Homework Assignment 3

Custom IP Creation, AXI Stream Transaction and Testbench in Vivado Environment

Tasks 1:

Design a custom IP that receives 100, 32bits blocks and store them inside a memory. This IP should calculate the summation of all of these inputs and store the result inside a register. Also, AXI Lite interface is needed for receiving a constant value as an offset.

Finally, summation register output should be added to the offset and each input block and the result should be sent back to the processor.

Inputs: $X_0, X_1, X_2 \ldots X_{99}$.

Outputs: $Y_0, Y_1, Y_2 \ldots Y_{99}$.

Sum = $X_0 + X_1 + X_2 +_{\ldots} + X_{99}$

$Y_0 = X_0 + \text{Sum} + \text{offset}$

$Y_1 = X_1 + \text{Sum} + \text{offset}$

$Y_2 = X_2 + \text{Sum} + \text{offset}$

.
.
.

$Y_{99} = X_{99} + \text{Sum} + \text{offset}$

Note: The maximum value for each input block is 255 in decimal.

Input and output ports should be exactly the same as the following IP:

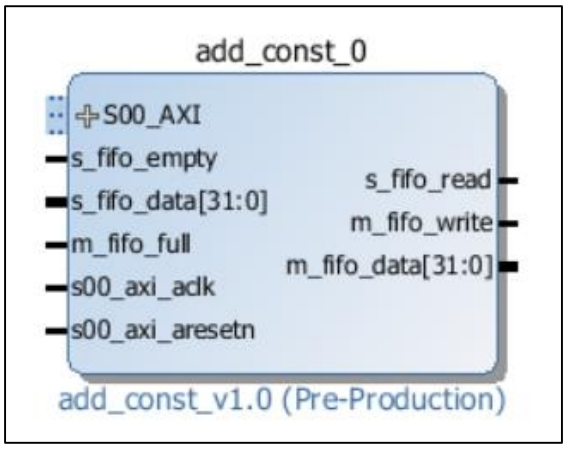
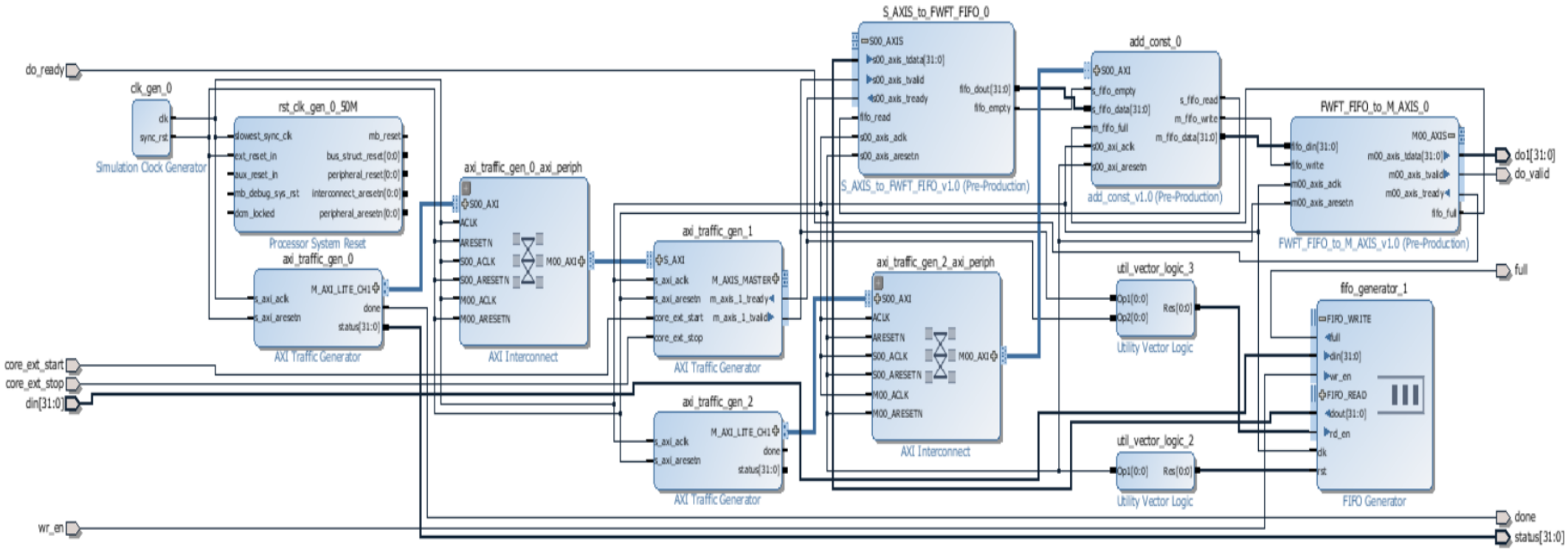


