**1. De-Morgan’s Law**

**L.H.S:**

module DMorg(Y, A, B);

output Y;

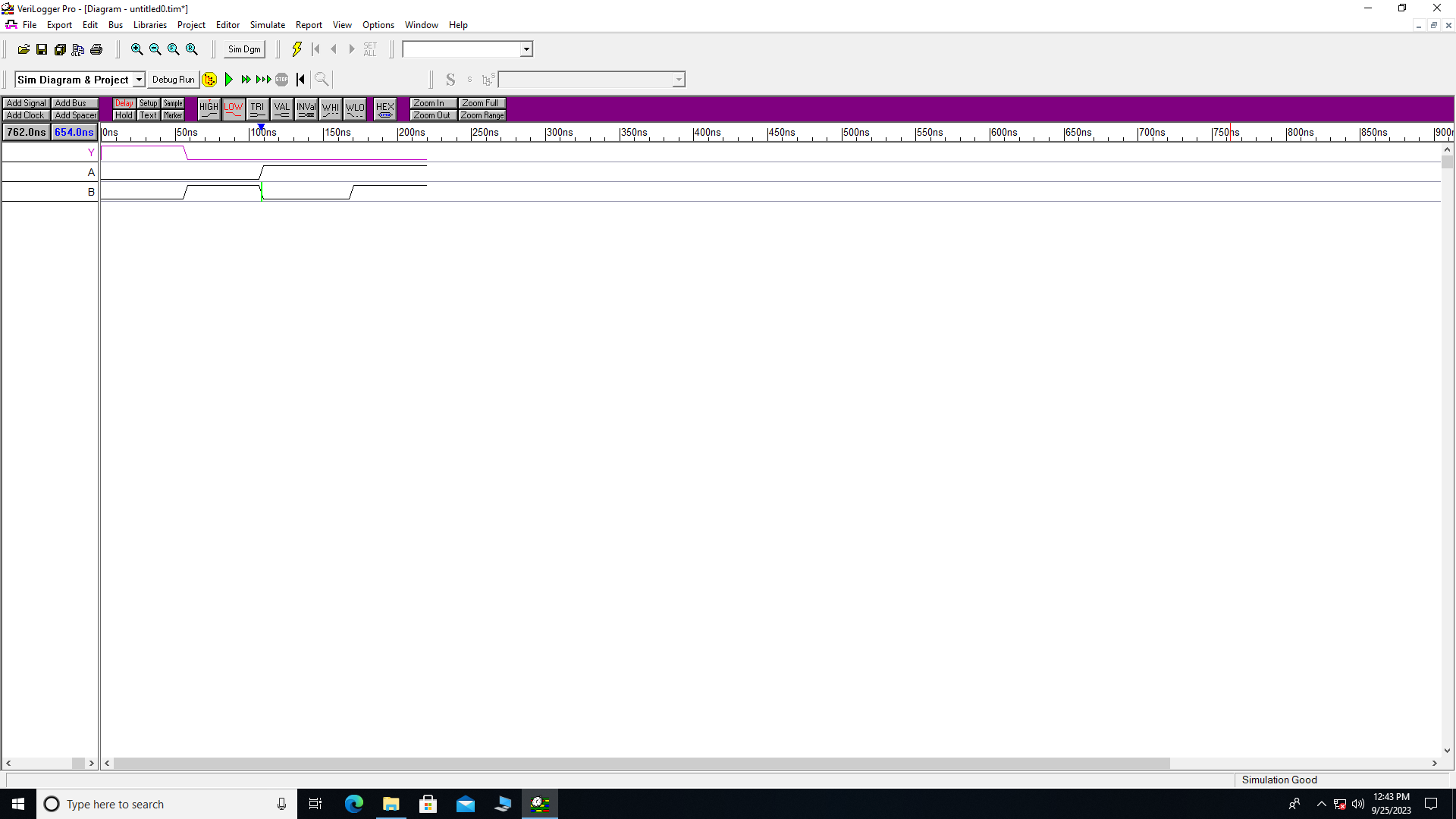
input A, B;

wire W1;

or r (W1, A, B);

not n (Y, W1);

endmodule



**R.H.S:**

module DMorg(Y, A, B);

output Y;

input A, B;

