module axi\_slice\_wrap

#(

parameter AXI\_ADDR\_WIDTH = 32,

parameter AXI\_DATA\_WIDTH = 64,

parameter AXI\_USER\_WIDTH = 6,

parameter AXI\_ID\_WIDTH = 6,

parameter SLICE\_DEPTH = 2

)

(

input logic clk\_i,

input logic rst\_ni,

input logic test\_en\_i,

AXI\_BUS.Slave axi\_slave,

AXI\_BUS.Master axi\_master

);

generate

if( SLICE\_DEPTH > 1)

begin : WITH\_SLICE

axi\_slice

#(

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

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

.AXI\_USER\_WIDTH ( AXI\_USER\_WIDTH ),

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

.SLICE\_DEPTH ( SLICE\_DEPTH )

)

axi\_slice\_i

(

.clk\_i ( clk\_i ),

.rst\_ni ( rst\_ni ),

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

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

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

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

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

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

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

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

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

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

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

.axi\_slave\_aw\_id\_i ( axi\_slave.aw\_id[AXI\_ID\_WIDTH-1:0]),

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

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

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

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

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

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

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

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

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

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

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

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

.axi\_slave\_ar\_id\_i ( axi\_slave.ar\_id[AXI\_ID\_WIDTH-1:0]),

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

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

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

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

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

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

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

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

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

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

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

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

.axi\_slave\_r\_id\_o ( axi\_slave.r\_id[AXI\_ID\_WIDTH-1:0] ),

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

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

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

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

.axi\_slave\_b\_id\_o ( axi\_slave.b\_id[AXI\_ID\_WIDTH-1:0] ),

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

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

.axi\_master\_aw\_valid\_o ( axi\_master.aw\_valid ),

.axi\_master\_aw\_addr\_o ( axi\_master.aw\_addr ),

.axi\_master\_aw\_prot\_o ( axi\_master.aw\_prot ),

.axi\_master\_aw\_region\_o( axi\_master.aw\_region ),

.axi\_master\_aw\_len\_o ( axi\_master.aw\_len ),

.axi\_master\_aw\_size\_o ( axi\_master.aw\_size ),

.axi\_master\_aw\_burst\_o ( axi\_master.aw\_burst ),

.axi\_master\_aw\_lock\_o ( axi\_master.aw\_lock ),

.axi\_master\_aw\_cache\_o ( axi\_master.aw\_cache ),

.axi\_master\_aw\_qos\_o ( axi\_master.aw\_qos ),

.axi\_master\_aw\_id\_o ( axi\_master.aw\_id[AXI\_ID\_WIDTH-1:0]),

.axi\_master\_aw\_user\_o ( axi\_master.aw\_user ),

.axi\_master\_aw\_ready\_i ( axi\_master.aw\_ready ),

.axi\_master\_ar\_valid\_o ( axi\_master.ar\_valid ),

