AXI BUS TRAFFIC SNIFFER

Duo Wang, Phat N Le, Yuxuan Nie

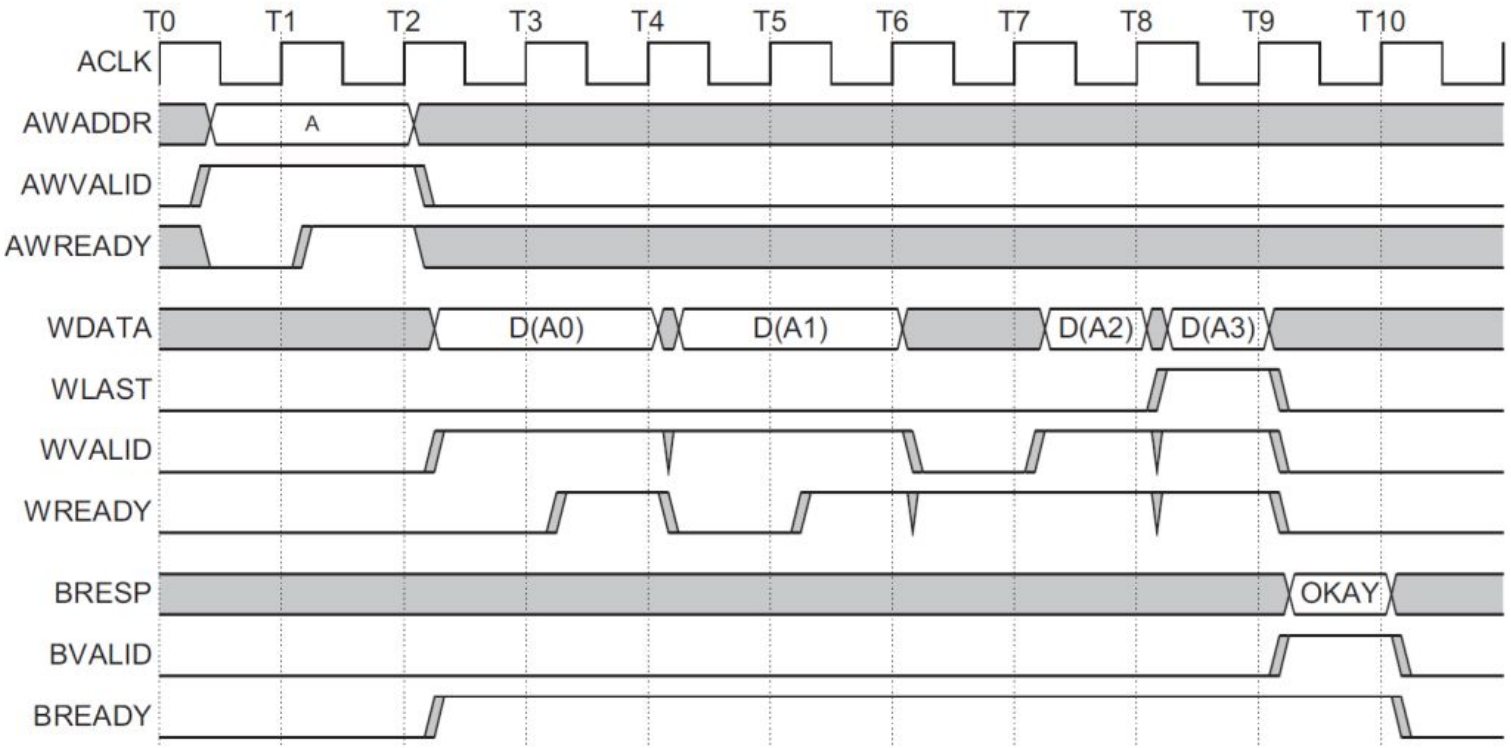
Motivation

- What happens on the AXI bus?
- For Labs 1/2 can we see what happens during a CDMA transfer for debugging?
- Official debugging tools cost a lot

