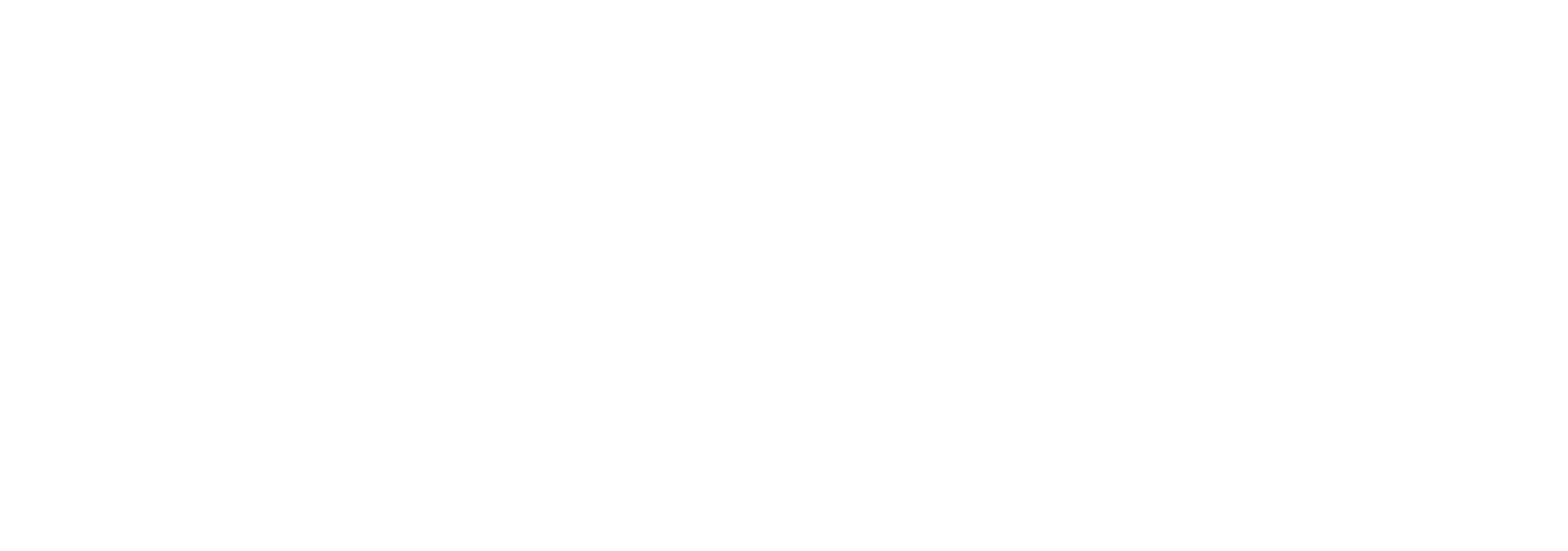
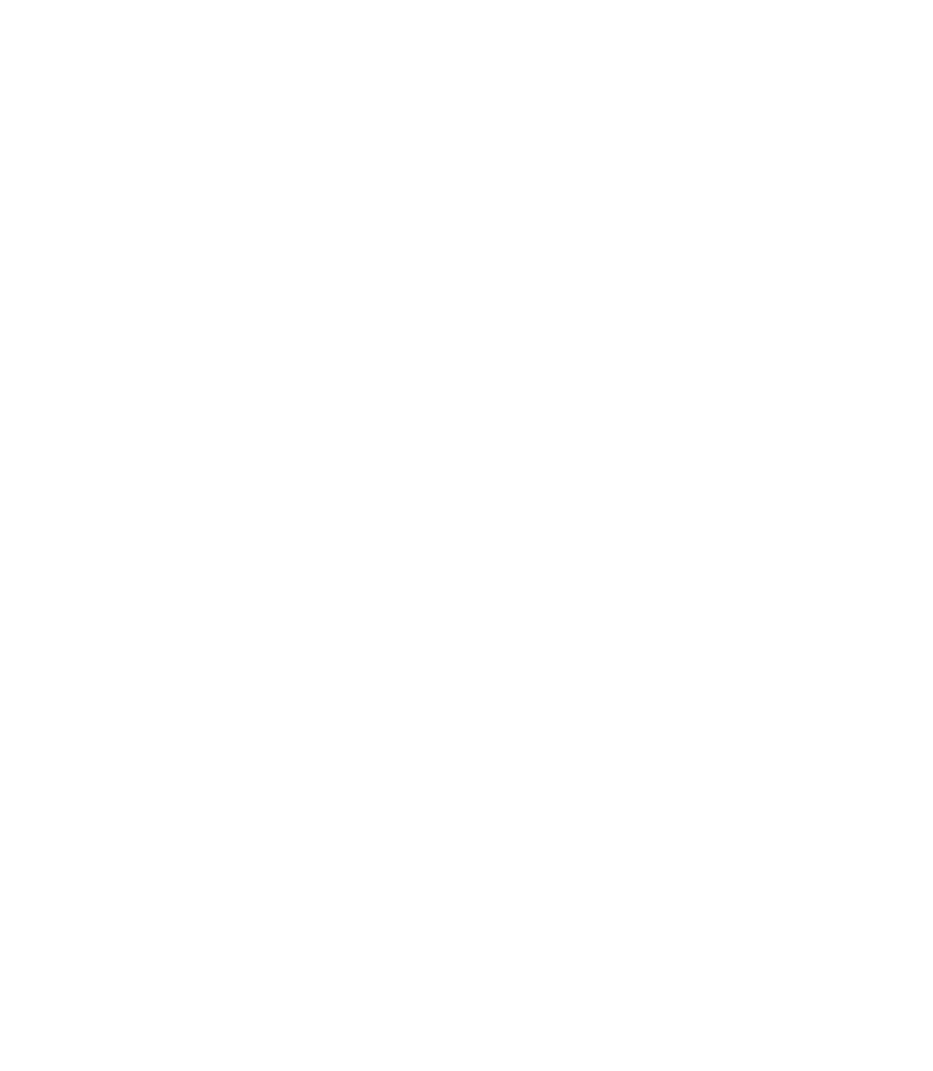
