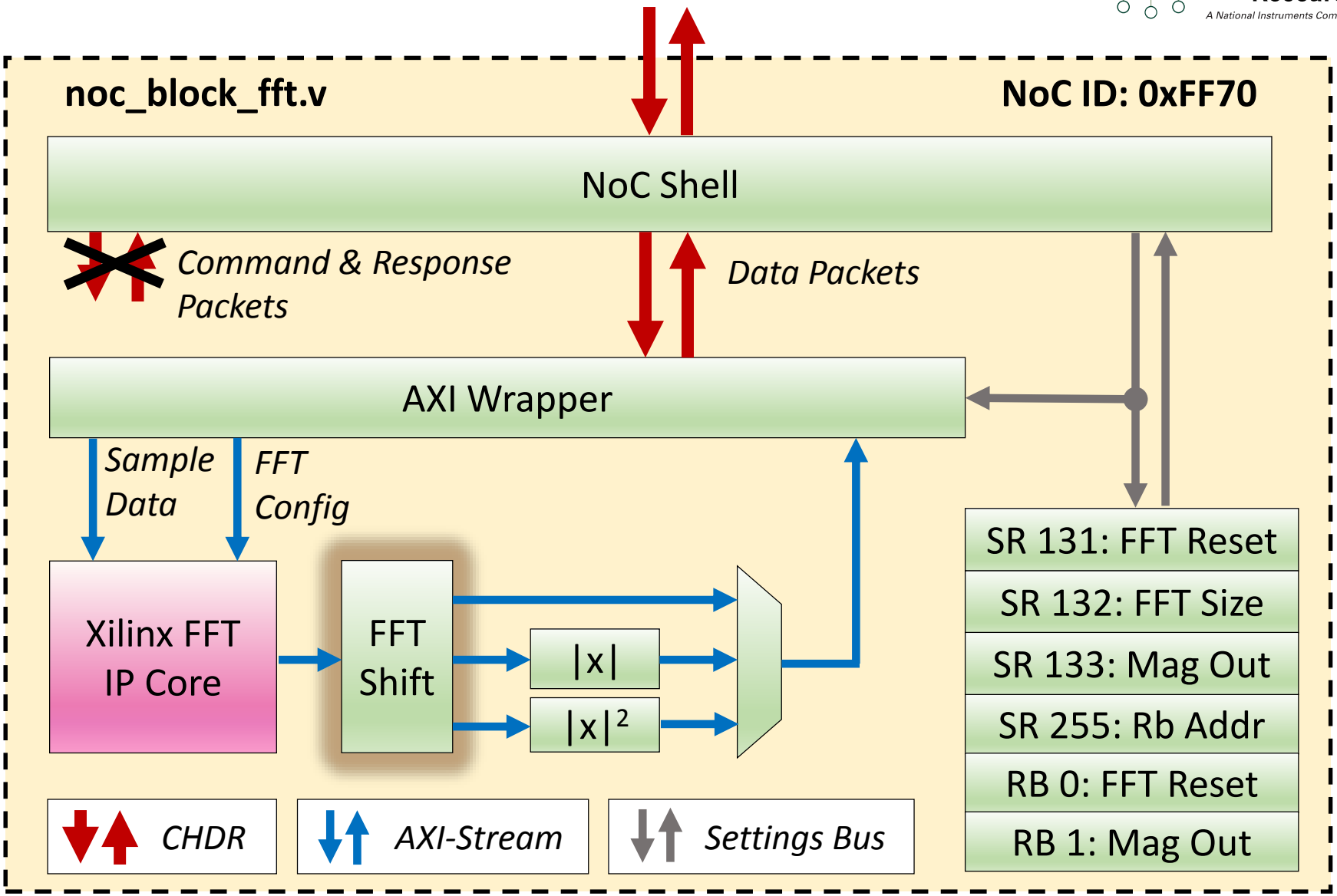
NoC Block FFT



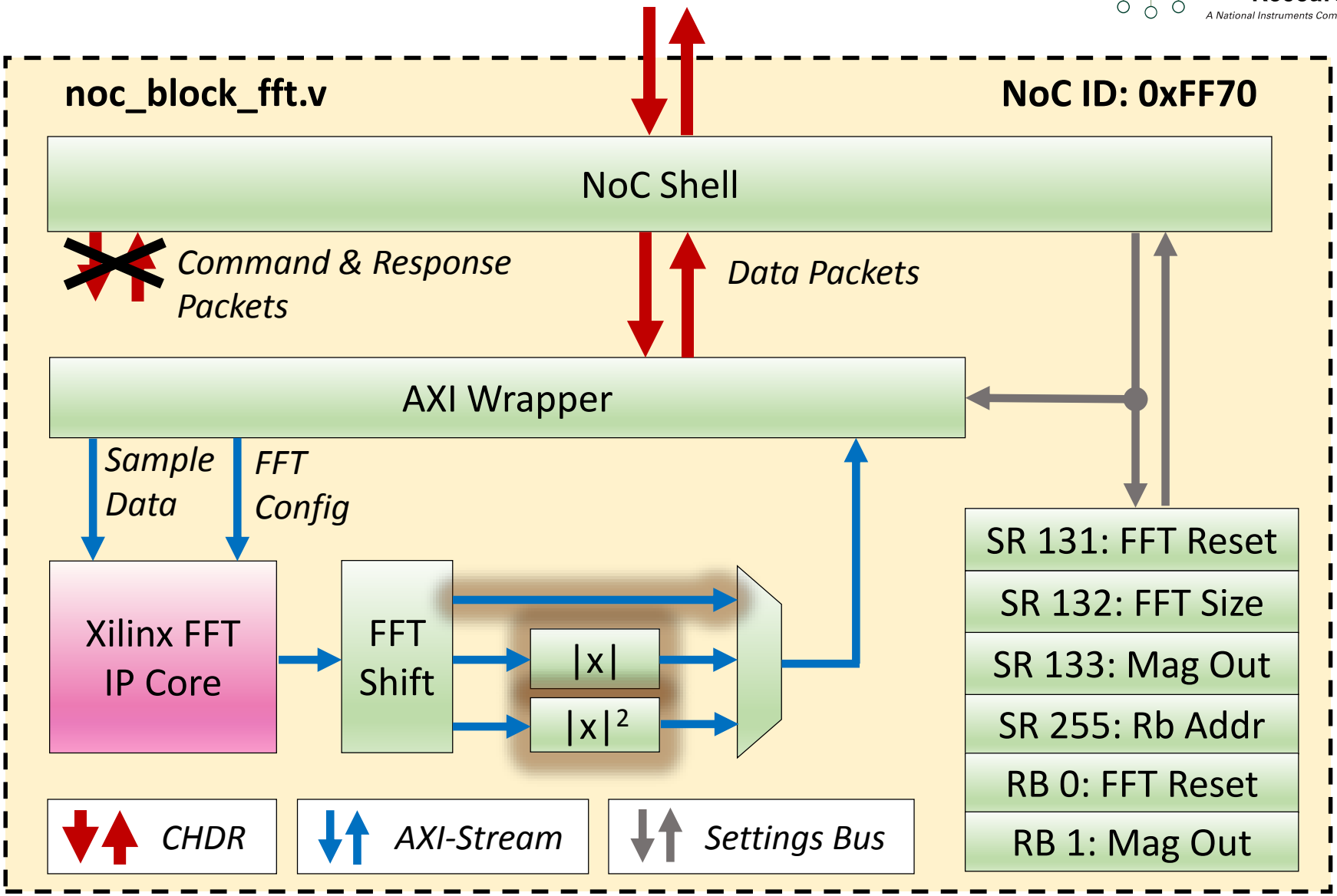
NoC Block FFT



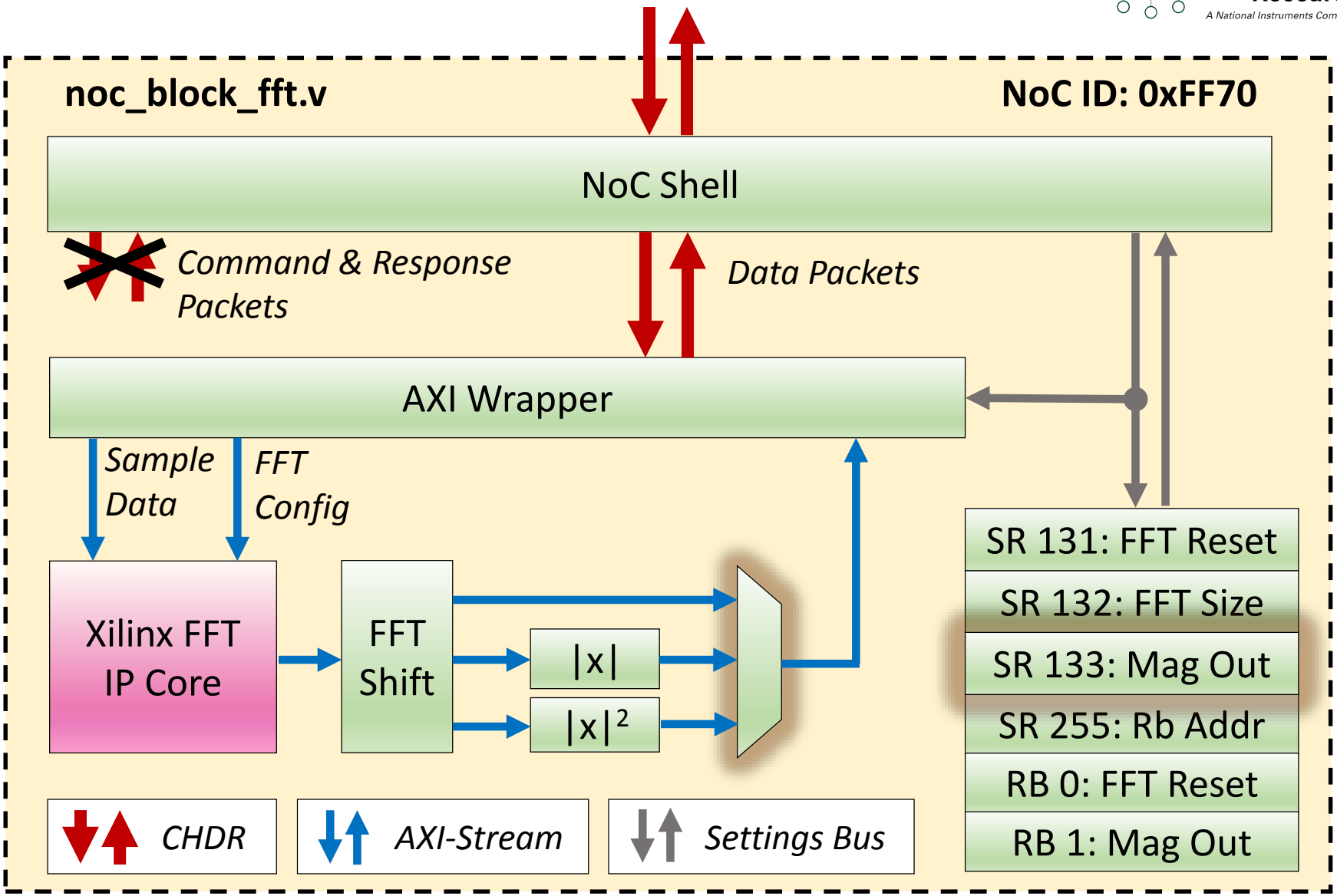
NoC Block FFT



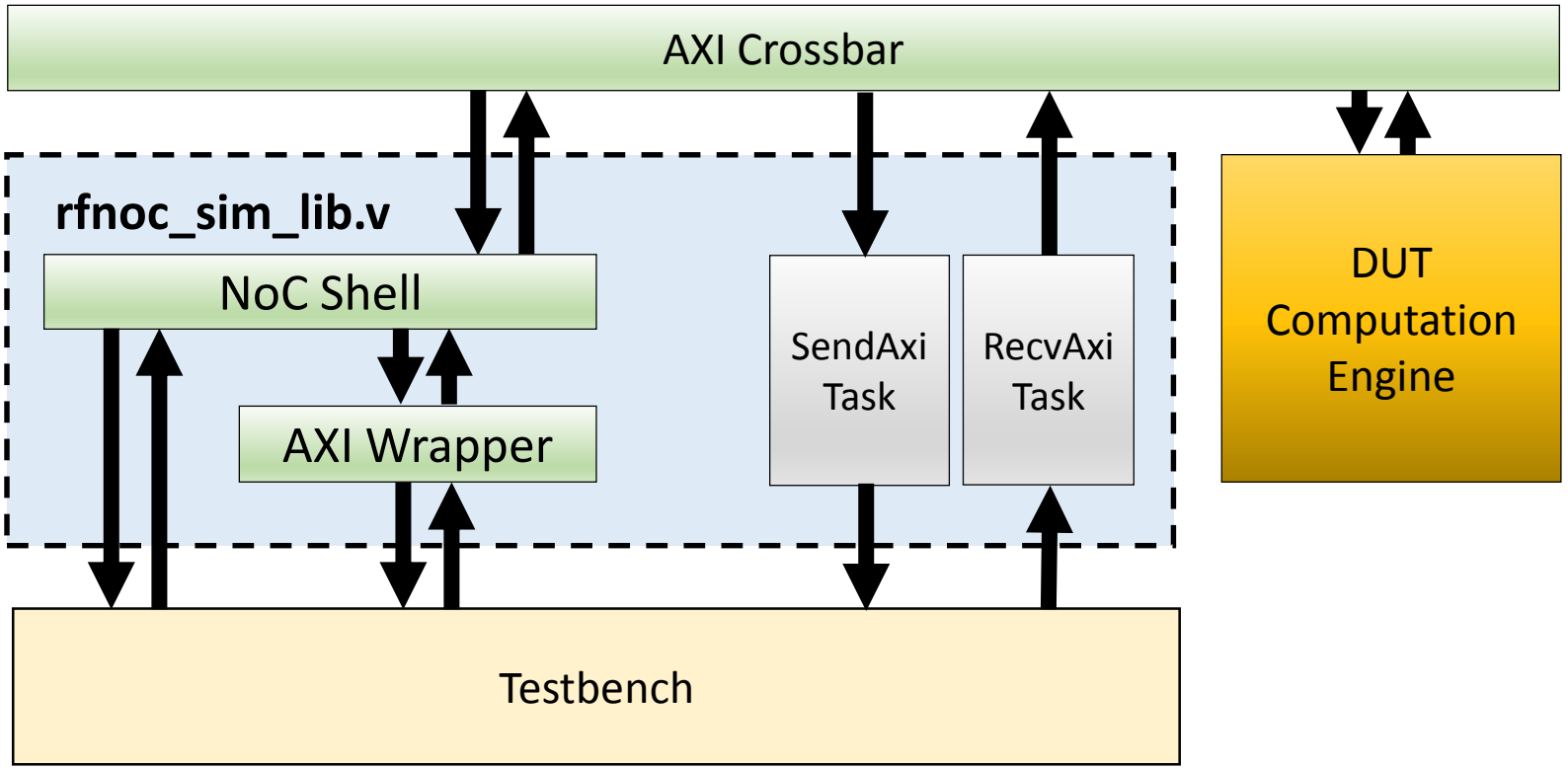
NoC Block FFT



NoC Block FFT



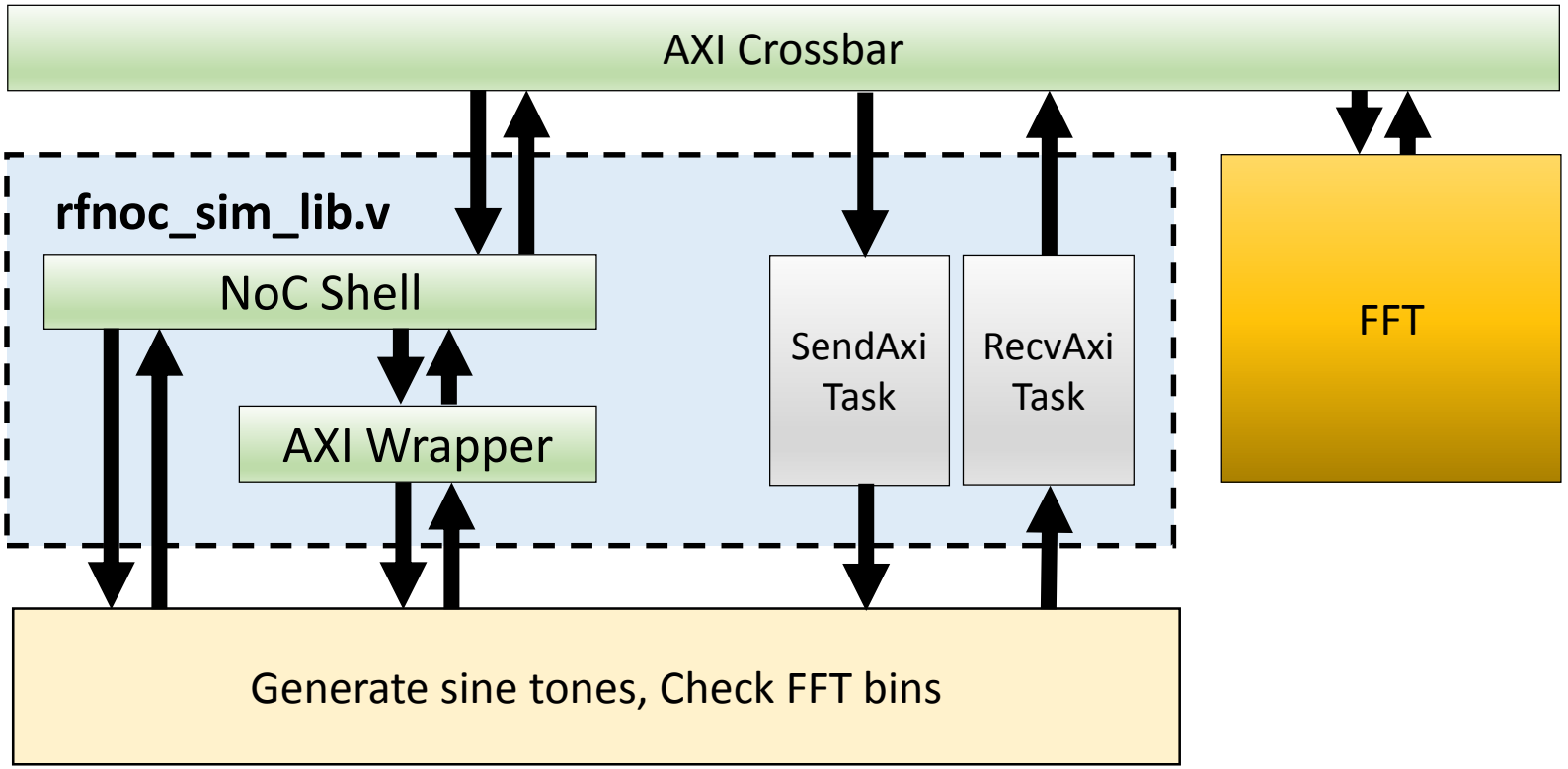
RFNoC Simlib



- Sets up crossbar and NoC Shell for testbench
- Provides useful tasks such as sending packets

RFNoC Simlib

Ettus Research™
A National Instruments Company