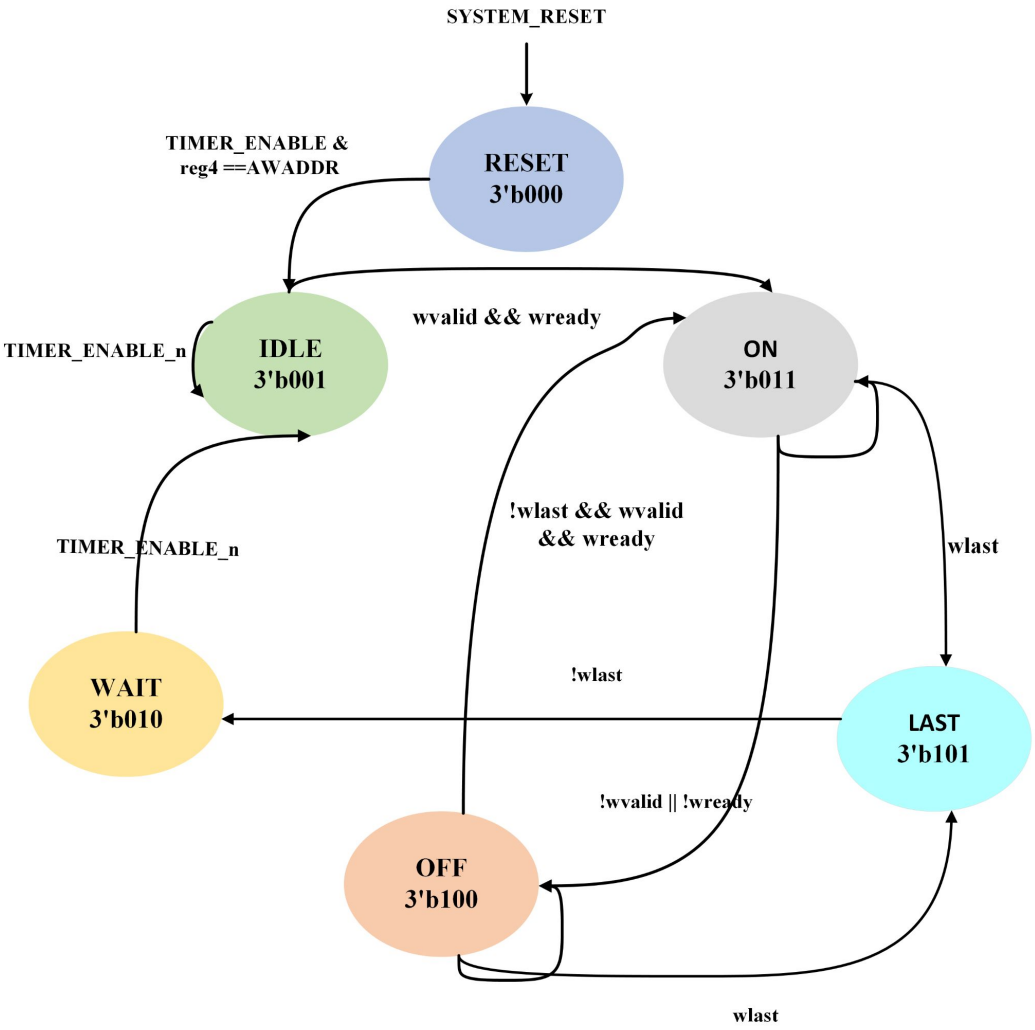
AXI Terminology

- Transfer
  - Packet of data
  - Size of data width
- Transaction
  - Made up of multiple transfers
  - Starts with an address then increments for each transfer
- AXI Burst Length
  - Number of transfers per transaction: burst length + 1
  - Vivado Burst Size
- AXI Burst Size
  - Number of bytes per transfer: 2^(burst size)
  - Vivado Data Width