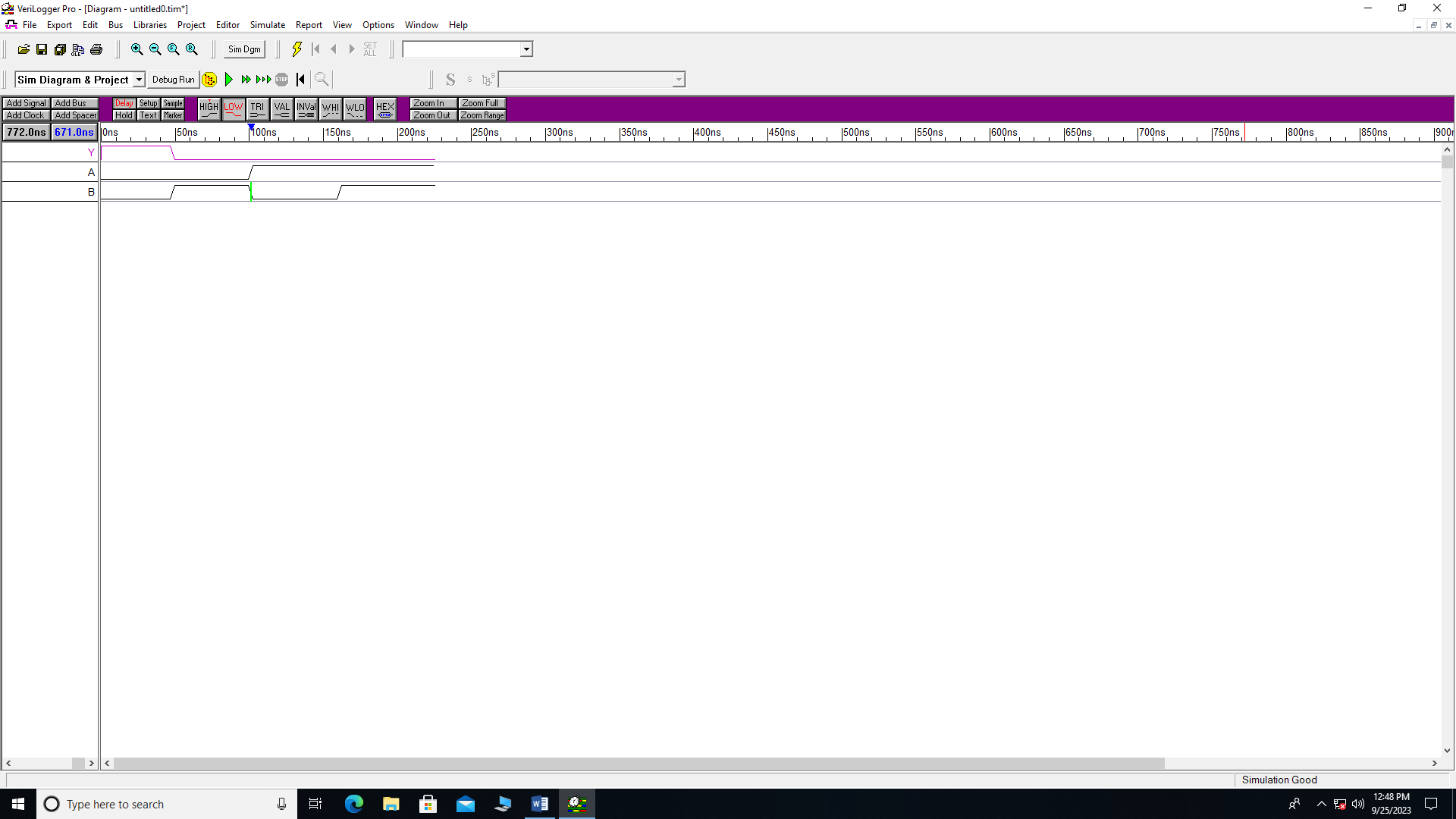
wire W1, W2;

not n1 (W1, A);

not n2 (W2, B);

and a(Y, W1, W2);

endmodule



**2. De-Morgan’s Law**

**L.H.S:**

module DMorg(Y, A, B);

output Y;

input A, B;

