**DESIGN INTERNSHIP (DI-26)**

FOR

****

**AHB TO APB BRIDGE DESIGN**

**PROJECT REPORT**

DONE BY -

**JAYARAM G**

**MODULE-1 AHB MASTER:**

module ahb\_master (input hclk,hresetn, hreadyout, input [31:0] hrdata, input [1:0] hresp,

output reg [31:0] haddr, hwdata, output reg hwrite,hreadyin, output reg [1:0] htrans);

reg [2:0] hburst;

reg [2:0] hsize;

task single\_write();

begin

@(posedge hclk);

#1;

begin

hwrite=1;

htrans=2'b10;

hsize=0;

hburst=0;

hreadyin=1;

haddr=32'h8000\_0001;

end

@(posedge hclk);

#1;

begin

htrans=2'b00;

hwdata=8'h80;

end

end

endtask

task single\_read();

begin

@(posedge hclk);

#1;

begin

hwrite=0;

htrans=2'b10;

hsize=0;

hburst=0;

hreadyin=1;

haddr=32'h8000\_0001;

end

@(posedge hclk);

#1;

begin

htrans=2'b00;

end

endendtask

endmodule

**MODULE-2 AHB SLAVE INTERFACE**

module ahb\_slave\_interface(input hclk,hresetn,hwrite,hready\_in, input [1:0] htrans,input [31:0] haddr,hwdata,

output reg valid,hwritereg,hwritereg\_1, output [1:0] hresp,output reg [2:0] temp\_selx,

output reg [31:0] haddr\_1,haddr\_2,hwdata\_1,hwdata\_2, output [31:0] hrdata,input [31:0] prdata);

always @(posedge hclk)

begin

if(!hresetn)

begin

haddr\_1<=0;

haddr\_2<=0;

end

else

begin

haddr\_1<=haddr;

haddr\_2<=haddr\_1;

end

end

always @(posedge hclk)

begin

if(!hresetn)

begin

hwdata\_1<=0;

hwdata\_2<=0;

end

else

begin

hwdata\_1<=hwdata;

hwdata\_2<=hwdata\_1;

end

end

always @(posedge hclk)

begin

if(!hresetn)

begin

hwritereg<=0;

hwritereg\_1<=0;

end

else

begin

hwritereg<=hwrite;

hwritereg\_1<=hwritereg;

end

end

always @(\*)

begin

valid=1'b0;

if(hready\_in==1 && haddr>=32'h0000\_0000 && haddr<32'h8c00\_0000 &&

htrans==2'b10||htrans==2'b11)

valid=1;

else

valid=0;

end

always @(\*)

begin

temp\_selx=3'b000;

if ( haddr>=32'h0000\_0000 && haddr<32'h8400\_0000)

temp\_selx=3'b001;

if ( haddr>=32'h8400\_0000 && haddr<32'h8800\_0000)

temp\_selx=3'b010;

if ( haddr>=32'h8800\_0000 && haddr<=32'h8c00\_0000)

temp\_selx=3'b000;

end

assign hrdata=prdata;

assign hresp=2'b0;

endmodule

**MODULE-3 APB CONTROLLER**

module apb\_controller (input hclk,hresetn,hwrite\_reg,hwrite,valid,

input [31:0]haddr,hwdata,hwdata1,hwdata2,haddr1,haddr2,pr\_data,

input [2:0] temp\_selx,

output reg penable, pwrite,

output reg hr\_readyout,

output reg [2:0] psel,

output reg [31:0] paddr,pwdata);

parameter ST\_IDLE=3'b000,

ST\_READ=3'b001,

ST\_RENABLE=3'b010,

ST\_WWAIT=3'b011,

ST\_WRITE=3'b100,

ST\_WRITEP=3'b101,

ST\_WENABLE=3'b110,

ST\_WENABLEP=3'b111;

reg [2:0] PS,NS;

always @(posedge hclk)

begin

if (!hresetn)

PS<=ST\_IDLE;

else

PS<=NS;

end

always @(\*)

begin

NS=ST\_IDLE;

case(PS)

ST\_IDLE :

if(valid==1&&hwrite==1)

NS=ST\_WWAIT;

else if(valid==1&&hwrite==0)

NS=ST\_READ;

else if(valid==0)

NS=ST\_IDLE;

ST\_READ: NS=ST\_RENABLE;

ST\_RENABLE:

if(valid==1&&hwrite==1)

NS=ST\_WWAIT;

else if(valid==1&&hwrite==0)

NS=ST\_READ;

else if(valid==0)

NS=ST\_IDLE;

ST\_WWAIT:

if (valid==1)

NS=ST\_WRITEP;

else if(valid==0)

NS=ST\_WRITE;

ST\_WRITE:

if (valid==1)

