**EXP2:BOOLEAN FUNCTION MINIMIZATION**

**BOOLEAN FUNCTION MINIMIZATION**

**AIM:**

To implement the given logic function verify its operation in Quartus using Verilog programming.

F1= A’B’C’D’+AC’D’+B’CD’+A’BCD+BC’D

F2=xy’z+x’y’z+w’xy+wx’y+wxy

**Equipment Required:**

Hardware – PCs, Cyclone II , USB flasher

**Software – Quartus prime**

**Theory :**

A combinational circuit is a circuit in which the output depends on the present combination of inputs. Combinational circuits are made up of logic gates. The output of each logic gate is determined by its logic function. Combinational circuits can be made using various logic gates, such as AND gates, OR gates, and NOT gates.

**Procedure:**

1. Type the program in Quartus software.
2. Compile and run the program.
3. Generate the RTL schematic and save the logic diagram.
4. Create nodes for inputs and outputs to generate the timing diagram.
5. For different input combinations generate the timing diagram.

**Program:**

module booleanmin(a, b, c, d, w, x, y, z, f1, f2);

input a, b, c, d, w, x, y, z;

output f1, f2;

wire not\_a, not\_b, not\_c, not\_d, not\_w, not\_x, not\_y, not\_z;

// Generate inverted signals

assign not\_a = ~a;

assign not\_b = ~b;

assign not\_c = ~c;

assign not\_d = ~d;

assign not\_w = ~w;

assign not\_x = ~x;

assign not\_y = ~y;

assign not\_z = ~z;

// Intermediate wires for logic

wire p = not\_a & not\_b & not\_d;

wire q = a & not\_b & c & not\_d;

wire r = not\_a & b & c & not\_c;

wire s = a & not\_b & not\_c & not\_d;

wire t = not\_w & not\_x & not\_y;

wire u = not\_w & x & not\_y;

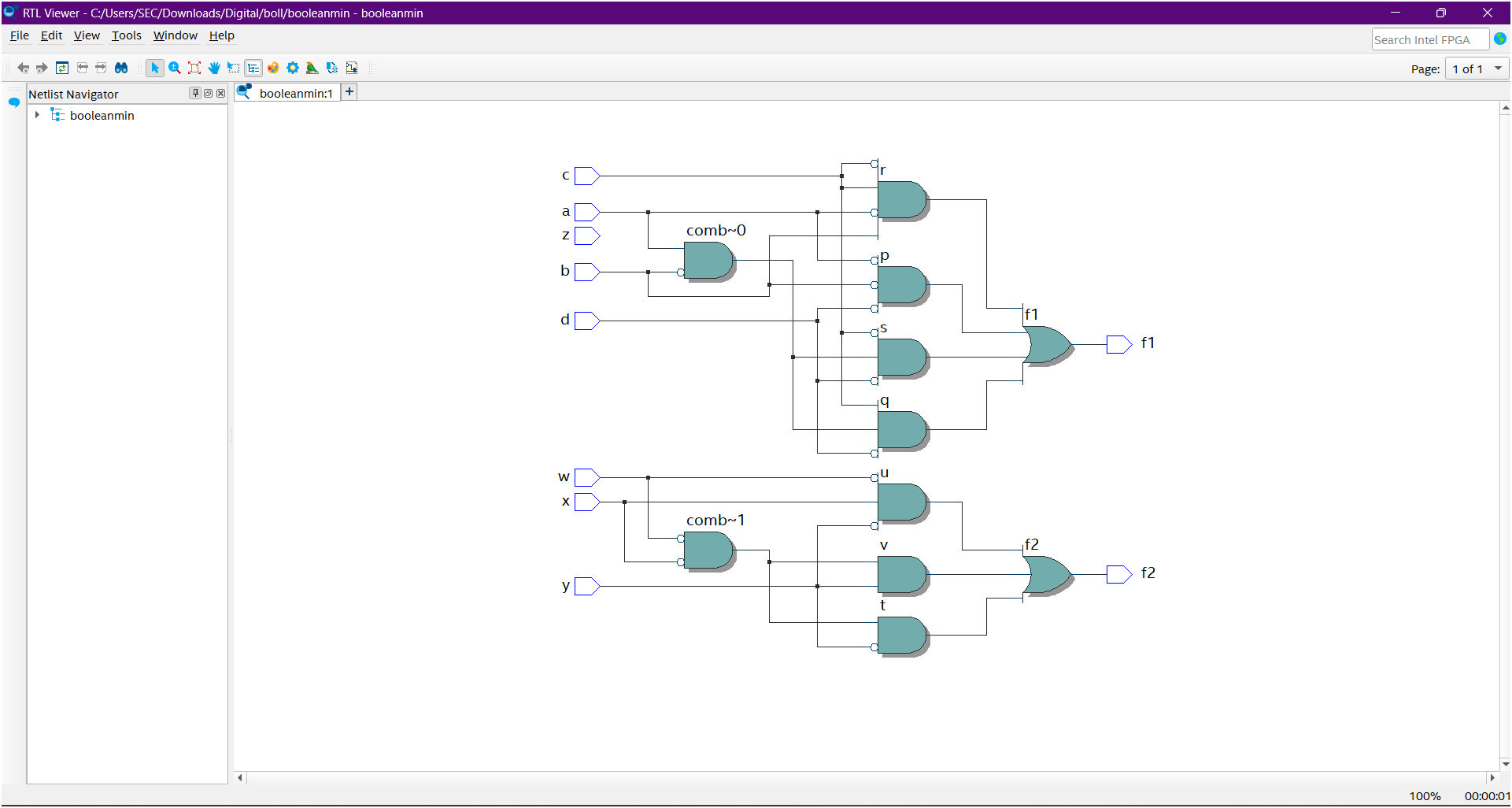
wire v = not\_w & not\_x & y;