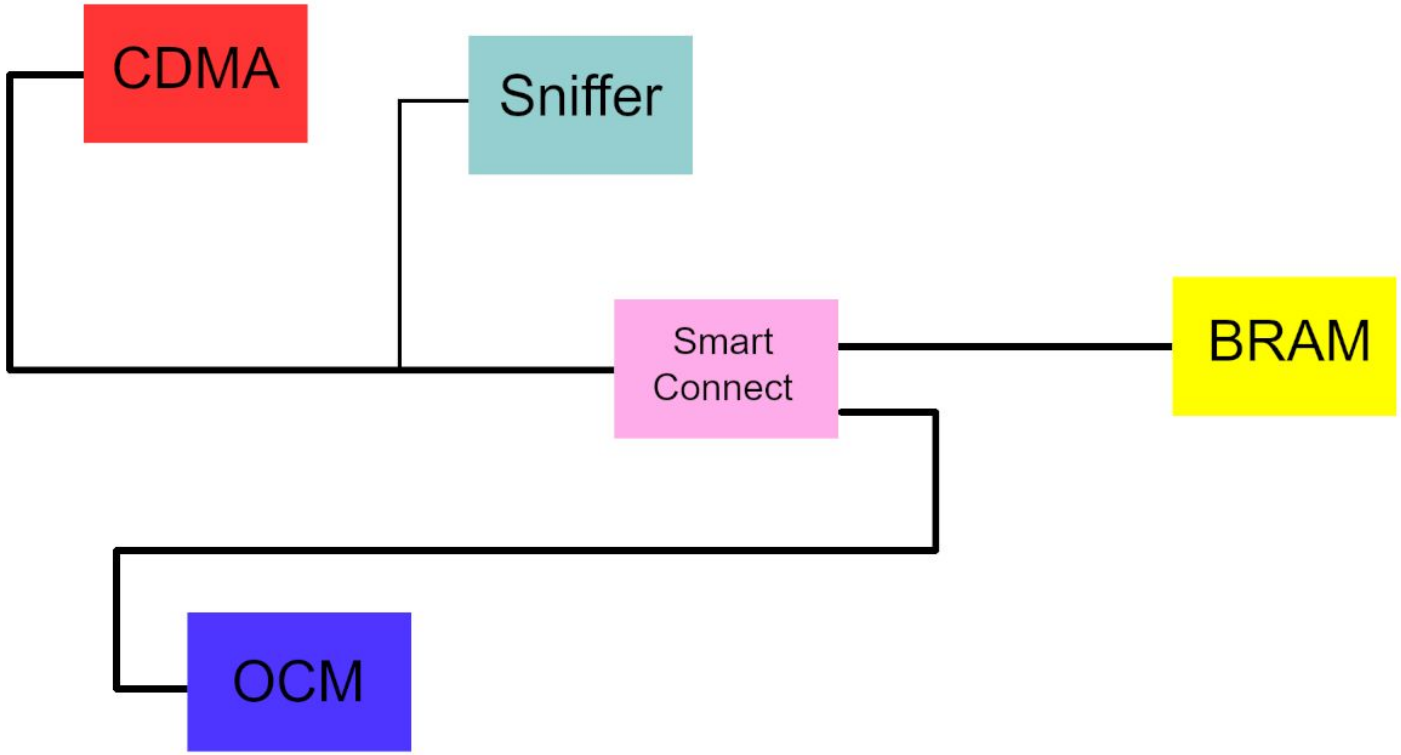
Write Transaction: AXI Handshake



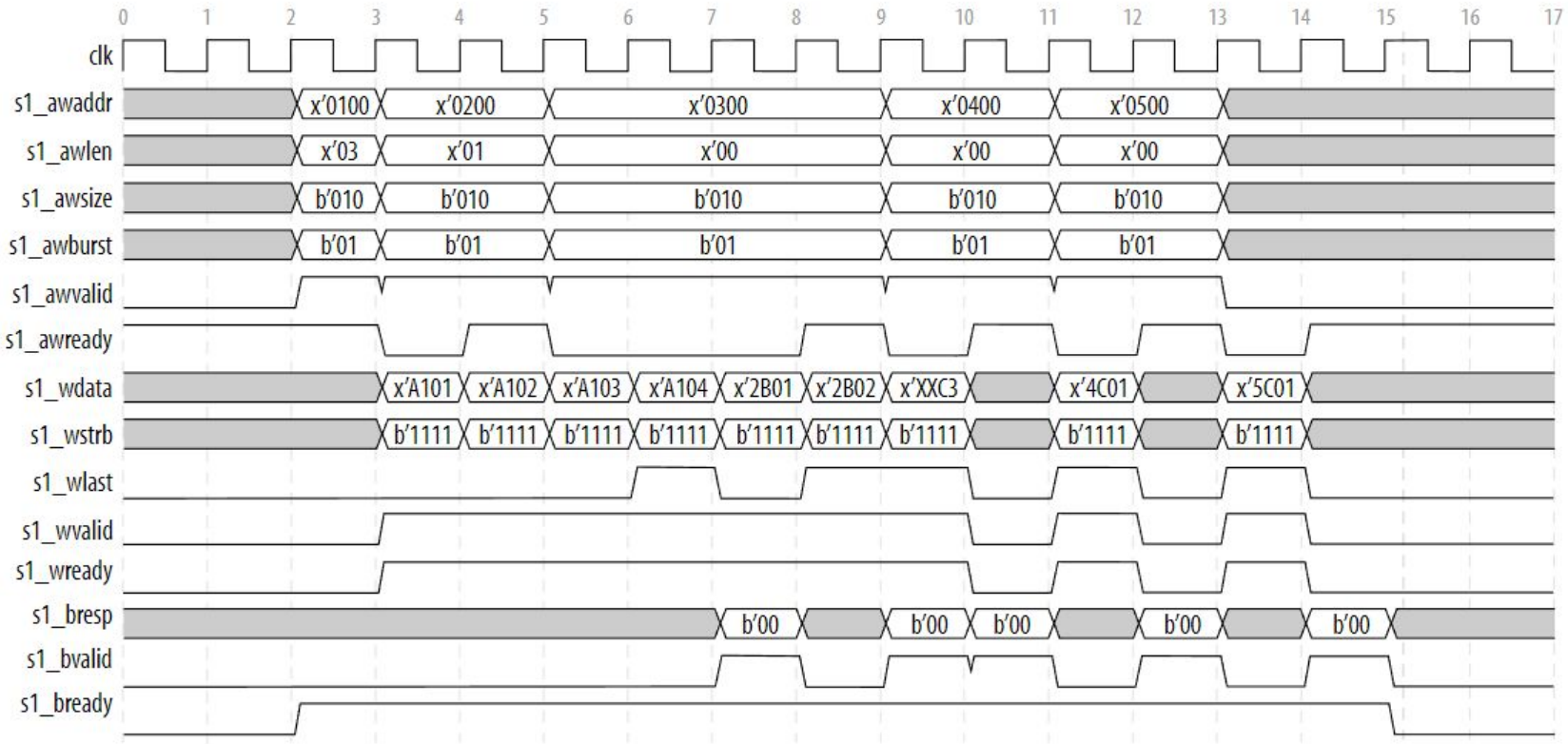
Initial State Machine



Our Implementation



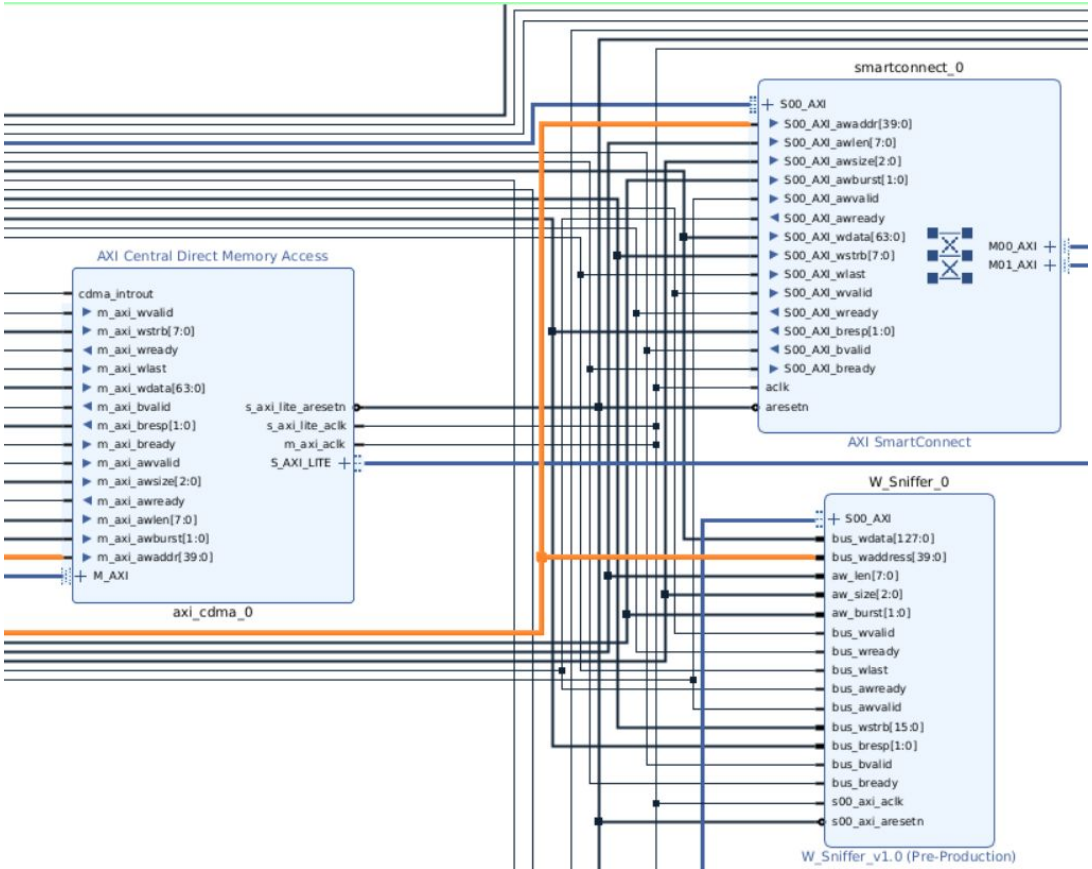
What Actually Happens On The Bus



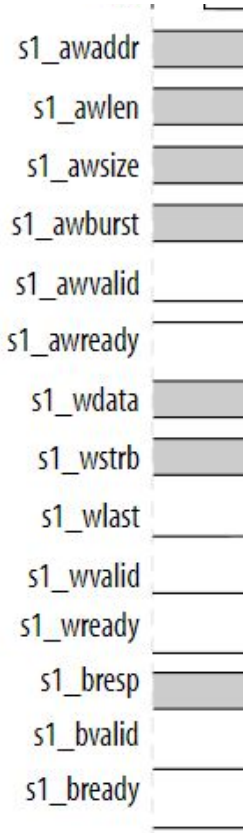
Current Capabilities of AXI Sniffer

- Will sniff for transactions for a given address range
- The IP supports data widths of up to 128 bits
- Records the total number of transactions, transfers, and clock cycles
  - Currently capped at 64 transactions and 32 transfers per transaction
- Records every AXI write signal from the first transaction in range to the last transaction in range
- With the data it records:
  - We can generate a .json file and create a timing diagram through Wavedrom https://wavedrom.com/tutorial.html
  - We can produce a real-time graph of the transactions

