`include "axi\_bus.sv"

module axi2apb\_wrap

#(

parameter AXI\_ADDR\_WIDTH = 32,

parameter AXI\_DATA\_WIDTH = 32,

parameter AXI\_USER\_WIDTH = 6,

parameter AXI\_ID\_WIDTH = 6,

parameter APB\_ADDR\_WIDTH = 32

)

(

input logic clk\_i,

input logic rst\_ni,

input logic test\_en\_i,

AXI\_BUS.Slave axi\_slave,

APB\_BUS.Master apb\_master

);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\* AXI2APB WRAPER \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

generate if (AXI\_DATA\_WIDTH == 32)

begin

axi2apb32

#(

.AXI4\_ADDRESS\_WIDTH ( AXI\_ADDR\_WIDTH ),

.AXI4\_RDATA\_WIDTH ( AXI\_DATA\_WIDTH ),

.AXI4\_WDATA\_WIDTH ( AXI\_DATA\_WIDTH ),

.AXI4\_ID\_WIDTH ( AXI\_ID\_WIDTH ),

.AXI4\_USER\_WIDTH ( 1 ),

.BUFF\_DEPTH\_SLAVE ( 2 ),

.APB\_ADDR\_WIDTH ( APB\_ADDR\_WIDTH )

)

axi2apb\_i

(

.ACLK ( clk\_i ),

.ARESETn ( rst\_ni ),

.test\_en\_i ( test\_en\_i ),

.AWID\_i ( axi\_slave.aw\_id ),

.AWADDR\_i ( axi\_slave.aw\_addr ),

.AWLEN\_i ( axi\_slave.aw\_len ),

.AWSIZE\_i ( axi\_slave.aw\_size ),

.AWBURST\_i ( axi\_slave.aw\_burst ),

.AWLOCK\_i ( axi\_slave.aw\_lock ),

.AWCACHE\_i ( axi\_slave.aw\_cache ),

.AWPROT\_i ( axi\_slave.aw\_prot ),

.AWREGION\_i ( axi\_slave.aw\_region ),

.AWUSER\_i ( axi\_slave.aw\_user ),

.AWQOS\_i ( axi\_slave.aw\_qos ),

.AWVALID\_i ( axi\_slave.aw\_valid ),

.AWREADY\_o ( axi\_slave.aw\_ready ),

.WDATA\_i ( axi\_slave.w\_data ),

.WSTRB\_i ( axi\_slave.w\_strb ),

.WLAST\_i ( axi\_slave.w\_last ),

.WUSER\_i ( axi\_slave.w\_user ),

.WVALID\_i ( axi\_slave.w\_valid ),

.WREADY\_o ( axi\_slave.w\_ready ),

.BID\_o ( axi\_slave.b\_id ),

.BRESP\_o ( axi\_slave.b\_resp ),

.BVALID\_o ( axi\_slave.b\_valid ),

.BUSER\_o ( axi\_slave.b\_user ),

.BREADY\_i ( axi\_slave.b\_ready ),

.ARID\_i ( axi\_slave.ar\_id ),

.ARADDR\_i ( axi\_slave.ar\_addr ),

.ARLEN\_i ( axi\_slave.ar\_len ),

.ARSIZE\_i ( axi\_slave.ar\_size ),

.ARBURST\_i ( axi\_slave.ar\_burst ),

.ARLOCK\_i ( axi\_slave.ar\_lock ),

.ARCACHE\_i ( axi\_slave.ar\_cache ),

.ARPROT\_i ( axi\_slave.ar\_prot ),

.ARREGION\_i ( axi\_slave.ar\_region ),

.ARUSER\_i ( axi\_slave.ar\_user ),

.ARQOS\_i ( axi\_slave.ar\_qos ),

.ARVALID\_i ( axi\_slave.ar\_valid ),

.ARREADY\_o ( axi\_slave.ar\_ready ),

.RID\_o ( axi\_slave.r\_id ),

.RDATA\_o ( axi\_slave.r\_data ),

.RRESP\_o ( axi\_slave.r\_resp ),

.RLAST\_o ( axi\_slave.r\_last ),

.RUSER\_o ( axi\_slave.r\_user ),

.RVALID\_o ( axi\_slave.r\_valid ),

.RREADY\_i ( axi\_slave.r\_ready ),

.PENABLE ( apb\_master.penable ),

.PWRITE ( apb\_master.pwrite ),

.PADDR ( apb\_master.paddr ),

.PSEL ( apb\_master.psel ),

.PWDATA ( apb\_master.pwdata ),

.PRDATA ( apb\_master.prdata ),

.PREADY ( apb\_master.pready ),

.PSLVERR ( apb\_master.pslverr )

