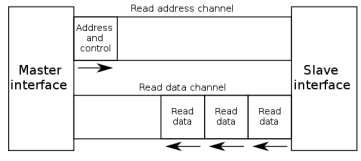
Goal: Conversion of Dan Gisselquist ZipCPU axil2axi.v to TL-Verilog.

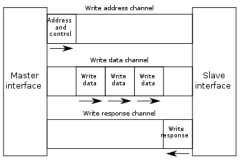
AXI offers a wide spectrum of features, including:

- separate address/control and data phases
- support for unaligned data accesses
- burst-based transfers, with a single transmission of the starting address
- separate and independent read and write channels
- support for outstanding transactions
- support for out-of-order transaction completion for transactions having different thread IDs on the same master port. (Transactions on the same master port that have the same thread ID must be completed in order. Additionally, different master ports may be completed out of order with respect to each other.)
- support for atomic operations.

"https://en.m.wikipedia.org/wiki/Advanced_eXtensible_Interface"



AXI Read Address and Read Data channels.

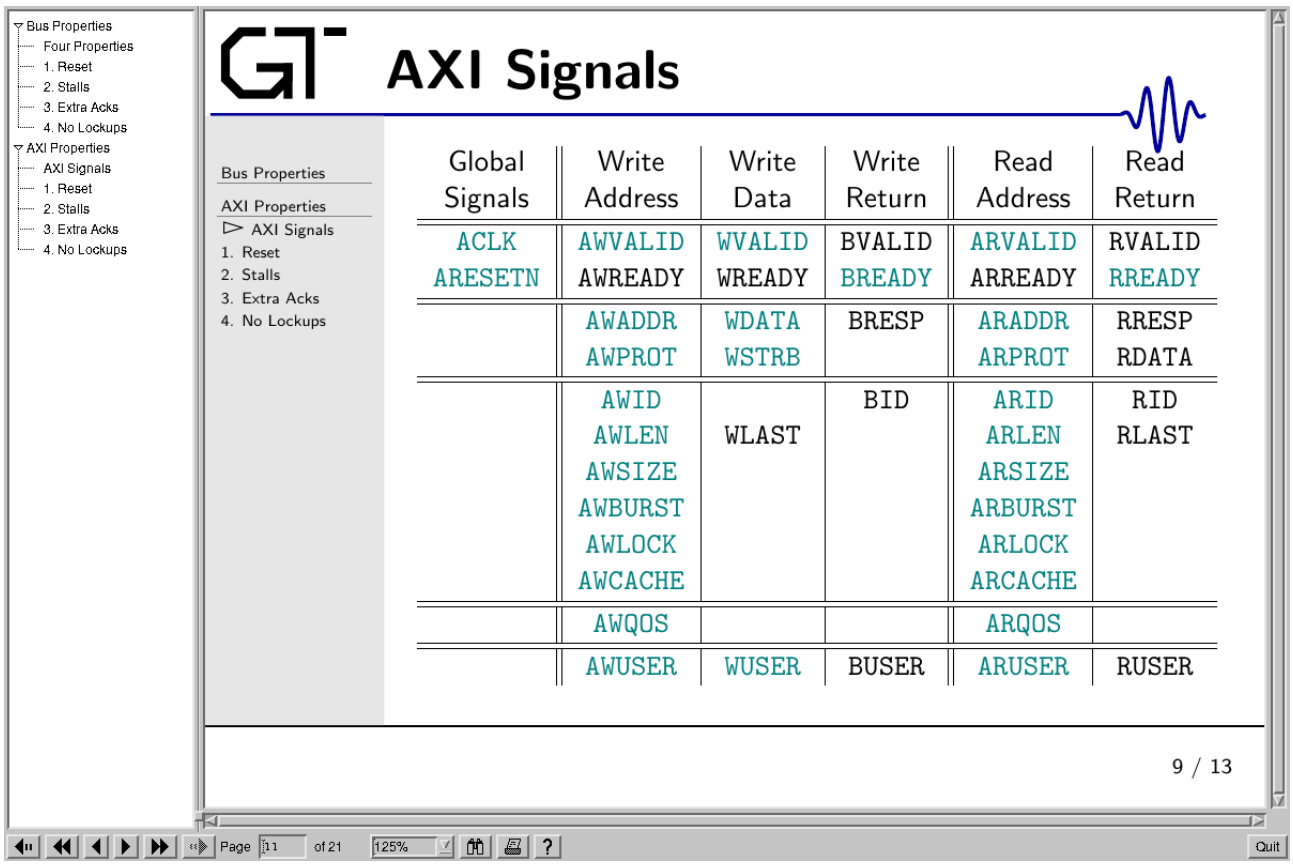


AXI Write Address, Write Data and Write Response channels.