Adding Computation Engines

Ettus Research™ A National Instruments Company

- How do we add CEs to the build system?
  1. Place new CE in usrp3/lib/rfnoc
  2. Add block(s) to Makefile.inc
  3. Edit rfnoc_auto_ce_inst_x300.v
     - Or rfnoc_auto_ce_inst_x310.v, rfnoc_auto_ce_inst_e310.v
  4. Run make X300_HGS_RFNOC
     1. Or X310_HGS_RFNOC, E310_RFNOC

RFNoC FPGA Design Tips

Ettus Research™ A National Instruments Company

- Avoid bubble cycles to improve throughput
- When splitting an AXI stream, use FIFOs (i.e. split_stream_fifo.v)
  - Prevents back up in one stream from blocking another
  - May need to adjust FIFO depth
- When combining streams, make sure to account for different path latency
  - Inserting a FIFO on the shorter path can prevent bubble cycles (waiting for data to “catch up”)
  - Manual intervention, best done in simulation

RFNoC FPGA Design Tips

Ettus Research™ A National Instruments Company

- Be mindful of dead lock situations
  - Mostly avoided by carefully following "do not deassert tvalid without tready" rule
- Watch out for tready, tvalid combinatorial paths
  - Subtle source of timing issues (when chaining multiple blocks)
  - Consider inserting register: axi_fifo.v SIZE=0
- Ignore output without a tvalid (when debugging)
- Crossbar statistics tool: firmware/usrp3/x300/x300_debug.py

Appendix

Flow Control

- Fine and course grained flow control
- Fine grained: AXI-Stream
- Course grained: Credit based flow control
  - Every endpoint (or consumer) has a receive window
  - Every source (or producer) knows window size
  - Producers send packets until consumer window is full
  - Consumer notifies (acks) producer as window empties
    - i.e. gives credits back to producer
  - Some similarities to TCP flow control

Flow Control Cont.

