class mem\_driver extends uvm\_driver#(mem\_sequence\_item);

`uvm\_component\_utils(mem\_driver)

virtual intf vif;

function new(string name = "mem\_driver",uvm\_component parent);

super.new(name,parent);

endfunction

virtual function void build\_phase(uvm\_phase phase);

super.build\_phase(phase);

uvm\_config\_db#(virtual intf)::get(this,"\*","vif",vif);

endfunction

virtual task run\_phase(uvm\_phase phase);

super.run\_phase(phase);

forever begin

seq\_item\_port.get\_next\_item(req);

drive();

// $display("------------driver-------------");

// req.print();

seq\_item\_port.item\_done();

end

endtask

virtual task drive();

begin

@(vif.hclk);

vif.enable=req.enable;

vif.dina=req.dina;

vif.dinb=req.dinb;

vif.addr=req.addr;

vif.wr=req.wr;

vif.slave\_sel=req.slave\_sel;

write(vif.slave\_sel,vif.addr,vif.dina,vif.dinb);

read(vif.slave\_sel,vif.addr);

@(vif.hclk);

end

endtask

task write(input [1:0] sel, input [31:0] address, input [31:0] a, input [31:0] b);

begin

@(posedge vif.hclk)

vif.slave\_sel = sel;

vif.enable = 1'b1;

vif.addr = address;

@(posedge vif.hclk)

vif.dina = a;

vif.dinb = b;

vif.wr = 1'b1;

@(posedge vif.hclk)

vif.enable = 1'b0;

end

endtask

task read(input [1:0] sel, input [31:0] address);

begin

@(posedge vif.hclk)

vif.enable = 1'b1;

vif.slave\_sel = sel;

vif.addr = address;

@(posedge vif.hclk)

vif.wr = 1'b0;

// 3 beats for read

@(posedge vif.hclk)

vif.wr = 1'b0;

@(posedge vif.hclk)

vif.wr = 1'b0;

@(posedge vif.hclk)

vif.wr = 1'b0;

@(posedge vif.hclk)

vif.enable = 1'b0;

end

endtask

endclass