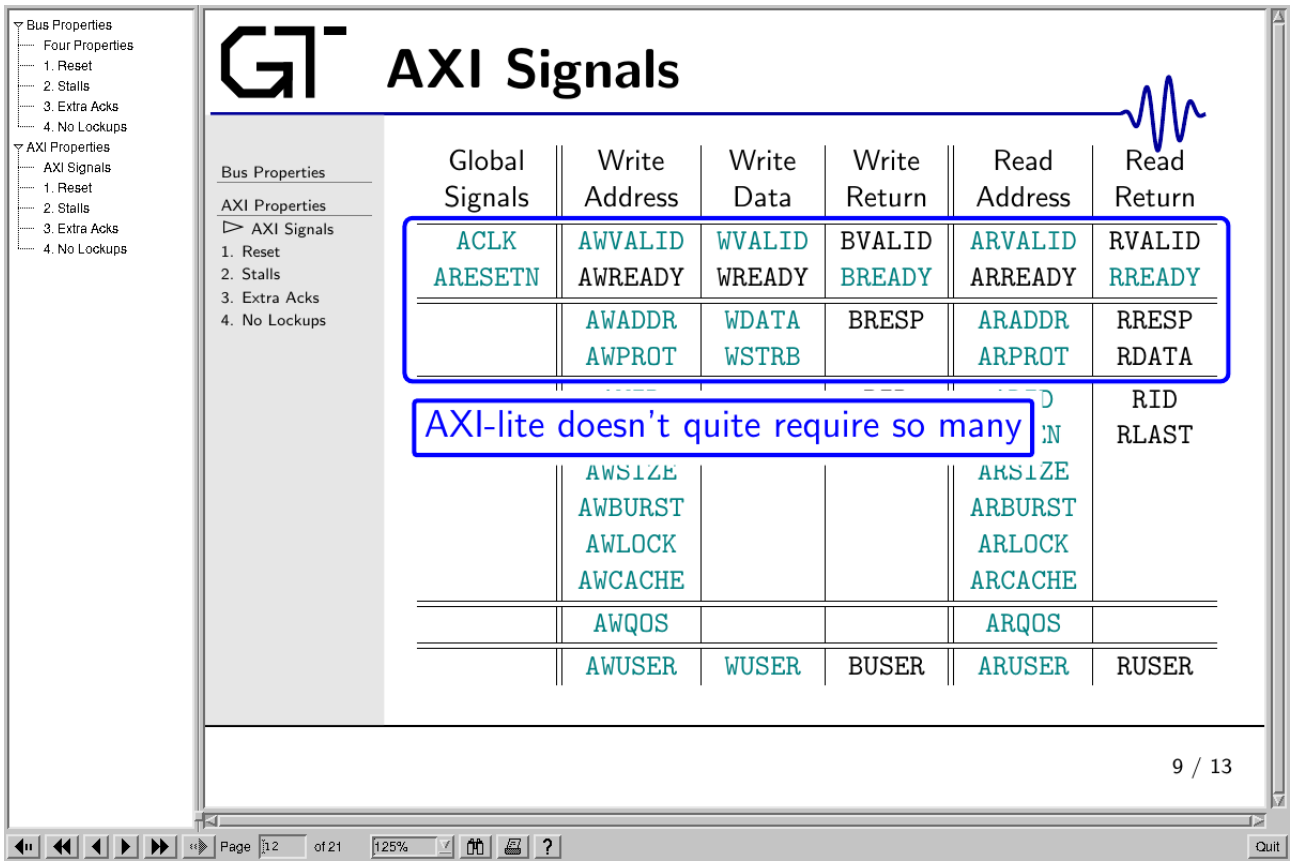
Using Alex Forencich "https://github.com/alexforencich/verilog-axi" since it can tested with Python, MyHDL and Icarus-Verilog.

The slides describing the AXI signals are from "https://github.com/ZipCPU/wb2axip/blob/master/doc/busprops.pdf"

Full AXI



AXI-Lite



On Raspberry Pi 4

“pip install myhdl”

“~/.local/share/myhdl/cosimulation/icarus/”

make

“sudo cp myhdl.vpi /usr/lib/arm-linux-gnueabihf/ivl/”

“cd tmp”

“git clone https://github.com/alexforencich/verilog-axi.git”

“cd verilog-axi/tb”

“python test_axil_adapter_16_32.py” creates test_axil_adapter_16_32.lxt

```
addr = 4
        test_data = b'\x11\x22\x33\x44'

        axil_master_inst.init_write(addr, test_data)

        yield axil_master_inst.wait()
        yield clk.posedge

        data = axil_ram_inst.read_mem(addr&0xffffff80, 32)
        for i in range(0, len(data), 16):
            print(" ".join(("{:02x}".format(c) for c in bytearray(data[i:i+16]))))

        assert axil_ram_inst.read_mem(addr, len(test_data)) == test_data

        yield delay(100)

        yield clk.posedge
        print("test 2: read")
        current_test.next = 2

        addr = 4
        test_data = b'\x11\x22\x33\x44'
```

gtkwave test_axil_adapter_16_32.lxt