15th Day

Asynchronous FIFO

Design

```
module async_fifo(input clk_write,clk_read,rst,wr,rd,[7:0]din,output reg [7:0]dout,output full,empty );


reg [7:0]mem[0:7];


reg [2:0]wr_ptr;
reg [2:0]rd_ptr;
reg [2:0]count;


always @(posedge clk_write)
begin
   if(rst)
   begin
   wr_ptr<=0;
   count<=0;
   for(integer i=0;i<8;i=i+1)
    mem[i]<=0;
   end


   else if(wr && !full)
   begin


      mem[wr_ptr]<=din;
      wr_ptr<=(wr_ptr==7)?0:wr_ptr+1;
      count<=count+1;
```

```
    end

end

always @(posedge clk_read)
begin
    if(rst)
    begin
    rd_ptr<=0;
    count<=0;
    dout<=0;
    end

    else if(rd && !empty)
    begin

        dout<=mem[rd_ptr];
        rd_ptr<=(rd_ptr==7)?0:rd_ptr+1;
        count<=count-1;
    end

end

        assign empty=(count==0);
        assign full=(count==7);

endmodule
```

Testbench

```
module async_fifo_tb();
reg clk_wr,clk_rd,rst,wr,rd;
reg [7:0]din;
wire full,empty;
wire [7:0]dout;

async_fifo dut(clk_wr,clk_rd,rst,wr,rd,din,dout,full,empty);

always #5 clk_wr=~clk_wr;
always #7 clk_rd=~clk_rd;

task clk_rst;
   begin
   clk_wr=1'b0;
   clk_rd=1'b0;
   rst=1'b1;
   #15;
   rst=1'b0;
   end
endtask

task wr_stimulus;
   begin
   wr=1'b1;
   repeat(8)
   begin
   din=$random;
   #10;
   end
```

```
   wr=1'b0;
   end
endtask

task rd_stimulus;
    begin
    rd=1'b1;
    #150;
    rd=1'b0;
    end
endtask

initial
begin
clk_rst;
wr_stimulus;
rd_stimulus;
#100;$finish;
end

endmodule
```

Simulation