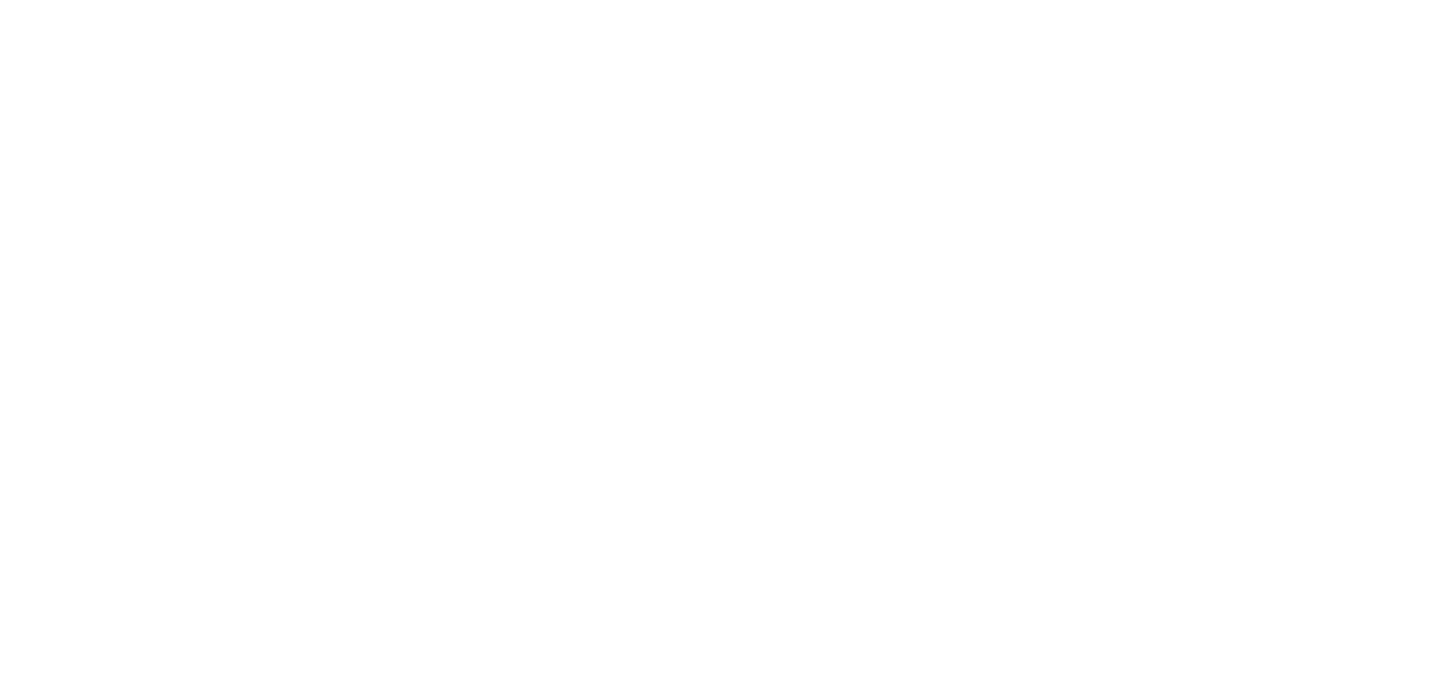
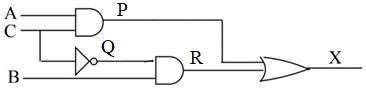
**Hardware runs**