Figure 1. Example of IP symbol with input and output ports.

Tasks 2:

Create a testbench in Vivado which can verify the functionality of your custom IP.

Testbench block design example:


do_ready
clk_gen_0
clk
sync_rst
Simulation Clock Generator
rst_clk_gen_0_50M
slowest_sync_clk
ext_reset_in
aux_reset_in
mb_debug_sys_rst
dcm_locked
mb_reset
bus_struct_reset[0:0]
peripheral_reset[0:0]
interconnect_aresetn[0:0]
peripheral_aresetn[0:0]
Processor System Reset
axi_traffic_gen_0
s_axi_aclk
s_axi_aresetn
M_AXI_LITE_CH1
done
status[31:0]
AXI Traffic Generator
axi_traffic_gen_0_axi_periph
S00_AXI
ACLK
ARESETN
S00_ACLK
S00_ARESETN
M00_ACLK
M00_ARESETN
M00_AXI
AXI Interconnect
axi_traffic_gen_1
S_AXI
s_axi_aclk
s_axi_aresetn
core_ext_start
core_ext_stop
M_AXIS_MASTER
m_axis_1_tready
m_axis_1_tvalid
AXI Traffic Generator
axi_traffic_gen_2
s_axi_aclk
s_axi_aresetn
M_AXI_LITE_CH1
done
status[31:0]
AXI Traffic Generator
S_AXIS_to_FWFT_FIFO_0
S00_AXIS
s00_axis_tdata[31:0]
s00_axis_tvalid
s00_axis_tready
fifo_read
s00_axis_aclk
s00_axis_aresetn
fifo_dout[31:0]
fifo_empty
S_AXIS_to_FWFT_FIFO_v1.0 (Pre-Production)
axi_traffic_gen_2_axi_periph
S00_AXI
ACLK
ARESETN
S00_ACLK
S00_ARESETN
M00_ACLK
M00_ARESETN
M00_AXI
AXI Interconnect
add_const_0
S00_AXI
s_fifo_empty
s_fifo_data[31:0]
m_fifo_full
s00_axi_aclk
s00_axi_aresetn
s_fifo_read
m_fifo_write
m_fifo_data[31:0]
add_const_v1.0 (Pre-Production)
util_vector_logic_3
Op1[0:0]
Op2[0:0]
Res[0:0]
Utility Vector Logic
util_vector_logic_2
Op1[0:0]
Res[0:0]
Utility Vector Logic
FWFT_FIFO_to_M_AXIS_0
fifo_din[31:0]
fifo_write
m00_axis_aclk
m00_axis_aresetn
M00_AXIS
m00_axis_tdata[31:0]
m00_axis_tvalid
m00_axis_tready
fifo_full
FWFT_FIFO_to_M_AXIS_v1.0 (Pre-Production)
fifo_generator_1
FIFO_WRITE
full
din[31:0]
wr_en
FIFO_READ
dout[31:0]
rd_en
clk
rst
FIFO Generator
core_ext_start
core_ext_stop
din[31:0]
wr_en
do1[31:0]
do_valid
full
done
status[31:0]

You should also write a VHDL testbench corresponding to you block design.

