PROJECT REPORT ON AHB2APB BRIDGE

MAY 2023 – JUNE 2023

MAVEN SILICON DESIGN INTERNSHIP

(DI-33)

Submitted By: -

SUBHAM SUNDAR MOHARANA

Admission No.: - MS/23-24/0424

Veer Surendra Sai University of Technology, Burla

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
```

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;
//present state logic
always @(posedge hclk)
begin
if (!hresetn)
PS<=ST_IDLE;
else
PS<=NS;
end
//next state
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
//temporary output logic
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: 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
ST_READ: begin
penable_temp=1;
hr_readyout_temp=1;
end
ST_RENABLE: 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
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
```

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 [2:0] temp_selx;
wire hwritereg,hwritereg_1;
wire valid;
ahb_slave_interface
a1(hclk,hresetn,hwrite,hready_in,htrans,haddr,hwdata,valid,hwritereg,hw
ritereg_1,
hresp,temp_selx,haddr_1,haddr_2,hwdata_1,hwdata_2,hrdata,prdata);
apb_controller
a2(hclk,hresetn,hwritereg,hwrite,valid,haddr,hwdata,hwdata_1,hwdata_2,h
addr_1,haddr_2,prdat
a,
temp_selx,penable,pwrite,hr_readyout,psel,paddr,pwdata);
endmodule
```

AHB MASTER INTERFACE: -