**Run 1: AND-OR implementation**

**Diagram**



**Truth Table**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A | B | C | P | Q | R | X |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 |

**Q1:** For the above circuit diagram fill in the following details

**A: No. of AND gates used: 2 No. of AND gate IC used:**

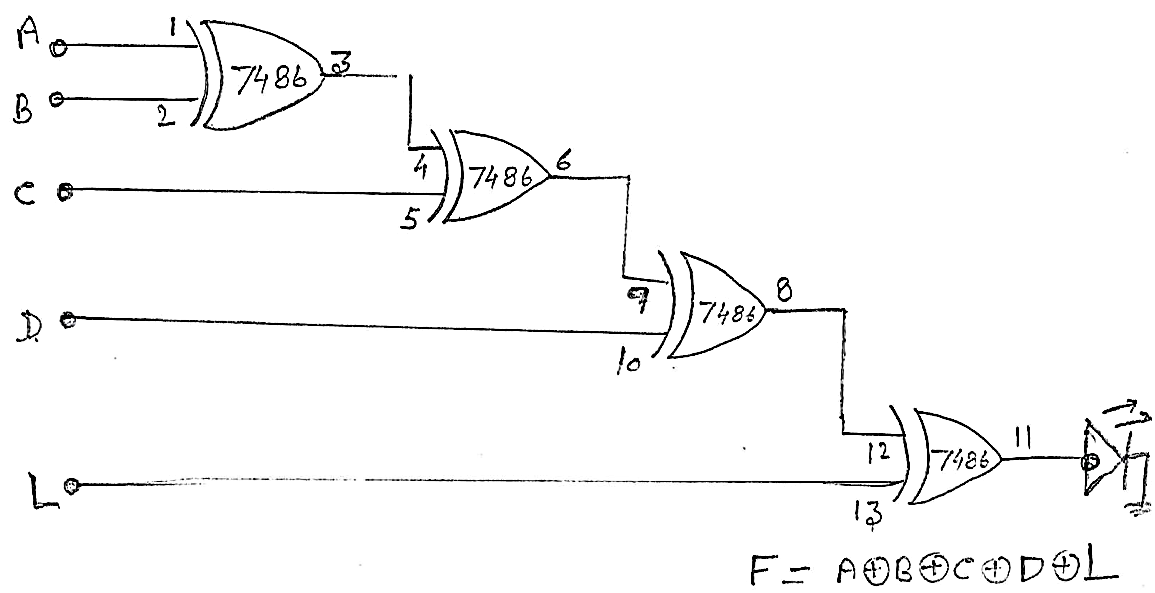
**No. of NOT gates used: 1 No. of NOT gate IC used:**

