EE258 Verilog assignment

190002055 Sai Pranavi.K

```
timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 29.07.2021 15:07:39
// Design Name:
// Module Name: Fifo
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


// This is linear queue / FIFO
// The queue length 10
// The data width is also 8 bits
module fifo(DATAOUT, full,rst, empty, clock,w, r, DATAIN);
  output reg [7:0] DATAOUT;
  output full, empty;
  input [7:0] DATAIN;//
```

```
input clock, w, r,rst; // w is write (for storing the data) and r is read (for getting the data)

reg [3:0] wptr=0, rptr=0; // pointers tracking the stack
reg [7:0] memory [0:9]; // the stack is 8 bit wide and 8 locations in size
reg [3:0] count=0;//fifo counting tells us if it is full or empty

assign full = (count==10);
assign empty = (count ==0);

always @(posedge clock)
begin:write
 if(w && !full) //ram is not full
 begin
 memory[wptr] <=DATAIN;
 end
 else if (r & w) //read=1 implies that data place is taken .We can write data then.
   begin
    memory[wptr] <= DATAIN;
   end
end

always @(posedge clock)
begin:read
if(r && !empty)//We can get output {read} only if it is not empty
begin
DATAOUT=memory[rptr];
end
else if(r && w)//if write=1,data is stored =>read=1 means data can be read
begin
DATAOUT=memory[rptr];
end
end

always @(posedge clock)
```

```
begin:pointer
if(rst)begin//reset makes read and write pointer '0'
rptr<=0;
wptr<=0;
end
else begin
wptr<=(w && !full)||(w && r)?wptr+1:wptr; //if any of the "write" conditions is true:
if(wptr==10)
begin
wptr=0;
end
  rptr<=(r && !empty)||(w && r)?rptr+1:rptr;//if any of the "read"conditions is true
if(rptr==10)
begin
rptr=0;
end
end
end


always @(posedge clock)
begin:counter
if(rst)
begin
count<=0;
end
else begin
case({w,r})
2'b00:count<=count;// write=read=0 implies no change in count
2'b01:count<=(count==0)?count:count-1;//read=1 and write =0 implies taken==count decreases
2'b10:count<=(count==10)?count:count+1;//read=0 and write =1 implies taken==count increases
2'b11:count<=count;//read=write=1 implies given and taken==>no change
default:count<=count;
endcase
end
```

