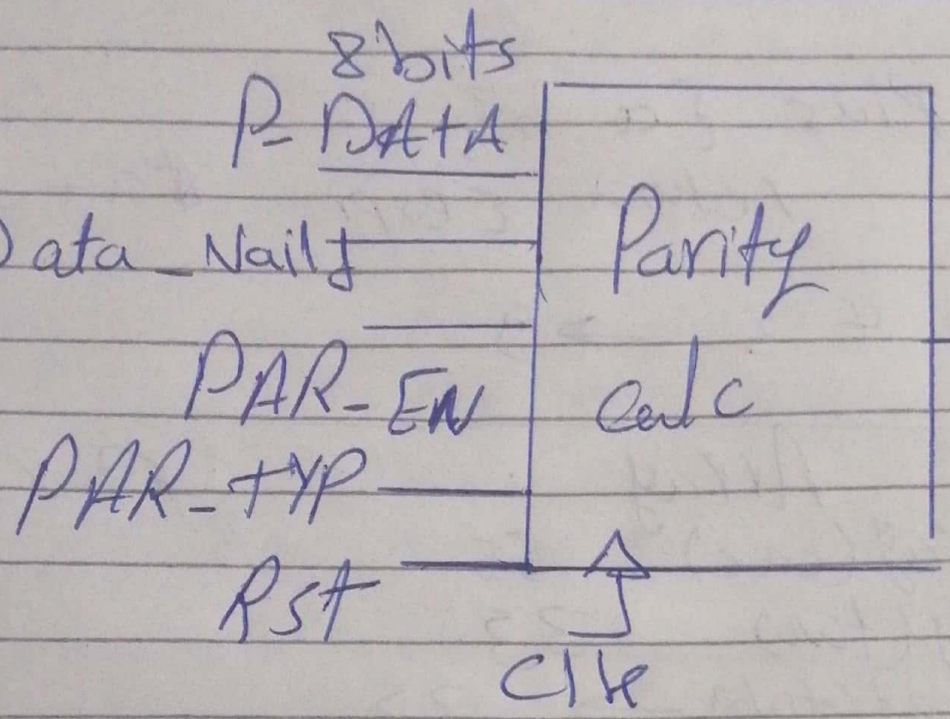
Parity Bit Calculator

in Verilog

NABIL YOUSSEF



even Parity 0 → even "1"

odd Parity 0 → odd "1"