**No. of OR gates used: 1 No. of OR gate IC used:**

**Total No. of ICs used:**

**Run 2: Parity generator**

**Diagram**

****

**Q2:** How many XOR gates are used?

**A:** 4

**Q3:** How many 7486 IC are used? Can the circuit be implemented using only **one** 7486 IC?

**A:**

**Q4:** Take any six-input combinations of your choice and complete the below table.

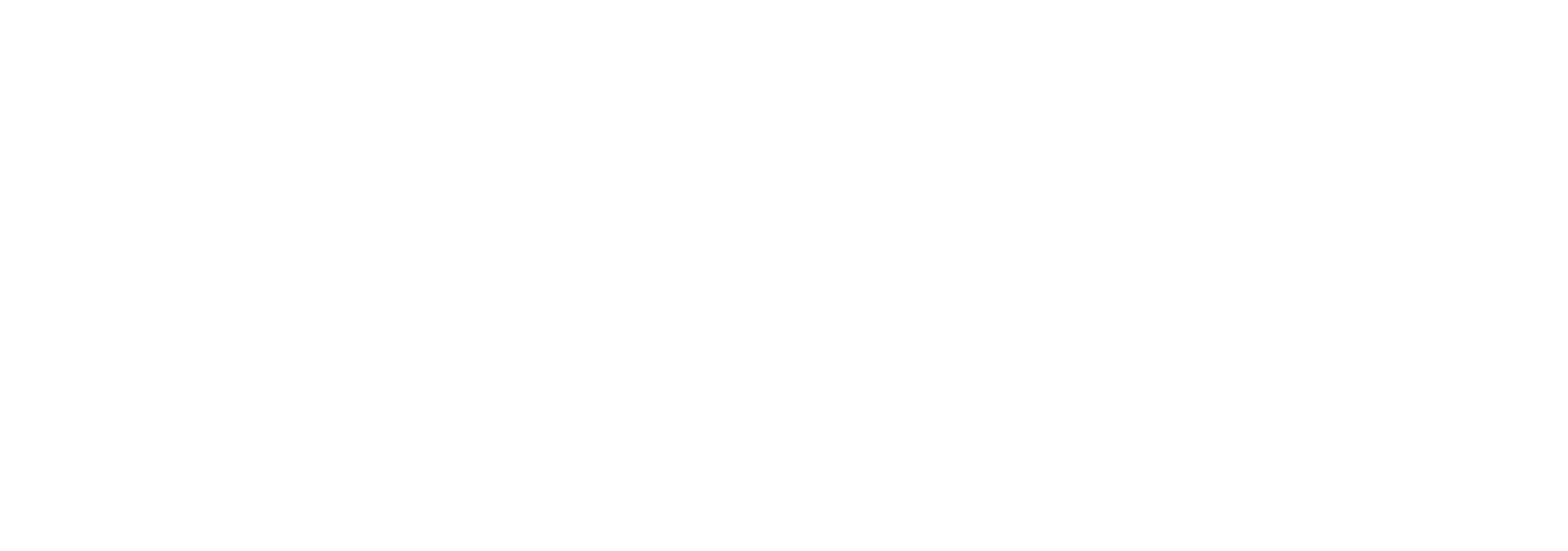
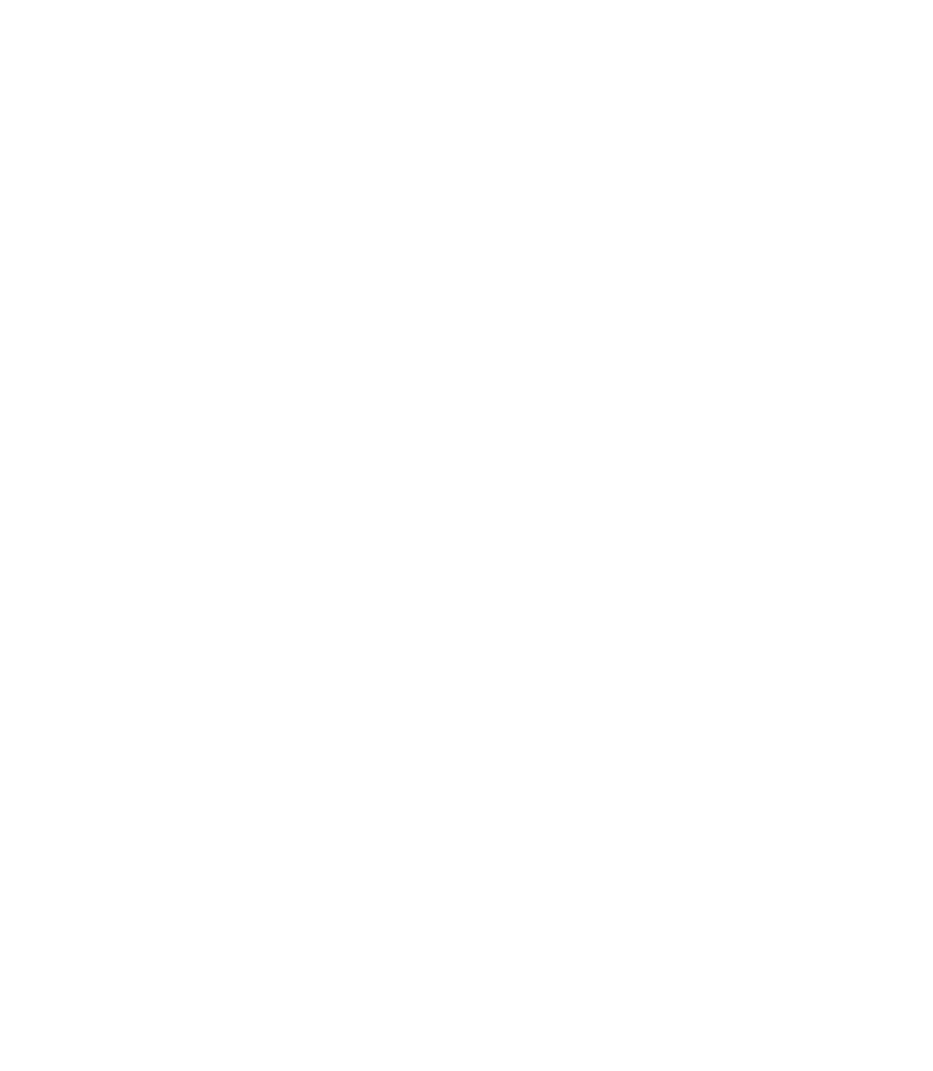
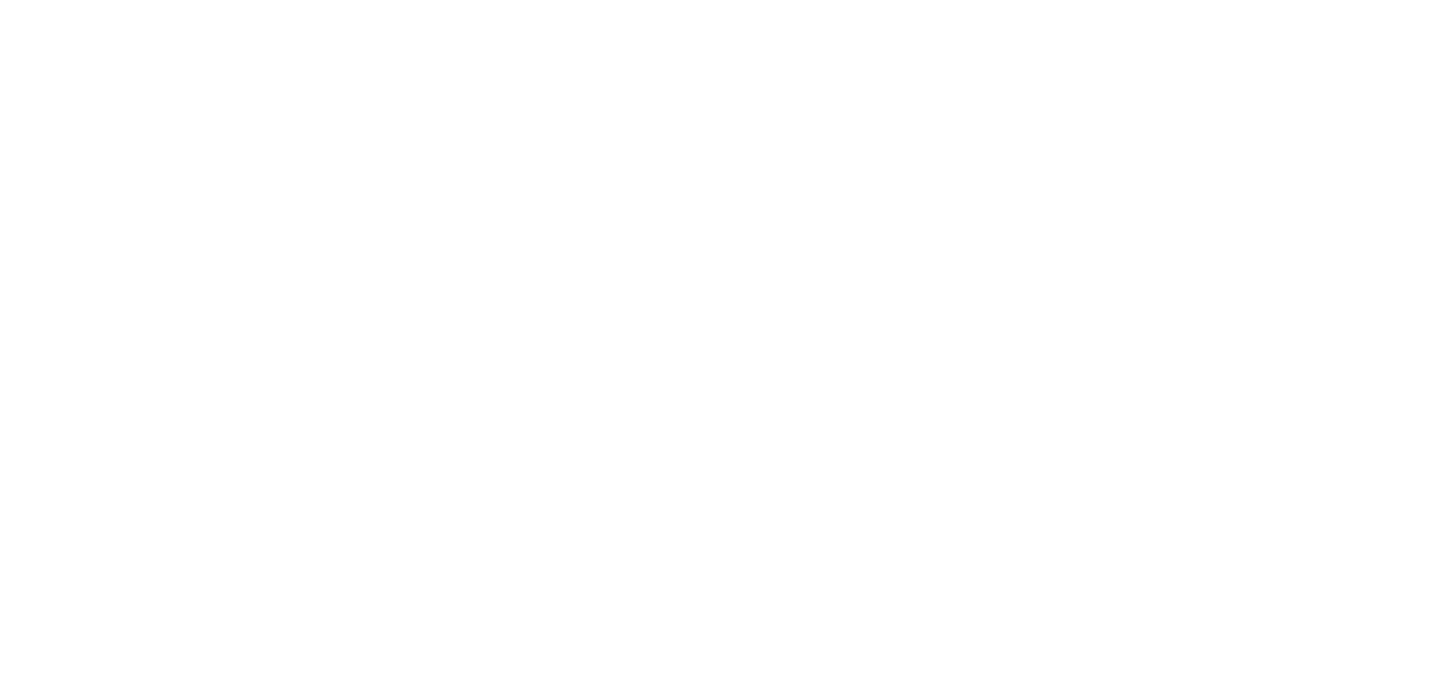
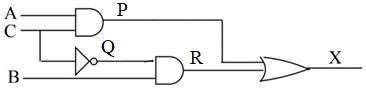
**Truth Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **L** | **P** |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |

**Software runs**

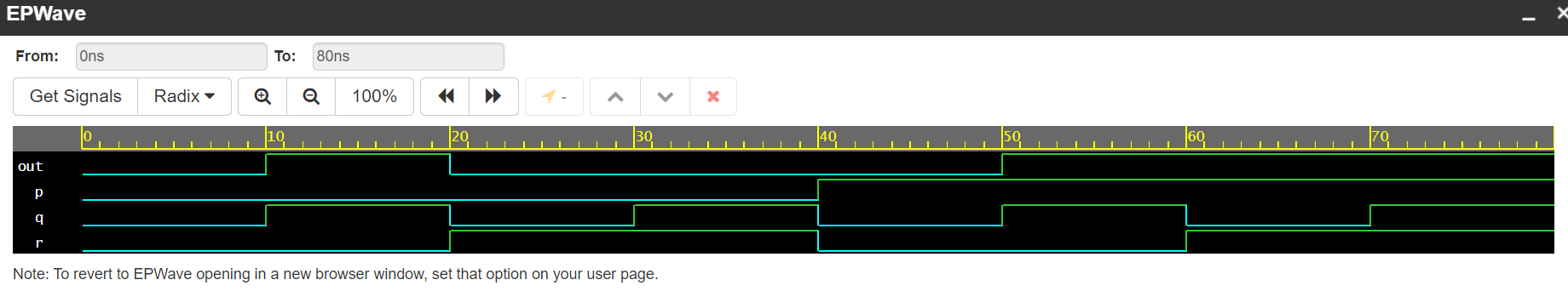
**Run 3: Circuit implementation**

1. Implement the below circuit using Gate level modeling, write the code as well as its testbench.



**Q5:** Paste the Image of your **Simvision** window where you get the waveforms for the above code.

**A:** [**https://www.edaplayground.com/x/T28c**](https://www.edaplayground.com/x/T28c)

****

**Run 4: Error detection and error correction codes**

**Q6:** Write Verilog code and testbench for generating even parity bit for 4 bit binary number. (Hint: you can use the structure of run-2 of this experiment also or y = A^B^C^D).