);

end

else if (AXI\_DATA\_WIDTH == 64)

begin

axi2apb

#(

.AXI4\_ADDRESS\_WIDTH ( AXI\_ADDR\_WIDTH ),

.AXI4\_RDATA\_WIDTH ( AXI\_DATA\_WIDTH ),

.AXI4\_WDATA\_WIDTH ( AXI\_DATA\_WIDTH ),

.AXI4\_ID\_WIDTH ( AXI\_ID\_WIDTH ),

.AXI4\_USER\_WIDTH ( 1 ),

.BUFF\_DEPTH\_SLAVE ( 2 ),

.APB\_ADDR\_WIDTH ( APB\_ADDR\_WIDTH )

)

axi2apb\_i

(

.ACLK ( clk\_i ),

.ARESETn ( rst\_ni ),

.test\_en\_i ( test\_en\_i ),

.AWID\_i ( axi\_slave.aw\_id ),

.AWADDR\_i ( axi\_slave.aw\_addr ),

.AWLEN\_i ( axi\_slave.aw\_len ),

.AWSIZE\_i ( axi\_slave.aw\_size ),

.AWBURST\_i ( axi\_slave.aw\_burst ),

.AWLOCK\_i ( axi\_slave.aw\_lock ),

.AWCACHE\_i ( axi\_slave.aw\_cache ),

.AWPROT\_i ( axi\_slave.aw\_prot ),

.AWREGION\_i ( axi\_slave.aw\_region ),

.AWUSER\_i ( axi\_slave.aw\_user ),

.AWQOS\_i ( axi\_slave.aw\_qos ),

.AWVALID\_i ( axi\_slave.aw\_valid ),

.AWREADY\_o ( axi\_slave.aw\_ready ),

.WDATA\_i ( axi\_slave.w\_data ),

.WSTRB\_i ( axi\_slave.w\_strb ),

.WLAST\_i ( axi\_slave.w\_last ),

.WUSER\_i ( axi\_slave.w\_user ),

.WVALID\_i ( axi\_slave.w\_valid ),

.WREADY\_o ( axi\_slave.w\_ready ),

.BID\_o ( axi\_slave.b\_id ),

.BRESP\_o ( axi\_slave.b\_resp ),

.BVALID\_o ( axi\_slave.b\_valid ),

.BUSER\_o ( axi\_slave.b\_user ),

.BREADY\_i ( axi\_slave.b\_ready ),

.ARID\_i ( axi\_slave.ar\_id ),

.ARADDR\_i ( axi\_slave.ar\_addr ),

.ARLEN\_i ( axi\_slave.ar\_len ),

.ARSIZE\_i ( axi\_slave.ar\_size ),

.ARBURST\_i ( axi\_slave.ar\_burst ),

.ARLOCK\_i ( axi\_slave.ar\_lock ),

.ARCACHE\_i ( axi\_slave.ar\_cache ),

.ARPROT\_i ( axi\_slave.ar\_prot ),

.ARREGION\_i ( axi\_slave.ar\_region ),

.ARUSER\_i ( axi\_slave.ar\_user ),

.ARQOS\_i ( axi\_slave.ar\_qos ),

.ARVALID\_i ( axi\_slave.ar\_valid ),

.ARREADY\_o ( axi\_slave.ar\_ready ),

.RID\_o ( axi\_slave.r\_id ),

.RDATA\_o ( axi\_slave.r\_data ),

.RRESP\_o ( axi\_slave.r\_resp ),

.RLAST\_o ( axi\_slave.r\_last ),

.RUSER\_o ( axi\_slave.r\_user ),

.RVALID\_o ( axi\_slave.r\_valid ),

.RREADY\_i ( axi\_slave.r\_ready ),

.PENABLE ( apb\_master.penable ),

.PWRITE ( apb\_master.pwrite ),

.PADDR ( apb\_master.paddr ),

.PSEL ( apb\_master.psel ),

.PWDATA ( apb\_master.pwdata ),

.PRDATA ( apb\_master.prdata ),

.PREADY ( apb\_master.pready ),

.PSLVERR ( apb\_master.pslverr )

);

end

endgenerate

endmodule