NS=ST\_WENABLEP;

else if(valid==0)

NS=ST\_WENABLE;

ST\_WRITEP: NS=ST\_WENABLEP;

ST\_WENABLEP:

if (valid==1&&hwrite\_reg==1)

NS=ST\_WRITEP;

else if (valid==0&&hwrite\_reg==1)

NS=ST\_WRITE;

else if(hwrite\_reg==0)

NS=ST\_READ;

ST\_WENABLE:

if (valid==1 && hwrite==1)

NS=ST\_WWAIT;

else if (valid==1 && hwrite==0)

NS=ST\_READ;

else

NS=ST\_IDLE;

endcase

end

reg penable\_temp,pwrite\_temp,hr\_readyout\_temp;

reg [2:0] psel\_temp;

reg [31:0] paddr\_temp,pwdata\_temp;

always @(\*)

begin

case(PS)

ST\_IDLE: begin

if(valid==1&&hwrite==0)

begin

paddr\_temp=haddr;

pwrite\_temp=hwrite;

psel\_temp=temp\_selx;

penable\_temp=0;

hr\_readyout\_temp=0;

end

else if(valid==1&&hwrite==1)

begin

psel\_temp=0;

penable\_temp=0;

hr\_readyout\_temp=1;

end

else

begin

psel\_temp=0;

penable\_temp=0;

hr\_readyout\_temp=1;

end

end

ST\_READ: begin

penable\_temp=1;

hr\_readyout\_temp=1;

end

ST\_RENABLE: begin

if(valid==1&&hwrite==0)

begin

paddr\_temp=haddr;

pwrite\_temp=hwrite;

psel\_temp=temp\_selx;

penable\_temp=0;

hr\_readyout\_temp=0;

end

else if(valid==1&&hwrite==1)

begin

psel\_temp=0;

penable\_temp=0;

hr\_readyout\_temp=1;

end

else

begin

psel\_temp=0;

penable\_temp=0;

hr\_readyout\_temp=1;

end

end

ST\_WWAIT: begin

paddr\_temp=haddr1;

pwdata\_temp=hwdata;

pwrite\_temp=hwrite;

psel\_temp=temp\_selx;

penable\_temp=0;

hr\_readyout\_temp=0;

end

ST\_WRITE:begin

penable\_temp=1;

hr\_readyout\_temp=1;

end

ST\_WRITEP:

begin

penable\_temp=1;

hr\_readyout\_temp=1;

end

ST\_WENABLE:

begin

hr\_readyout\_temp=1;

penable\_temp=0;

psel\_temp=0;

end

ST\_WENABLEP:

begin

paddr\_temp=haddr2;

hr\_readyout\_temp=0;

pwdata\_temp=hwdata;

penable\_temp=1;

end

endcase

end

always @(posedge hclk)

begin

if(!hresetn)

begin

paddr<=0;

pwdata<=0;

pwrite<=0;

psel<=0;

penable<=0;

hr\_readyout<=1;

end

else

begin

paddr<=paddr\_temp;

pwdata<=pwdata\_temp;

pwrite<=pwrite\_temp;

psel<=psel\_temp;

penable<=penable\_temp;

hr\_readyout<=hr\_readyout\_temp;

end

end

endmodule

**MODULE-4 APB INTERFACE**

module apb\_interface (input pwrite,penable,input [2:0] pselx, input [31:0] paddr,pwdata,

output pwrite\_out,penable\_out,output [2:0] psel\_out,output [31:0] paddr\_out,pwdata\_out,output reg [31:0] prdata);

assign pwrite\_out=pwrite;

assign penable\_out=penable;

assign psel\_out=pselx;

assign pwdata\_out=pwdata;

assign paddr\_out=paddr;

always @(\*)

begin

if(!pwrite==1 &&penable)

prdata=8'd25;

end

endmodule

**MODULE-5 BRIDGE TOP**

module bridge\_top(input hclk,hresetn,hwrite,hready\_in,input [1:0] htrans,input [31:0] haddr,hwdata,prdata,

output penable, pwrite, hr\_readyout,output [2:0] psel,

output [1:0] hresp, output [31:0] paddr,pwdata,hrdata);

wire [31:0] hwdata\_1,hwdata\_2,haddr\_1,haddr\_2;

wire valid;

wire [2:0] temp\_selx;

wire hwritereg,hwritereg\_1;

ahb\_slave\_interface

ahb\_sl(hclk,hresetn,hwrite,hready\_in,htrans,haddr,hwdata,valid,hwritereg,hwritereg\_1,

hresp,temp\_selx,haddr\_1,haddr\_2,hwdata\_1,hwdata\_2,hrdata,prdata);