Our Implementation



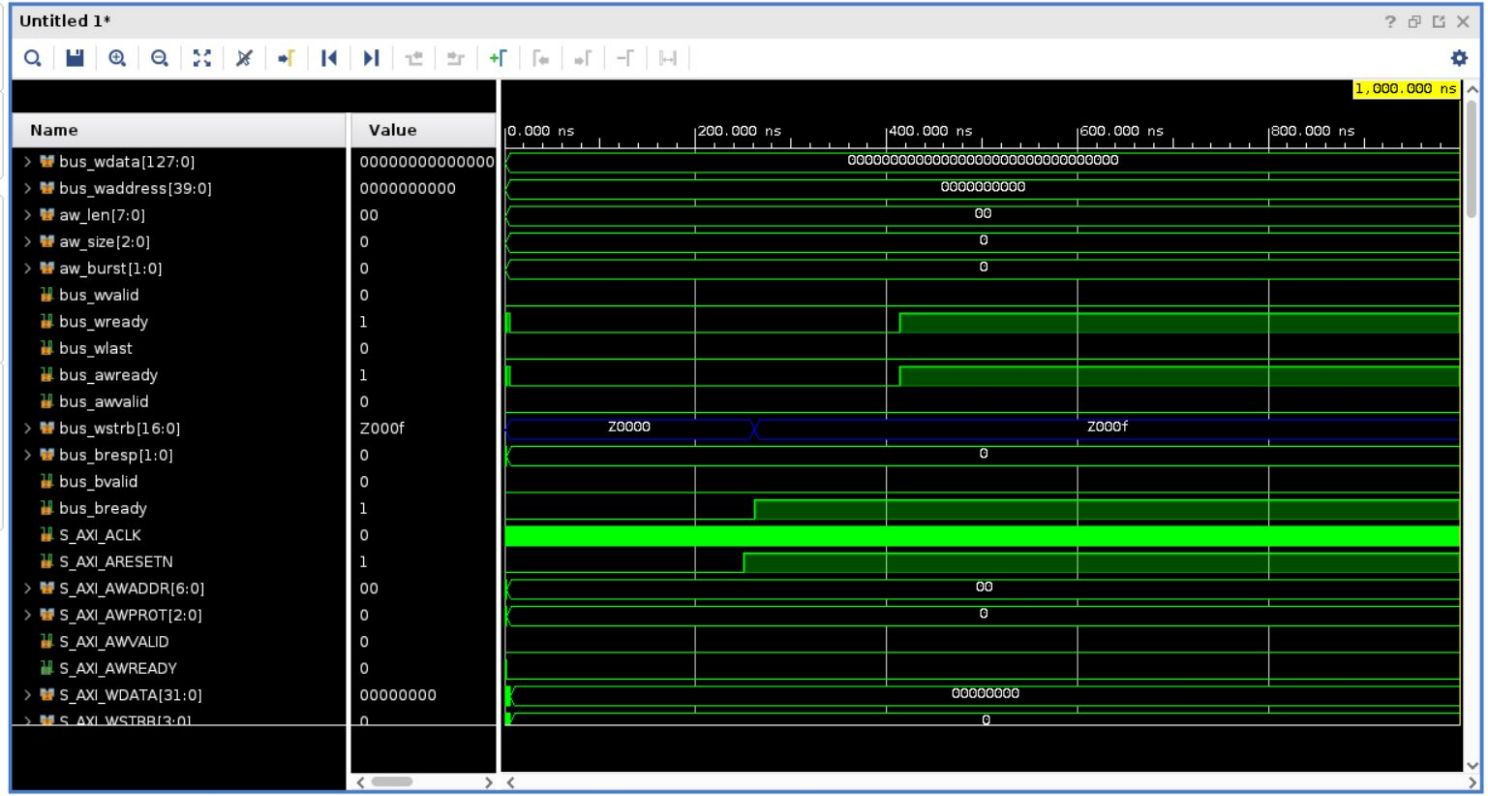
Our Implementation



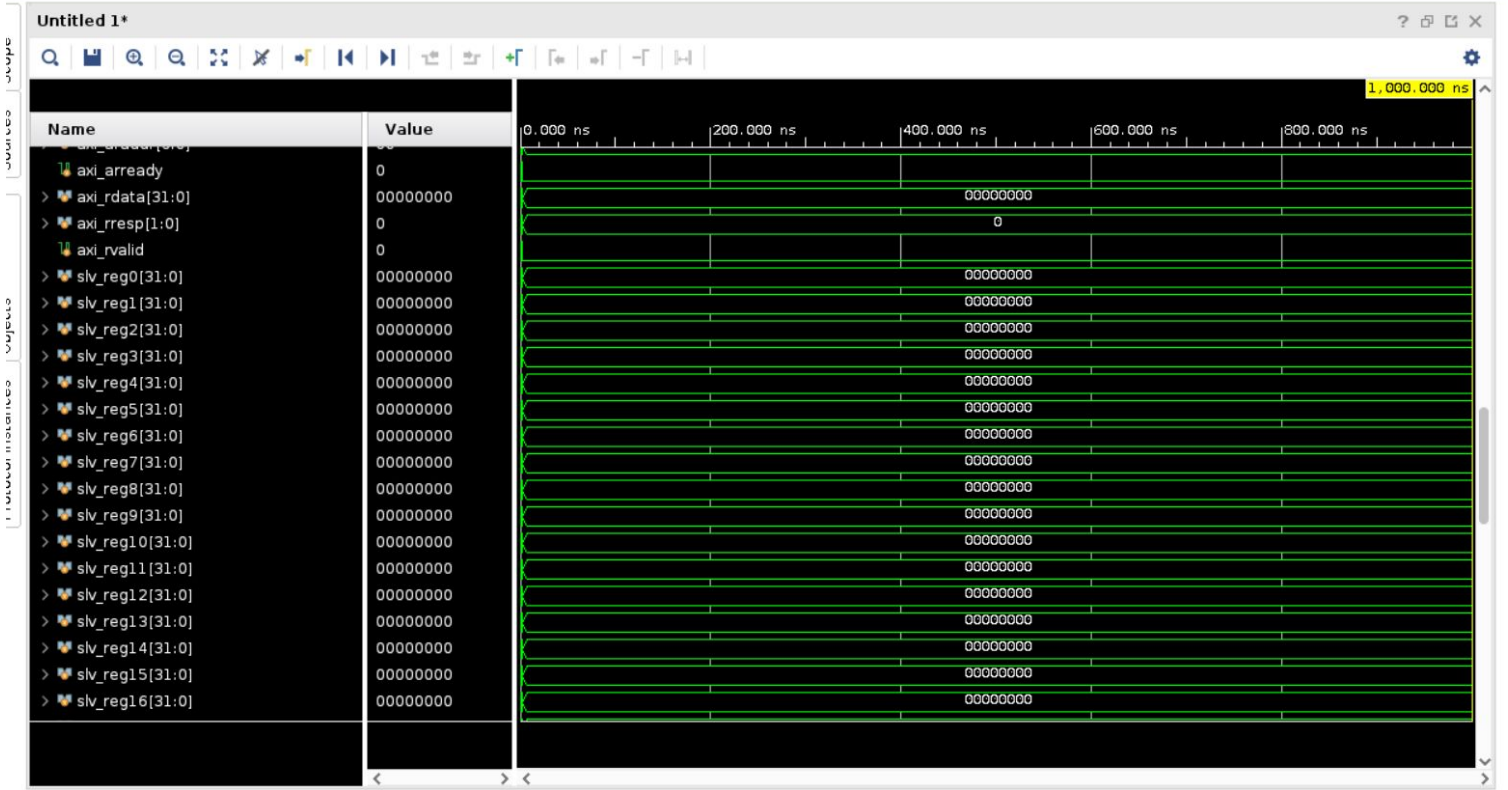

```
case (slv_reg16)
    2'h0: chunk_wdata <= all_wdata[slv_reg7][31:0];
    2'h1: chunk_wdata <= all_wdata[slv_reg7][63:32];
    2'h2: chunk_wdata <= all_wdata[slv_reg7][95:64];
    2'h3: chunk_wdata <= all_wdata[slv_reg7][127:96];
    default: chunk_wdata <= 32'hDAAAAAAD;
endcase

    // Address decoding for reading registers
    case ( axi_araddr[ADDR_LSB+OPT_MEM_ADDR_BITS:ADDR_LSB] )
      5'h00   : reg_data_out <= wsniff_start;
      5'h01   : reg_data_out <= wsniff_end;
      5'h02   : reg_data_out <= {28'h0FACADE, 3'h0, timer_enable};
      5'h03   : reg_data_out <= wtransaction_count;
      5'h04   : reg_data_out <= bresp_count;
      5'h05   : reg_data_out <= wtransfer_count;
      5'h06   : reg_data_out <= slv_reg6;    //user set get transaction
      5'h07   : reg_data_out <= slv_reg7;    //user set get transfer
      5'h08   : reg_data_out <= slv_reg8;    //user set get clock cycle
      5'h09   : reg_data_out <= count;
      5'h0A   : reg_data_out <= all_awaddress[slv_reg6];
      5'h0B   : reg_data_out <= all_burst_info[slv_reg6];
      5'h0C   : reg_data_out <= all_bresp[slv_reg6];
      5'h0D   : reg_data_out <= chunk_wdata;
      5'h0E   : reg_data_out <= all_wstrb[slv_reg7];
      5'h0F   : reg_data_out <= {24'hFACADE, signals[slv_reg8]};
      5'h10   : reg_data_out <= slv_reg16;   //user set get wdata chunk
      5'h11   : reg_data_out <= slv_reg17;
```

```
parameter AWVALID = 0,
parameter AWREADY = 1,
parameter WLAST = 2,
parameter WVALID = 3,
parameter WREADY = 4,
parameter BVALID = 5,
parameter BREADY = 6
```