// Output logic

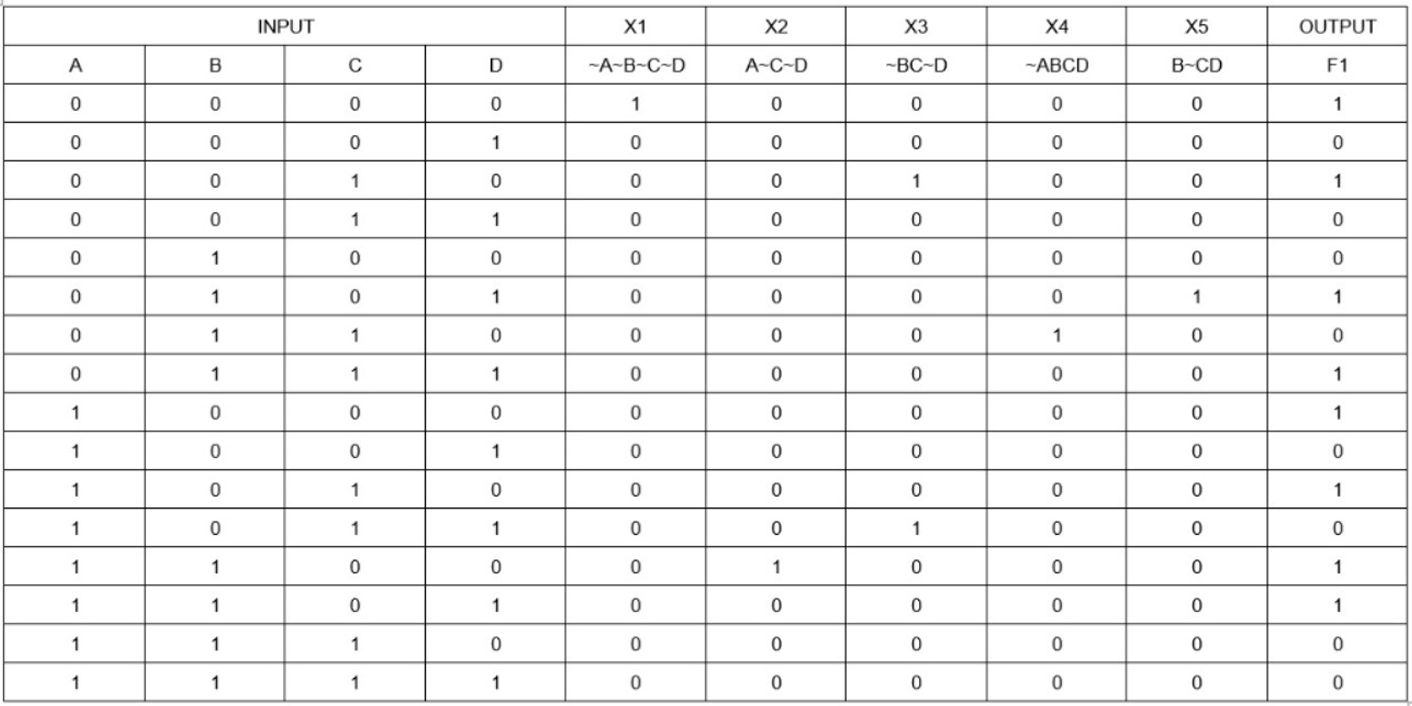
assign f1 = p | q | r | s;

assign f2 = t | u | v;