Ettus Research™ A National Instruments Company

- A few additional design rules
  1. One consumer per producer
     - Otherwise very difficult to keep track of window
     - Use *block ports* to allow a CE to receive multiple streams
  2. Producers must buffer entire packet before releasing
     - Prevents deadlock in crossbar
     - Prevents slow packets from causing congestion
  3. All routing based on first line of CHDR header
     - Higher performance
  4. Samples / data always dropped at producer
     - Deterministic -- ensures no lost packets in the middle
     - Single point to restart stream

AXI Stream

Ettus Research™
A National Instruments Company

- Part of ARM AMBA standard
- Simple handshake protocol:
  - Upstream block asserts **tvalid**
  - Downstream block asserts **tready**
  - Data is consumed when tvalid & tready == 1
  - **tlast** used to delimit packets
- **Once tvalid is asserted, it cannot be deasserted without at least one tready cycle**
- Why is AXI Stream so useful?
  - No need for complicated strobes – data flows through

Settings Bus

Ettus Research™ A National Instruments Company

- Common bus in USRP FPGA designs
- Implements control & status registers
- set_addr, set_data, set_stb, rb_data
  - Why no rb_addr? It is a control register
- Control packet payload sets addr & data
  - [63:0] = {24'd0, set_addr[7:0], set_data[31:0]}
- Response packet payload has readback data
  - All control packets receive a response packet
  - Usually write readback address control register to read a status register

Block Ports

Ettus Research™ A National Instruments Company

- CEs support up to 16 block ports
- Allows CEs to accept multiple streams
  - Time share a function (i.e. FFT)
  - Use data from multiple sources (i.e. addsub block)
- Each block port has dedicated NoC shell output
  - Sample data packets only (str_src_t*, str_sink_t*)
    - Bit width widened to 64*N where N = number of block ports
    - Individual sample data FIFOs for each block port
  - Command packets share interface (cmdout_t*, ackin_t*)
  - Resource utilization can increase quickly
- Shared input bus
  - Does not allow receiving two packets simultaneously

Generating IP with Vivado

Ettus Research™
A National Instruments Company

Demo

Adding IP cores

Ettus Research™ A National Instruments Company

- Find *.xci file in build directory
  - Example: build-E310/project_1/project_1.srcs/sources_1/ip/xfft_0/xfft_0.xci
- Make IP specific directory in *usrp3/lib/ip* and add *.xci file
  - Example: usrp3/lib/ip/xfft_0/xfft_0.xci
- Edit usrp3/lib/ip/Makefile.inc
- Add IP specific Makefile.inc in IP subdirectory
  - Use existing Makefile.inc as template
  - Example: usrp3/lib/ip/xfft_0/Makefile.inc
- Note: We are likely to change this approach to make adding “OOT” RFNoC modules easier

Simulating our Custom Block

Ettus Research™ A National Instruments Company

- Modelsim simulation infrastructure
  - Tested with ModelSim SE on Linux
- Makefile based – easy to add new testbenches
- Example testbenches in usrp3/lib/rfnoc
  - rfnoc_fir_filter_tb
  - rfnoc_fft_vector_iir_tb
- Note: Big changes coming soon
  - More generic infrastructure
  - Support for Vivado simulator

In system debugging

Ettus Research™ A National Instruments Company

- Vivado ILA: Integrated Logic Analyzer
  - Real time logic analyzer inserted into design
  - Integrated into Vivado GUI
  - Uses logic resources (Slices, BRAM)
- Multiple ways to instantiate, see Xilinx docs
- Easiest to mark nets & registers for debug in HDL
  - Verilog:

    (* keep = “true”, dont_touch = “true”, mark_debug = “true” *) wire a;
    (* keep = “true”, dont_touch = “true”, mark_debug = “true” *) reg b;
  - VHDL:

    ATTRIBUTE MARK_DEBUG : string;
    ATTRIBUTE MARK_DEBUG of a : SIGNAL IS "TRUE";
    …