**FIFO**

**TEST BENCH:**

module tb\_fifo\_gray\_operations;

// Testbench signals

reg clk;

reg reset;

reg wr\_en;

reg rd\_en;

reg [3:0] binary\_in;

wire [3:0] fifo\_data\_out;

wire and\_result;

wire or\_result;

// Clock generation

always #5 clk = ~clk;

// Instantiate the FIFO with Gray code and operations module

fifo\_with\_gray\_operations uut (

.clk(clk),

.reset(reset),

.wr\_en(wr\_en),

.rd\_en(rd\_en),

.binary\_in(binary\_in),

.fifo\_data\_out(fifo\_data\_out),

.and\_result(and\_result),

.or\_result(or\_result)

);

initial begin

// Initialize signals

clk = 0;

reset = 1;

wr\_en = 0;

rd\_en = 0;

binary\_in = 4'b0000;

// Release reset

#10 reset = 0;

// Test 1: Write binary value 1010 (0xA) into FIFO

#10 binary\_in = 4'b1010;

wr\_en = 1;

#10 wr\_en = 0;

// Test 2: Write binary value 1100 (0xC) into FIFO

#10 binary\_in = 4'b1100;

wr\_en = 1;

#10 wr\_en = 0;

// Read first data from FIFO and perform operations

#10 rd\_en = 1;

#10 rd\_en = 0;

// Read second data from FIFO and perform operations

#10 rd\_en = 1;

#10 rd\_en = 0;

// End simulation

#20 $finish;

end

endmodule