Tasks 3:

- Build a project in Vivado to verify the functionality of your design experimentally on the board.

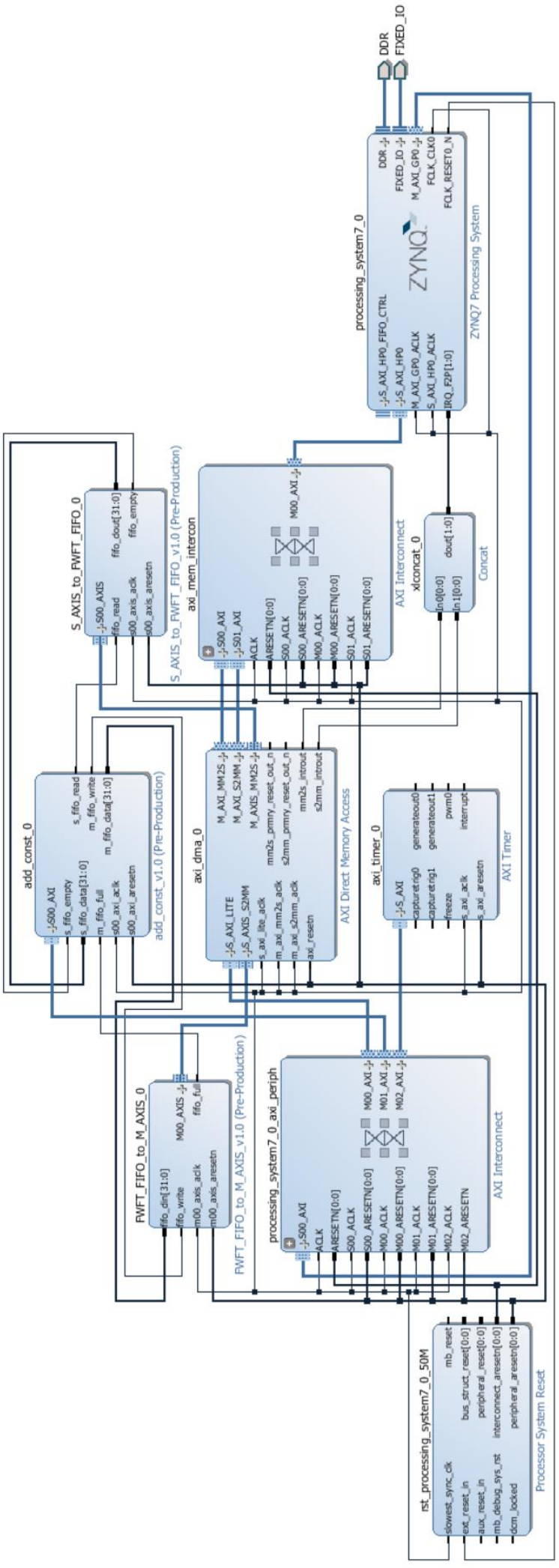
  Your design should contain the following IPs:

  1. Zynq Processing System
  2. Your own Custom IP
  3. S_AXIS_to_FWFT
  4. FWFT_to_M_AXIS
  5. AXI DMA
  6. AXI Timer
  7. Concat

- Write the reference C code corresponding to the custom IP. Therefore, we can verify the result which we are receiving from hardware and also compare the execution time of same process running on the ARM core and FPGA.

- You should use AXI Timer to measure the execution time on software and hardware.

Top level block design:



Deliverables:

A zip file include the following files:

1- Project folder that includes all the software and hardware files necessary to regenerate the results for task 2 and task 3.
2- Custom IP resources
3- VHDL testbench
4- Xilinx SDK C code
5- Screenshot from the SDK console which indicates software execution time, hardware execution time and result verification status (Pass or Fail).

Important Dates

Deliverables Due	**Thursday, 02/17/2015, 5 PM**