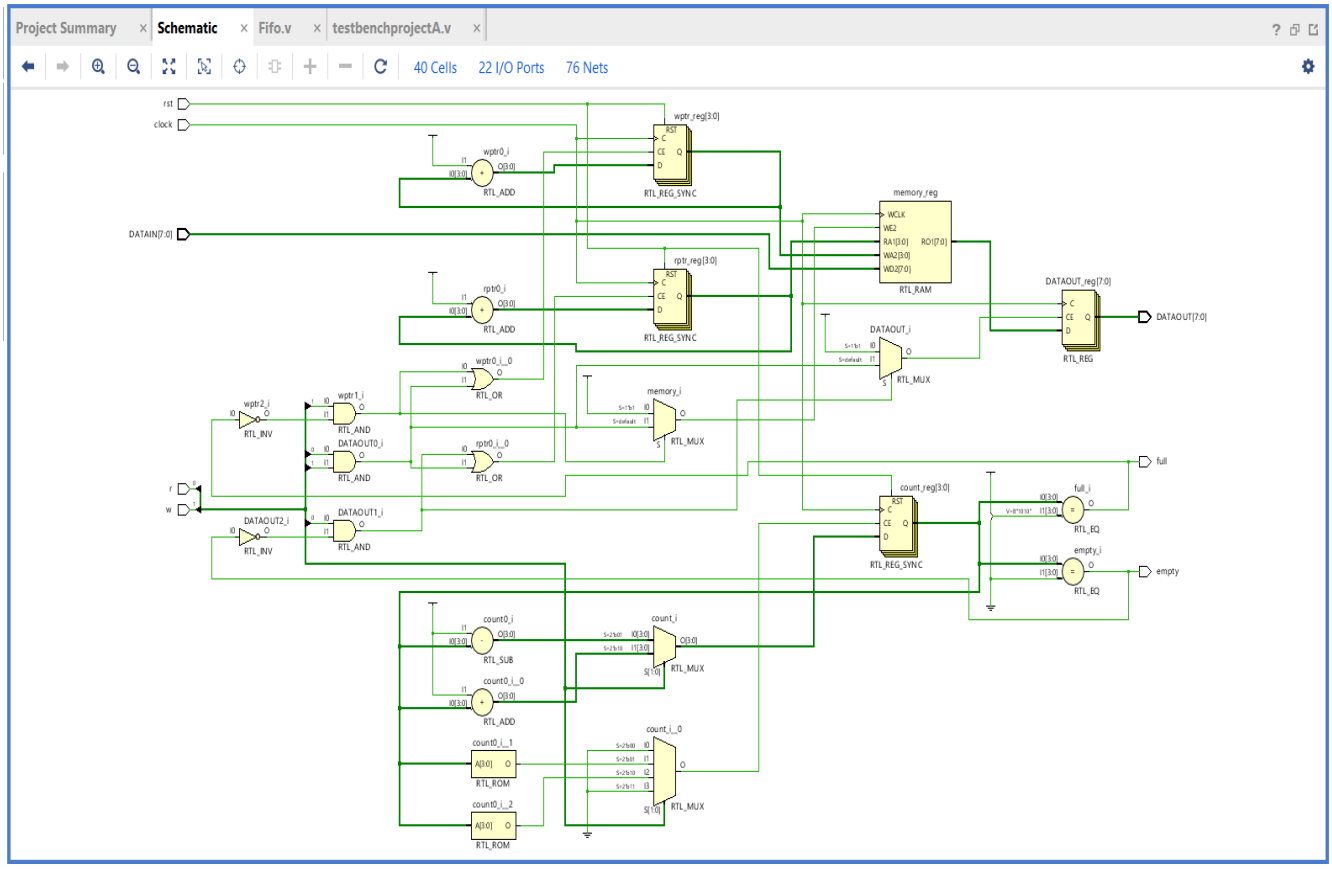
```
 end


endmodule
```

Test bench

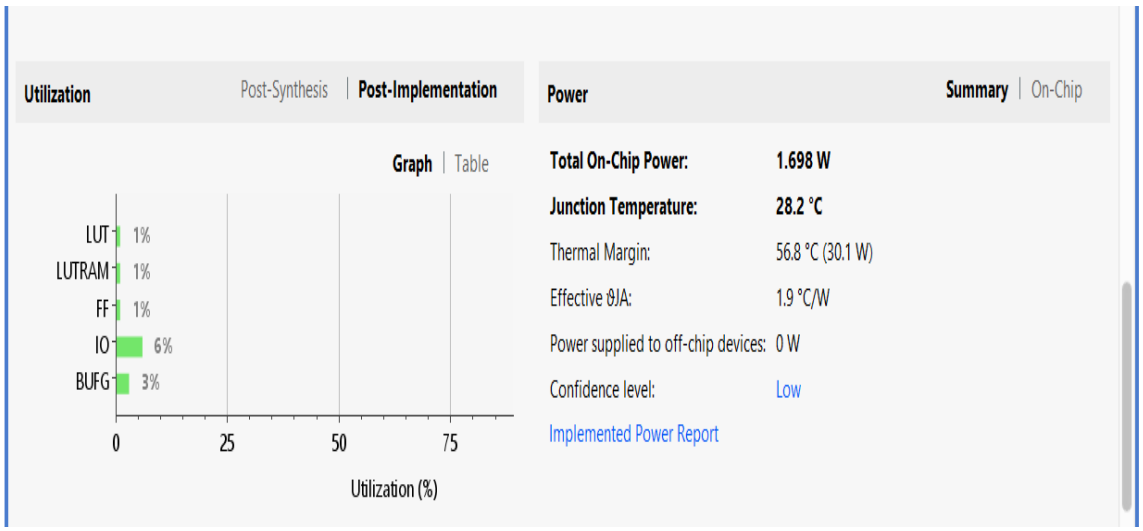
```
`timescale 1ns / 1ps
module testbench();
reg clk, rst, wr_en, rd_en ;
reg[7:0] data_in;
reg[7:0] tempdata;
wire [7:0] data_out;
wire [10:0] fifo_counter;
wire count_empty,count_full;



fifo_reg ff( .clk(clk), .rst(rst), .data_in(data_in), .data_out(data_out),
        .wr_en(wr_en), .rd_en(rd_en), .count_empty(count_empty),
        .count_full(count_full), .fifo_counter(fifo_counter) );

initial begin


  clk = 1'b0;
  rst = 1'b1;
  rd_en = 1'b0;
  wr_en = 1'b0;
  tempdata = 16'h0;
  data_in = 16'h0;



 #100;
  wr_en = 1'b1;
```

```
 rst=1'b1;
  #20
    rst=1'b0;
  wr_en = 1'b1;
  data_in = 16'h0;
  #20;
  data_in  = 16'h1;
  #20;
  data_in  = 16'h2;
  #20;
  data_in  = 16'h3;

  #20;
  data_in  = 16'h4;
  #20;
  data_in  = 16'h5;
  #20;
  data_in  = 16'h6;
  #20;
  data_in  = 16'h7;
  #20;
  data_in  = 16'h8;
  #20;
  data_in  = 16'h9;
  #20;
  wr_en = 1'b0;
  rd_en = 1'b1;
 end
 always #10 clk = ~clk;
endmodule
```