.axi\_master\_ar\_addr\_o ( axi\_master.ar\_addr ),

.axi\_master\_ar\_prot\_o ( axi\_master.ar\_prot ),

.axi\_master\_ar\_region\_o( axi\_master.ar\_region ),

.axi\_master\_ar\_len\_o ( axi\_master.ar\_len ),

.axi\_master\_ar\_size\_o ( axi\_master.ar\_size ),

.axi\_master\_ar\_burst\_o ( axi\_master.ar\_burst ),

.axi\_master\_ar\_lock\_o ( axi\_master.ar\_lock ),

.axi\_master\_ar\_cache\_o ( axi\_master.ar\_cache ),

.axi\_master\_ar\_qos\_o ( axi\_master.ar\_qos ),

.axi\_master\_ar\_id\_o ( axi\_master.ar\_id[AXI\_ID\_WIDTH-1:0]),

.axi\_master\_ar\_user\_o ( axi\_master.ar\_user ),

.axi\_master\_ar\_ready\_i ( axi\_master.ar\_ready ),

.axi\_master\_w\_valid\_o ( axi\_master.w\_valid ),

.axi\_master\_w\_data\_o ( axi\_master.w\_data ),

.axi\_master\_w\_strb\_o ( axi\_master.w\_strb ),

.axi\_master\_w\_user\_o ( axi\_master.w\_user ),

.axi\_master\_w\_last\_o ( axi\_master.w\_last ),

.axi\_master\_w\_ready\_i ( axi\_master.w\_ready ),

.axi\_master\_r\_valid\_i ( axi\_master.r\_valid ),

.axi\_master\_r\_data\_i ( axi\_master.r\_data ),

.axi\_master\_r\_resp\_i ( axi\_master.r\_resp ),

.axi\_master\_r\_last\_i ( axi\_master.r\_last ),

.axi\_master\_r\_id\_i ( axi\_master.r\_id[AXI\_ID\_WIDTH-1:0] ),

.axi\_master\_r\_user\_i ( axi\_master.r\_user ),

.axi\_master\_r\_ready\_o ( axi\_master.r\_ready ),

.axi\_master\_b\_valid\_i ( axi\_master.b\_valid ),

.axi\_master\_b\_resp\_i ( axi\_master.b\_resp ),

.axi\_master\_b\_id\_i ( axi\_master.b\_id[AXI\_ID\_WIDTH-1:0] ),

.axi\_master\_b\_user\_i ( axi\_master.b\_user ),

.axi\_master\_b\_ready\_o ( axi\_master.b\_ready )

);

end

else

begin : NO\_SLICE

assign axi\_master.aw\_valid = axi\_slave.aw\_valid;

assign axi\_master.aw\_addr = axi\_slave.aw\_addr;

assign axi\_master.aw\_prot = axi\_slave.aw\_prot;

assign axi\_master.aw\_region = axi\_slave.aw\_region;

assign axi\_master.aw\_len = axi\_slave.aw\_len;

assign axi\_master.aw\_size = axi\_slave.aw\_size;

assign axi\_master.aw\_burst = axi\_slave.aw\_burst;

assign axi\_master.aw\_lock = axi\_slave.aw\_lock;

assign axi\_master.aw\_cache = axi\_slave.aw\_cache;

assign axi\_master.aw\_qos = axi\_slave.aw\_qos;

assign axi\_master.aw\_id = axi\_slave.aw\_id;

assign axi\_master.aw\_user = axi\_slave.aw\_user;

assign axi\_slave.aw\_ready = axi\_master.aw\_ready;

assign axi\_master.ar\_valid = axi\_slave.ar\_valid;

assign axi\_master.ar\_addr = axi\_slave.ar\_addr;

assign axi\_master.ar\_prot = axi\_slave.ar\_prot;

assign axi\_master.ar\_region = axi\_slave.ar\_region;

assign axi\_master.ar\_len = axi\_slave.ar\_len;

assign axi\_master.ar\_size = axi\_slave.ar\_size;

assign axi\_master.ar\_burst = axi\_slave.ar\_burst;

assign axi\_master.ar\_lock = axi\_slave.ar\_lock;

assign axi\_master.ar\_cache = axi\_slave.ar\_cache;

assign axi\_master.ar\_qos = axi\_slave.ar\_qos;

assign axi\_master.ar\_id = axi\_slave.ar\_id;

assign axi\_master.ar\_user = axi\_slave.ar\_user;

assign axi\_slave.ar\_ready = axi\_master.ar\_ready;

assign axi\_master.w\_valid = axi\_slave.w\_valid;

assign axi\_master.w\_data = axi\_slave.w\_data;

assign axi\_master.w\_strb = axi\_slave.w\_strb;

assign axi\_master.w\_user = axi\_slave.w\_user;

assign axi\_master.w\_last = axi\_slave.w\_last;

assign axi\_slave.w\_ready = axi\_master.w\_ready;

assign axi\_slave.r\_valid = axi\_master.r\_valid;

assign axi\_slave.r\_data = axi\_master.r\_data;

assign axi\_slave.r\_resp = axi\_master.r\_resp;

assign axi\_slave.r\_last = axi\_master.r\_last;

assign axi\_slave.r\_id = axi\_master.r\_id;

assign axi\_slave.r\_user = axi\_master.r\_user;

assign axi\_master.r\_ready = axi\_slave.r\_ready;

assign axi\_slave.b\_valid = axi\_master.b\_valid;

assign axi\_slave.b\_resp = axi\_master.b\_resp;

assign axi\_slave.b\_id = axi\_master.b\_id;

assign axi\_slave.b\_user = axi\_master.b\_user;

assign axi\_master.b\_ready = axi\_slave.b\_ready;

end

endgenerate

endmodule