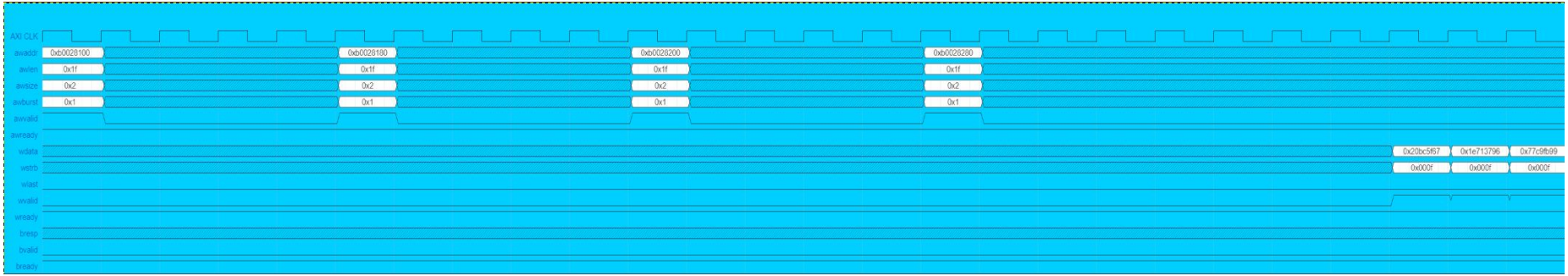
Testing and Debugging

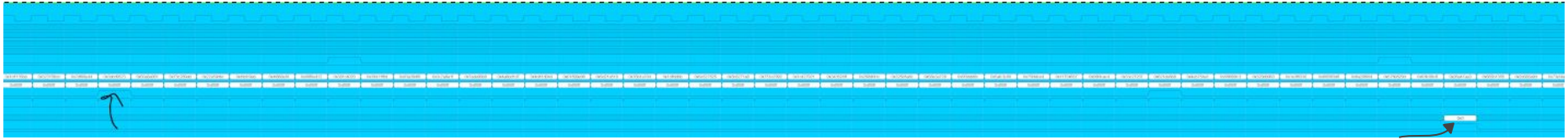


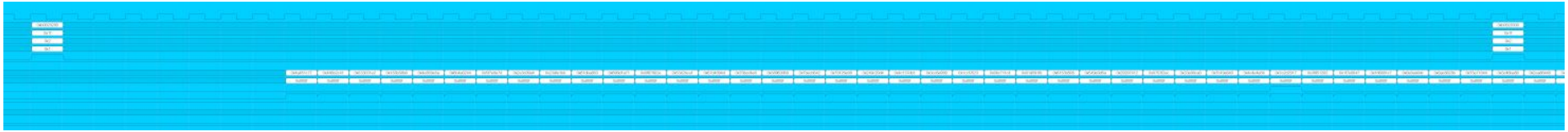
Testing and Debugging



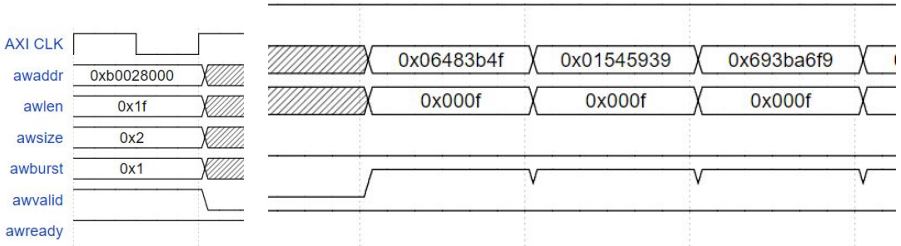
Results: Observed Behavior

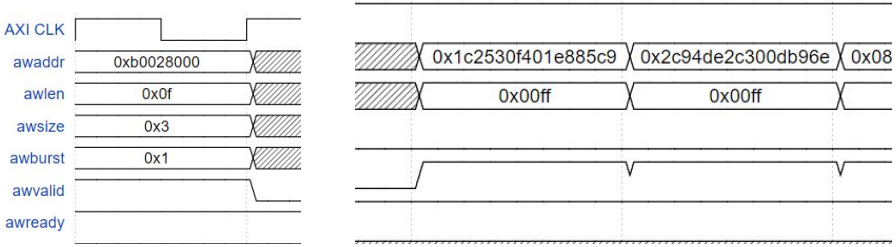


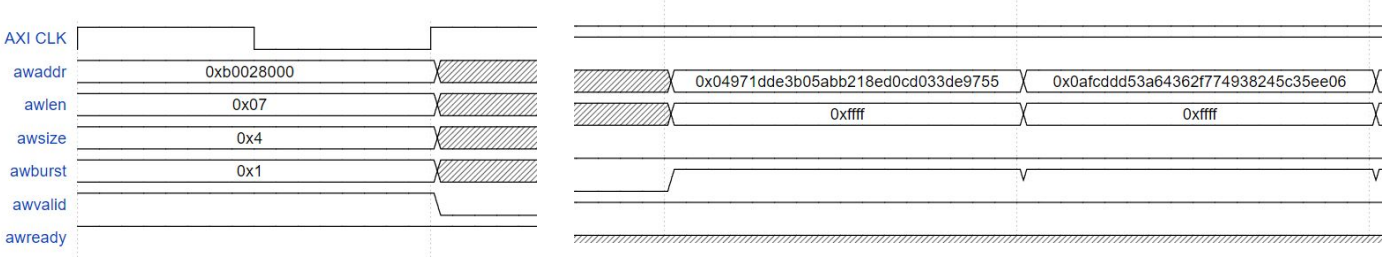




Results: Different Data Widths







Demo (real time graphing generation)

Future Work

- Refinement
  - Verify sniffer functionality on all possible scenarios
  - Optimize hardware resource usage
  - Refine by comparing different bus width and data width
  - Graphical UX interface, more options (like clock changes or data size) to fully optimize the debugging experience
  - Faster speed with multi-processing
- Expansion
  - Sniff read signals on the AXI bus
  - Sniff other AXI interfaces
  - Create software that reports human-readable performance metrics without the timing diagram

Questions?

References

- TA's project (Jatin Khare and Abhijjith Venkkateshraj)
- https://www.xilinx.com/content/dam/xilinx/support/documents/sw_manuals/xilinx2021_2/ug974-vivado-ultrascale-libraries.pdf
- https://www.intel.com/content/www/us/en/docs/programmable/683130/22-2/axi-interface-timing-diagram.html