^P_Data XOR for bits of data stream

1 → odd ones , 0 → even ones

Parity Calculator

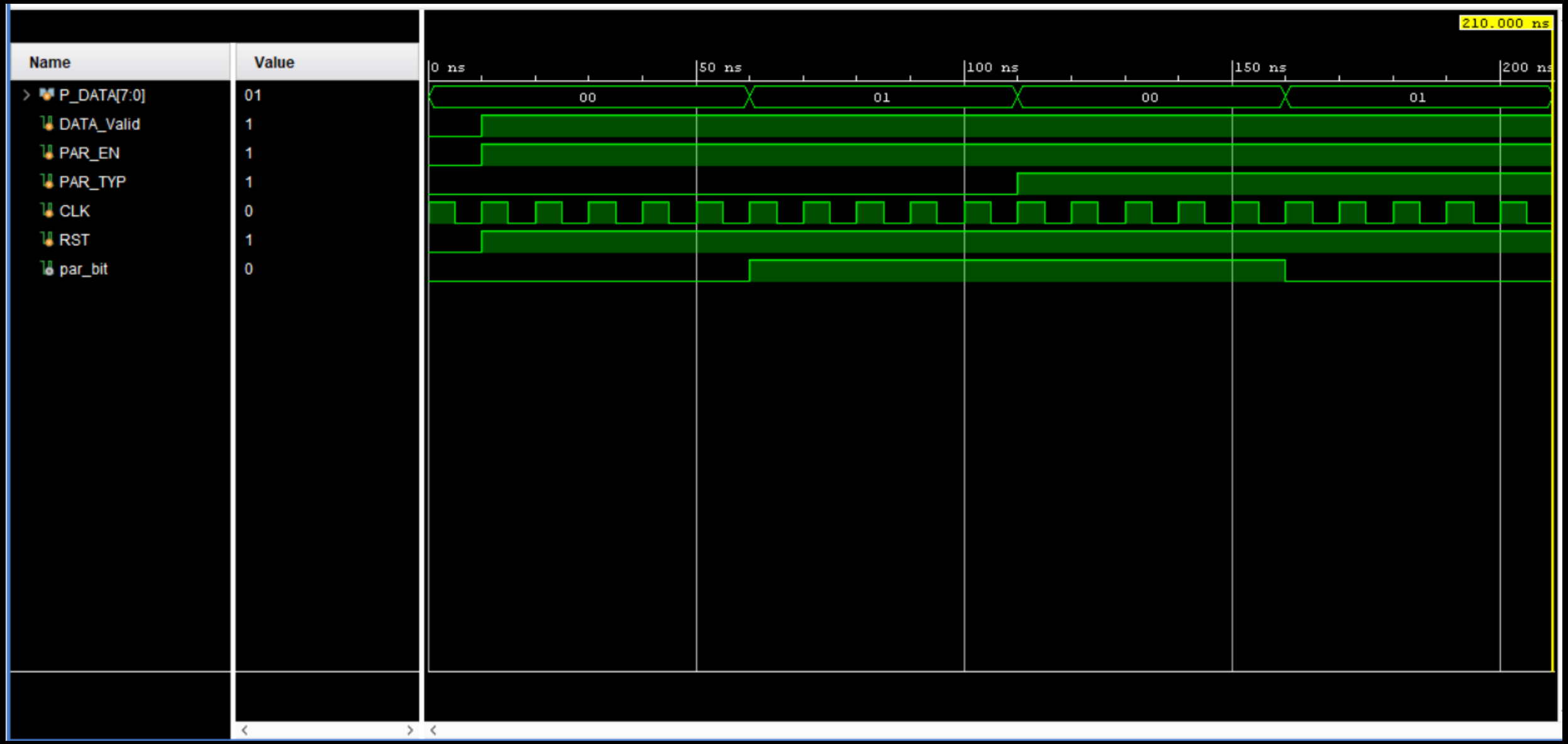
- In even parity bit, the parity bit is ‘0’ if there are even number of 1s in the data stream and the parity bit is ‘1’ if there are odd number of 1s in the data stream.
- In odd parity bit, the parity bit is ‘1’ if there are even number of 1s in the data stream and the parity bit is ‘0’ if there are odd number of 1s in the data stream. Let us discuss both even and odd parity generators.
- ^P_DATA is a bitwise XOR operation between all the bits in the data stream. The result is 1 if the number of 1 bits in the data stream is odd, and 0 if the number of 1 bits is even.

Verilog Code

```
`timescale 1ns / 1ps
module Parity_Calc(
    input wire [7:0] P_DATA,
    input wire DATA_Valid,
    input wire PAR_EN,
    input wire PAR_TYP, // '0' even, '1' odd
    input CLK,RST,
    output reg par_bit);
 always @(posedge CLK or negedge RST) begin
    if (!RST)
        par_bit <= 1'd0;
    else if (DATA_Valid && PAR_EN) begin // if both are onez
        if (( PAR_TYP && (^P_DATA)) || (~PAR_TYP && (~^P_DATA))) begin
        // ^P_DATA is 1 >> odd Ones
        //            0 >> even ones
        par_bit <= 1'd0;
        end
        else par_bit <= 1'd1;
    end
    else par_bit <= par_bit ;
 end
endmodule
```

TestBench

```
`timescale 1ns/1ps

module Parity_Calc_TB;

    // Inputs
    reg [7:0] P_DATA;
    reg DATA_Valid;
    reg PAR_EN;
    reg PAR_TYP;
    reg CLK;
    reg RST;

    // Outputs
    wire par_bit;

    Parity_Calc uut (
    .P_DATA(P_DATA),
    .DATA_Valid(DATA_Valid),
    .PAR_EN(PAR_EN),
    .PAR_TYP(PAR_TYP),
    .CLK(CLK),
    .RST(RST),
    .par_bit(par_bit)
    );
initial begin
    // Initialize inputs
    P_DATA = 8'b00000000;
    DATA_Valid = 1'b0;
    PAR_EN = 1'b0;
    PAR_TYP = 1'b0;
    CLK = 1'b1;
    RST = 1'b0;
    #10;
    RST = 1'b1;
    P_DATA = 8'b00000000;
    DATA_Valid = 1'b1;
    PAR_EN = 1'b1;
    PAR_TYP = 1'b0;
    #50;
    P_DATA = 8'b00000001;
    #50;
    PAR_TYP = 1'b1;
    P_DATA = 8'b00000000;
    #50;
    P_DATA = 8'b00000001;
    #50;
    $finish;
end

    always #5 CLK = ~CLK;

endmodule
```


Name
Value
P_DATA[7:0]
01
DATA_Valid
1
PAR_EN
1
PAR_TYP
1
CLK
0
RST
1
par_bit
0
210.000 ns
0 ns
50 ns
100 ns
150 ns
200 ns
00
01
00
01

GitHub