```
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; //single,4,16,...
reg [2:0] hsize; //size 8bit,16bit..
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
end
endtask
endmodule
```

APB INTERFACE: -

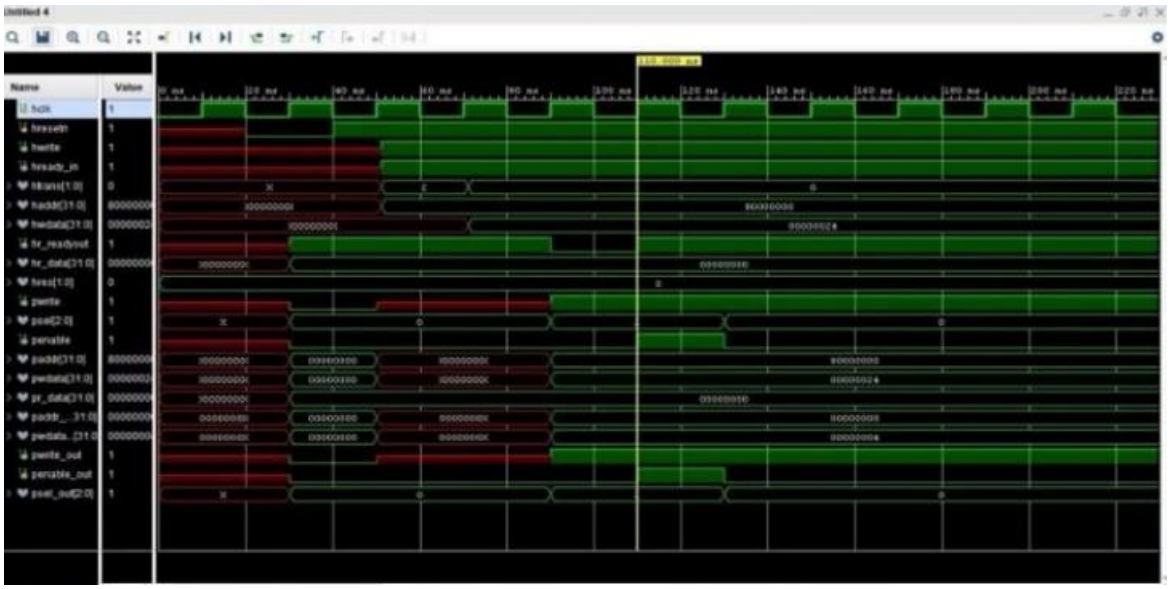
```
module apb_interface (input pwrite,penable,input [2:0] pselx, input
[31:0] paddr,pwdata,
output pwrite_out,penable_out,output [2:0] psel_out,output [31:0]
paddr_out,pwdata_out,output
reg [31:0] prdata);
assign pwrite_out=pwrite;
assign penable_out=penable;
assign psel_out=pselx;
assign pwdata_out=pwdata;
assign paddr_out=paddr;
always @(*)
begin
if(!pwrite==1 &&penable)
prdata=8'd25;
end
endmodul
```

TOP TEST BENCH(TB): -

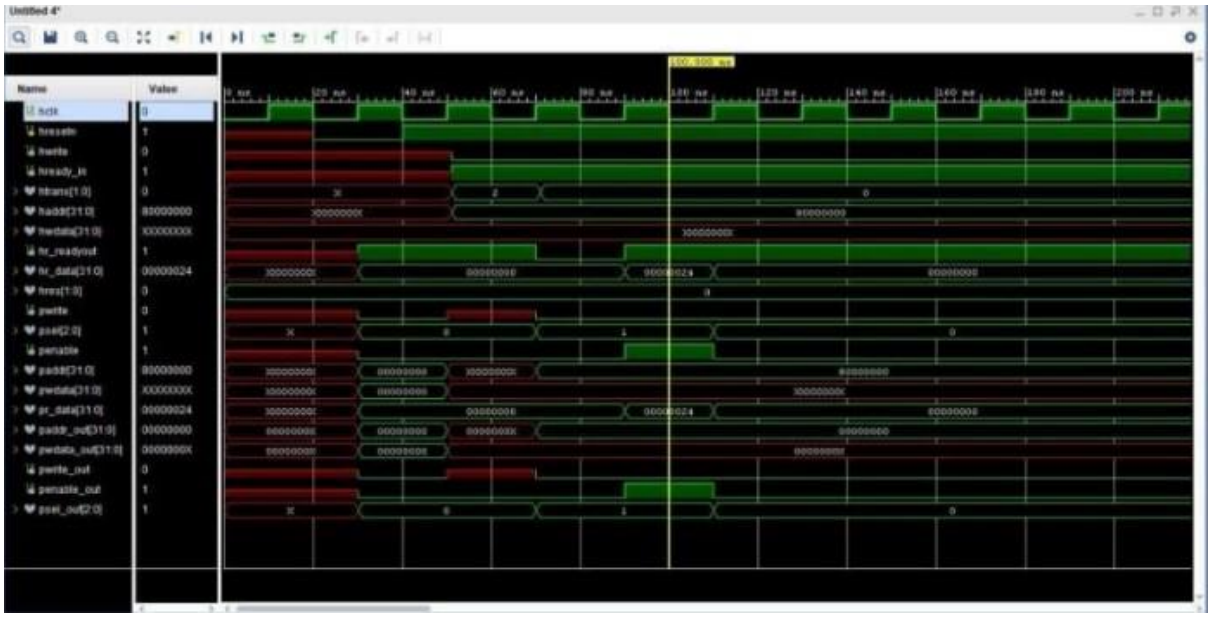
```
module top_tb();
reg hclk;
reg hresetn;
wire [31:0]
hrdata,haddr,hwdata,paddr,pwdata,paddr_out,pwdata_out,prdata;
wire hwrite, hreadyin;
wire [1:0] htrans;
wire [1:0] hresp = 0;
wire penable,pwrite,hreadyout,pwrite_out,penable_out;
wire [2:0] psel,psel_out;
ahb_master
ahb(hclk,hresetn,hreadyout,hrdata,hresp,haddr,hwdata,hwrite,hreadyin,ht
rans);
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,pa
ddr_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();
//ahb.burst_4_incr_write();
end
endmodul
```

SIMULATION WAVEFORM RESULTS: -

1. Single Write



2. Single Read



3. Burst-4 Increment Write