`include "axi\_bus.sv"

module axi\_mem\_if\_SP\_wrap

#(

parameter AXI\_ADDR\_WIDTH = 32,

parameter AXI\_DATA\_WIDTH = 64,

parameter AXI\_ID\_WIDTH = 10,

parameter AXI\_USER\_WIDTH = 10,

parameter MEM\_ADDR\_WIDTH = 10

)(

// Clock and Reset

input logic clk,

input logic rst\_n,

input logic test\_en\_i,

output logic mem\_req\_o,

output logic [MEM\_ADDR\_WIDTH-1:0] mem\_addr\_o,

output logic mem\_we\_o,

output logic [AXI\_DATA\_WIDTH/8-1:0] mem\_be\_o,

input logic [AXI\_DATA\_WIDTH-1:0] mem\_rdata\_i,

output logic [AXI\_DATA\_WIDTH-1:0] mem\_wdata\_o,

AXI\_BUS.Slave slave

);

logic cen;

logic wen;

axi\_mem\_if\_SP

#(

.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 ( AXI\_USER\_WIDTH ),

.MEM\_ADDR\_WIDTH ( MEM\_ADDR\_WIDTH )

)

axi\_mem\_if\_SP\_i

(

.ACLK ( clk ),

.ARESETn ( rst\_n ),

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

.AWID\_i ( slave.aw\_id ),

.AWADDR\_i ( slave.aw\_addr ),

.AWLEN\_i ( slave.aw\_len ),

.AWSIZE\_i ( slave.aw\_size ),

.AWBURST\_i ( slave.aw\_burst ),

.AWLOCK\_i ( slave.aw\_lock ),

.AWCACHE\_i ( slave.aw\_cache ),

.AWPROT\_i ( slave.aw\_prot ),

.AWREGION\_i ( slave.aw\_region ),

.AWUSER\_i ( slave.aw\_user ),

.AWQOS\_i ( slave.aw\_qos ),

.AWVALID\_i ( slave.aw\_valid ),

.AWREADY\_o ( slave.aw\_ready ),

.WDATA\_i ( slave.w\_data ),

.WSTRB\_i ( slave.w\_strb ),

.WLAST\_i ( slave.w\_last ),

.WUSER\_i ( slave.w\_user ),

.WVALID\_i ( slave.w\_valid ),

.WREADY\_o ( slave.w\_ready ),

.BID\_o ( slave.b\_id ),

.BRESP\_o ( slave.b\_resp ),

.BVALID\_o ( slave.b\_valid ),

.BUSER\_o ( slave.b\_user ),

.BREADY\_i ( slave.b\_ready ),

.ARID\_i ( slave.ar\_id ),

.ARADDR\_i ( slave.ar\_addr ),

.ARLEN\_i ( slave.ar\_len ),

.ARSIZE\_i ( slave.ar\_size ),

.ARBURST\_i ( slave.ar\_burst ),

.ARLOCK\_i ( slave.ar\_lock ),

.ARCACHE\_i ( slave.ar\_cache ),

.ARPROT\_i ( slave.ar\_prot ),

.ARREGION\_i ( slave.ar\_region ),

.ARUSER\_i ( slave.ar\_user ),

.ARQOS\_i ( slave.ar\_qos ),

.ARVALID\_i ( slave.ar\_valid ),

.ARREADY\_o ( slave.ar\_ready ),

.RID\_o ( slave.r\_id ),

.RDATA\_o ( slave.r\_data ),

.RRESP\_o ( slave.r\_resp ),

.RLAST\_o ( slave.r\_last ),

.RUSER\_o ( slave.r\_user ),

.RVALID\_o ( slave.r\_valid ),

.RREADY\_i ( slave.r\_ready ),

.CEN\_o ( cen ),

.WEN\_o ( wen ),

.A\_o ( mem\_addr\_o ),

.D\_o ( mem\_wdata\_o ),

.BE\_o ( mem\_be\_o ),

.Q\_i ( mem\_rdata\_i )

);

assign mem\_req\_o = ~cen;

assign mem\_we\_o = ~wen;

endmodule