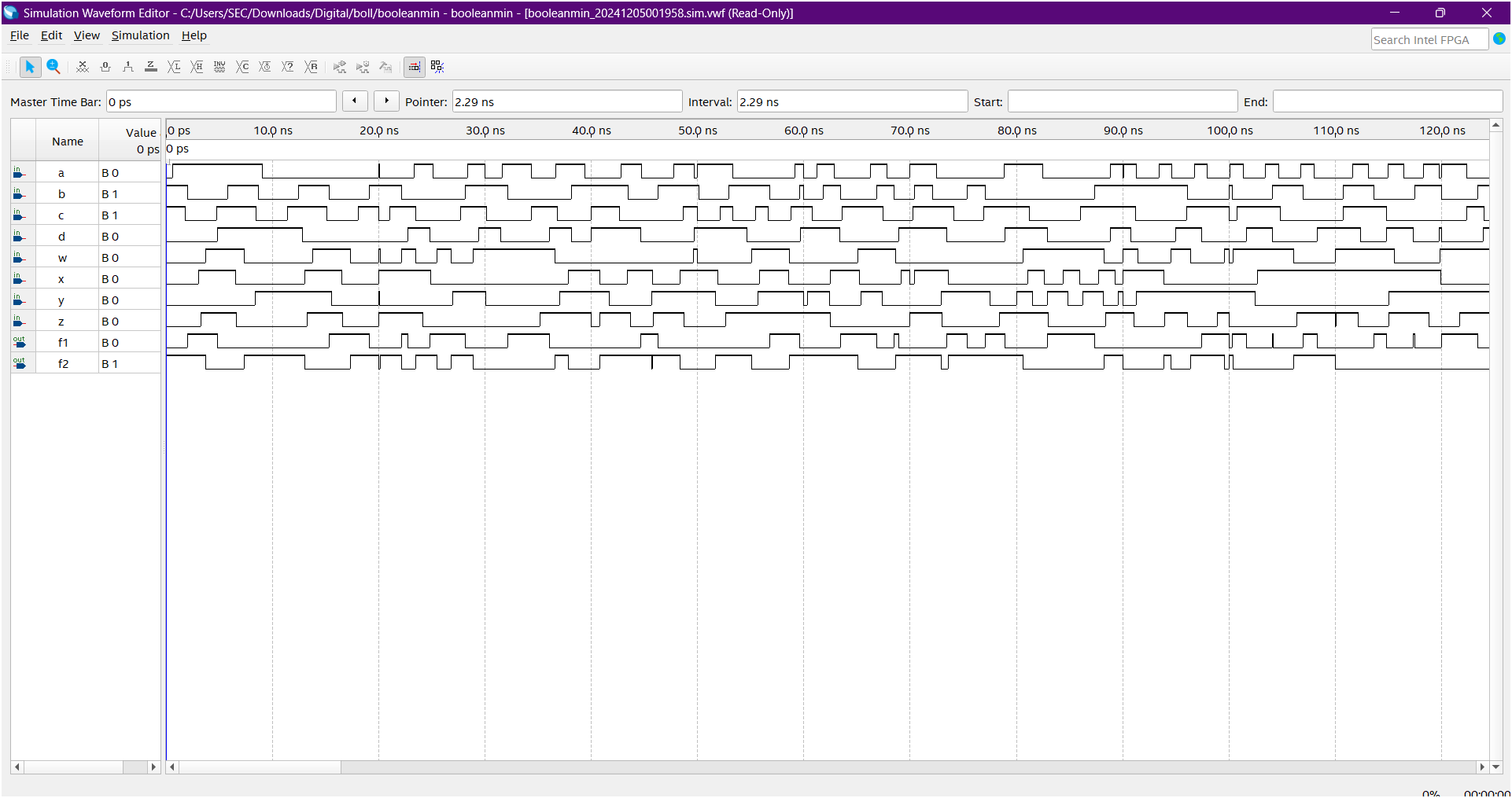
endmodule

**RTL realization :**

**TRUTH TABLE:**

****

**TIMING DIGRAMS:**

****

**Result:**

Thus the given logic functions are implemented using and their operations are verified using Verilog programming.

**Developed by: Rhudhresh**

**Register Number: 212223050039**