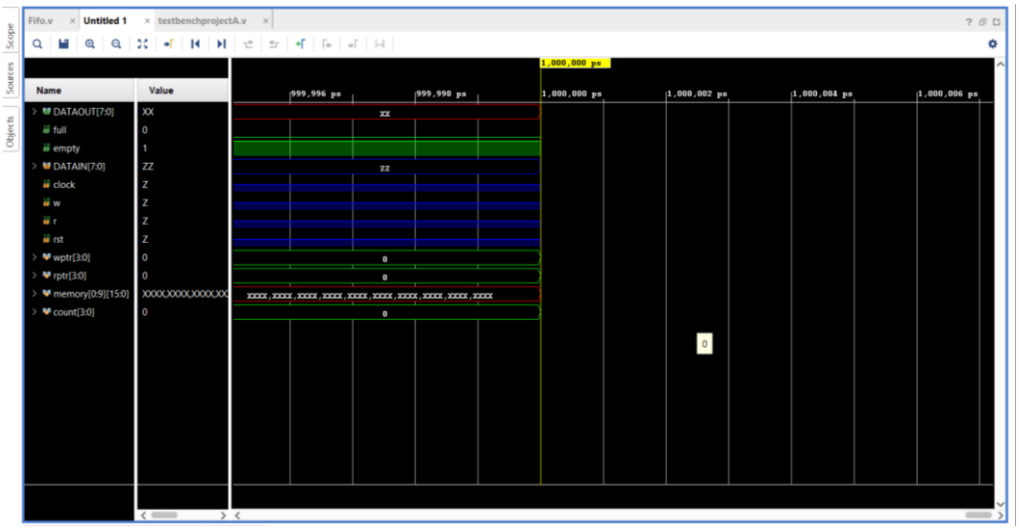
RTL Analysis



Project Summary



Behavioural simulation


Fifo.v
Untitled 1
testbenchprojectA.v
Scope
Sources
Objects
Name
Value
DATAOUT[7:0]
XX
full
0
empty
1
DATAIN[7:0]
ZZ
clock
Z
w
Z
r
Z
rst
Z
wptr[3:0]
0
rptr[3:0]
0
memory[0:9][15:0]
count[3:0]
0
999,996 ps
999,998 ps
1,000,000 ps
1,000,002 ps
1,000,004 ps
1,000,006 ps
1,000,000 ps
XX
ZZ
0
0
0
0