apb\_controller

apb\_c(hclk,hresetn,hwritereg,hwrite,valid,haddr,hwdata,hwdata\_1,hwdata\_2,haddr\_1,haddr\_2,prdata,

temp\_selx,penable,pwrite,hr\_readyout,psel,paddr,pwdata);

endmodule

**MODULE-6 TOP MODULE**

module top();

reg hclk,hresetn;

wire hreadyout,hwrite,hreadyin;

wire [31:0] hrdata,haddr,hwdata,paddr,pwdata,paddr\_out,pwdata\_out,prdata;

wire [1:0] hresp,htrans;

wire penable,pwrite,pwrite\_out,penable\_out;

wire [2:0] psel,psel\_out;

ahb\_master

ahb(hclk,hresetn,hreadyout,hrdata,hresp,haddr,hwdata,hwrite,hreadyin,htrans);

bridge\_top

bridge(hclk,hresetn,hwrite,hreadyin,htrans,haddr,hwdata,prdata,penable,pwrite,hreadyout,

psel,hresp,paddr,pwdata,hrdata);

apb\_interface

apb(pwrite,penable,psel,paddr,pwdata,pwrite\_out,penable\_out,psel\_out,paddr\_out,pwdata\_out,prdata);

initial

begin

hclk=1'b0;

forever #10 hclk=~hclk;

end

task reset();

begin

@(negedge hclk)

hresetn=1'b0;

@(negedge hclk)

hresetn=1'b1;

end

endtask

initial

begin

reset;

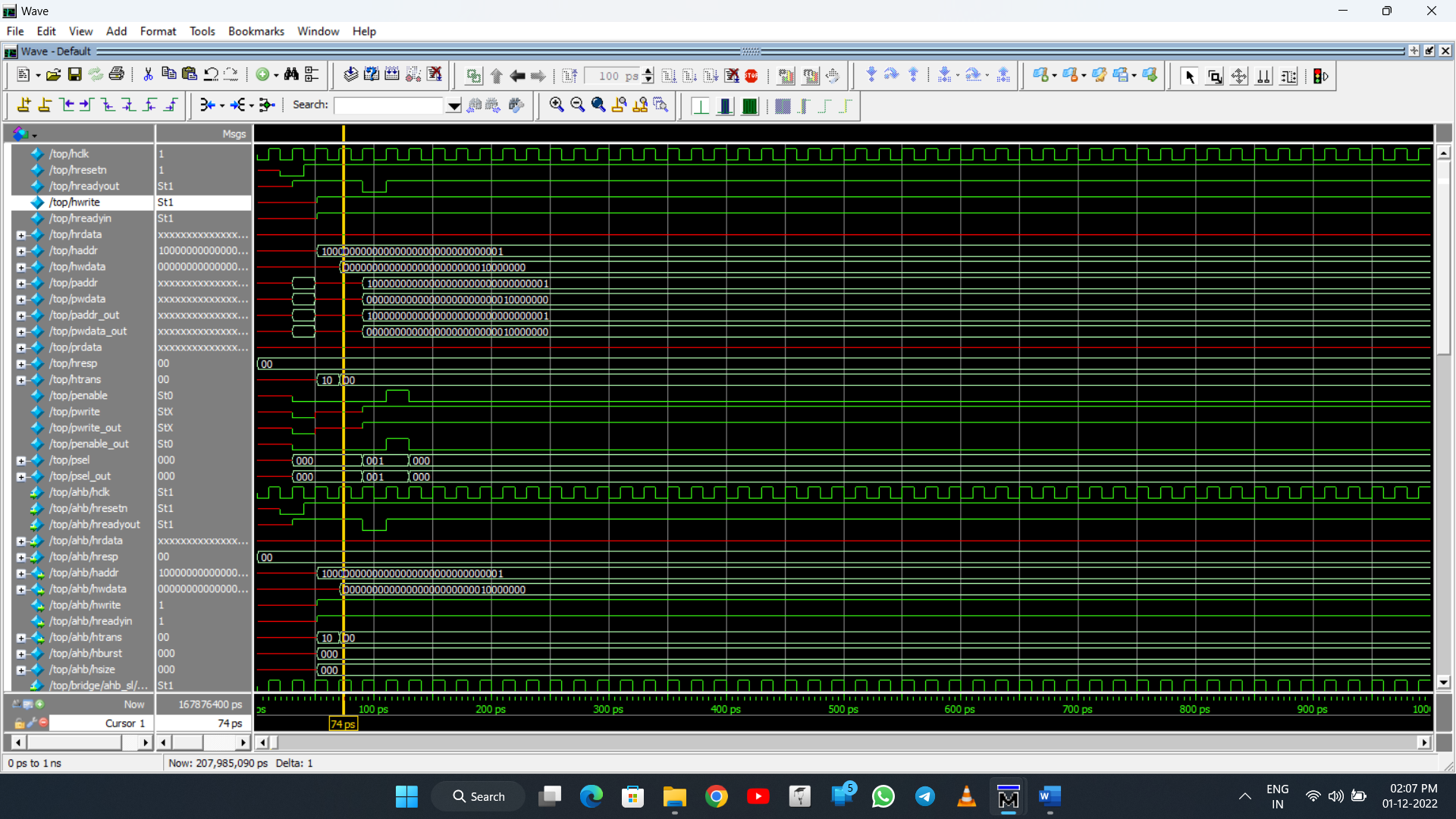
ahb.single\_write();