**A: Verilog Code-** [**https://www.edaplayground.com/x/nWEx**](https://www.edaplayground.com/x/nWEx)

module parity\_gen(data, par\_data);

input [3:0] data;

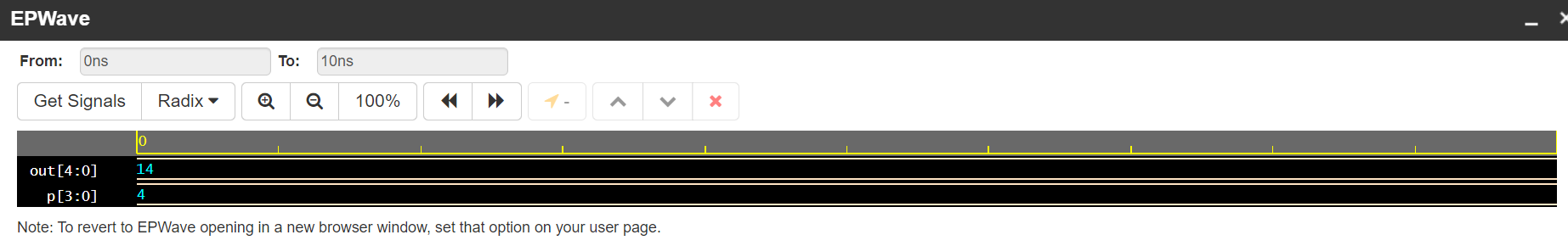
output [4:0] par\_data;

assign par\_data[3:0]=data[3:0];

assign par\_data[4]=data[0]^data[1]^data[2]^data[3];

endmodule

**Q7:** Paste the Image of your **Simvision** window where you get the waveforms for the above code.

**A:**

**Q8:** Write Verilog code and testbench for generating even parity Hamming code for 4-bit data. (Hint: P0 = D2^D1^D0, P1 = D3^D1^D0, P2 = D3^D2^D0)

**A: Verilog Code-** [**https://www.edaplayground.com/x/EeJB**](https://www.edaplayground.com/x/EeJB)

module hamming\_code(data, out);

input [3:0] data;

output [6:0] out;

wire p1,p2,p3;

assign p1=(data[0]^data[1]^data[3]);

assign p2=(data[0]^data[2]^data[3]);

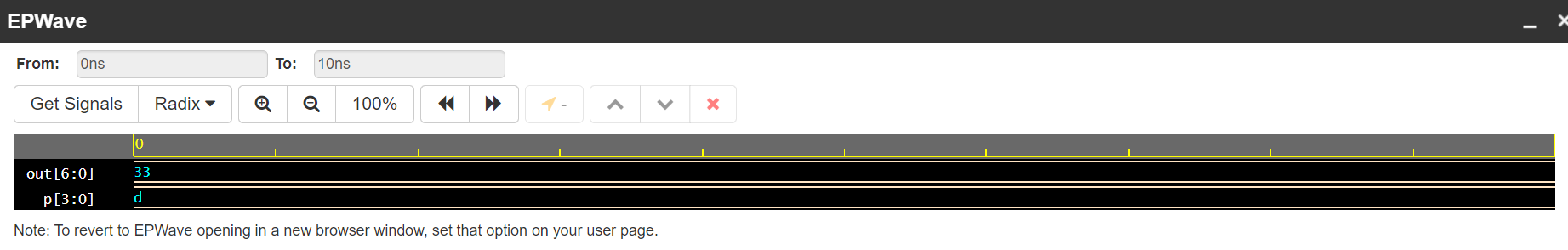
assign p3=(data[1]^data[2]^data[3]);

assign out={p1,p2,data[0],p3,data[1],data[2],data[3]};

endmodule

**Q9:** Paste the Image of your **Simvision** window where you get the waveforms for the above code.

**A:**

****