//ahb.single\_read();

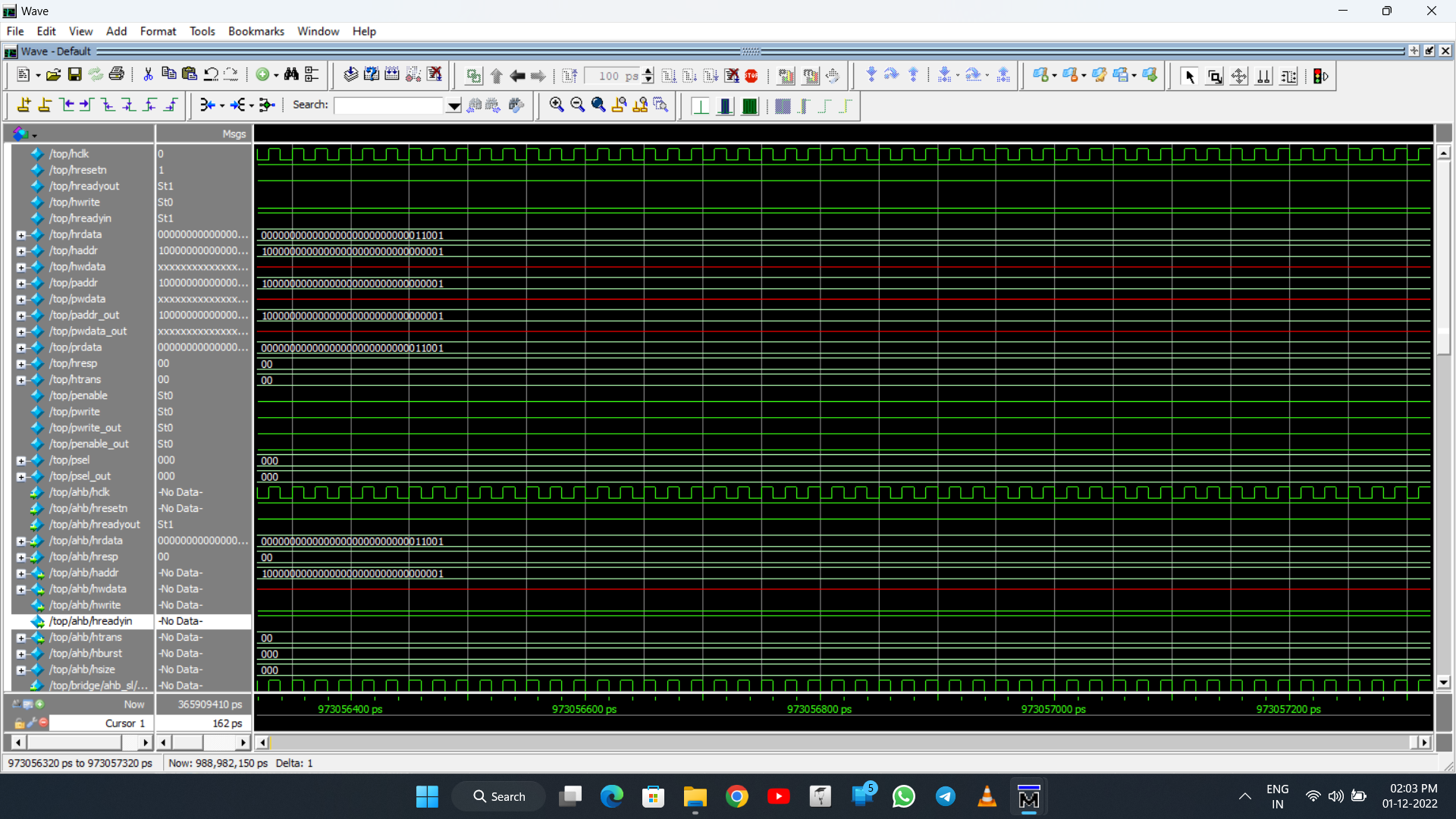
end

endmodule

**SINGLE WRITE OPERATION**



**SINGLE READ OPERATION**

****

**AHB-APB BRIDGE INTERFACE ARCHITECTURE**