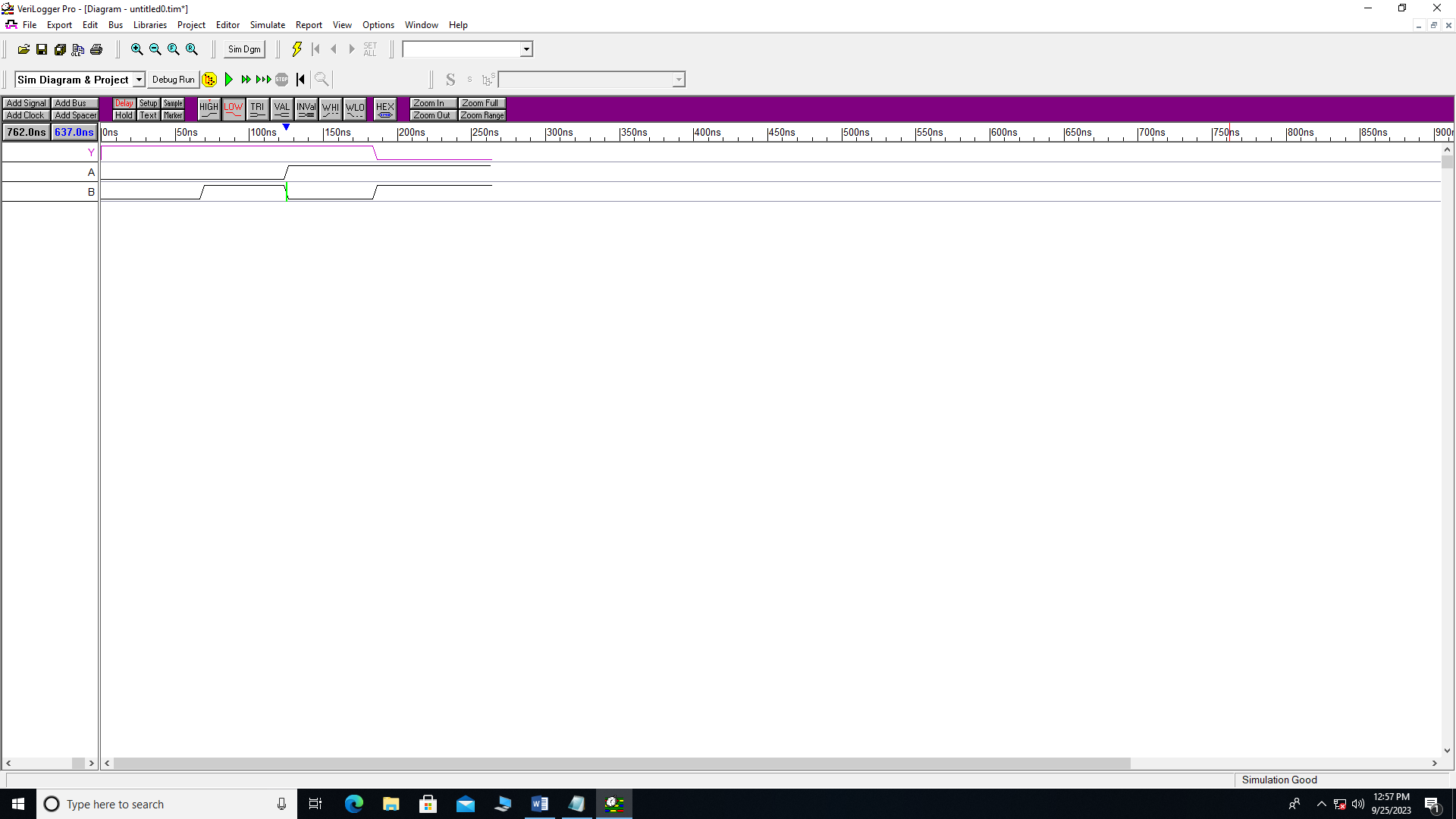
wire W1, W2;

not n1 (W1, A);

not n2 (W2, B);

or a(Y, W1, W2);

endmodule



**R.H.S:**

module DMorg(Y, A, B);

output Y;

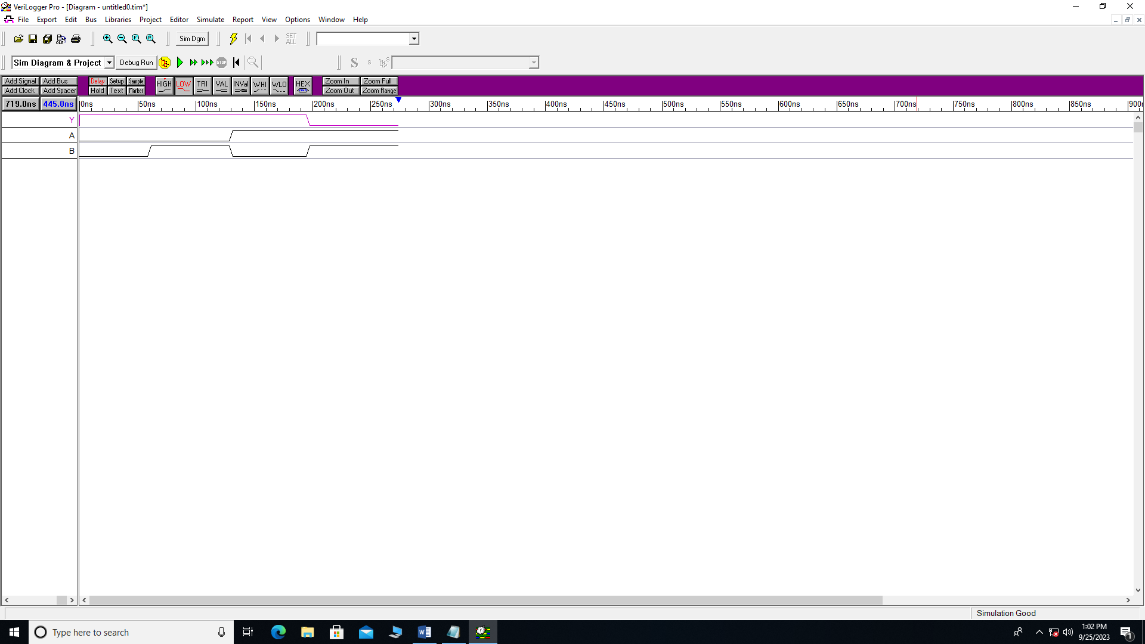
input A, B;

wire W1;

and a(W1, A, B);

not n1 (Y, W1);

endmodule



**Gates:**

module gates(Y, X, Z, L, A, B, C, D, E, M, N);

output Y, X, Z, L;

input A, B, C, D, E, M, N;

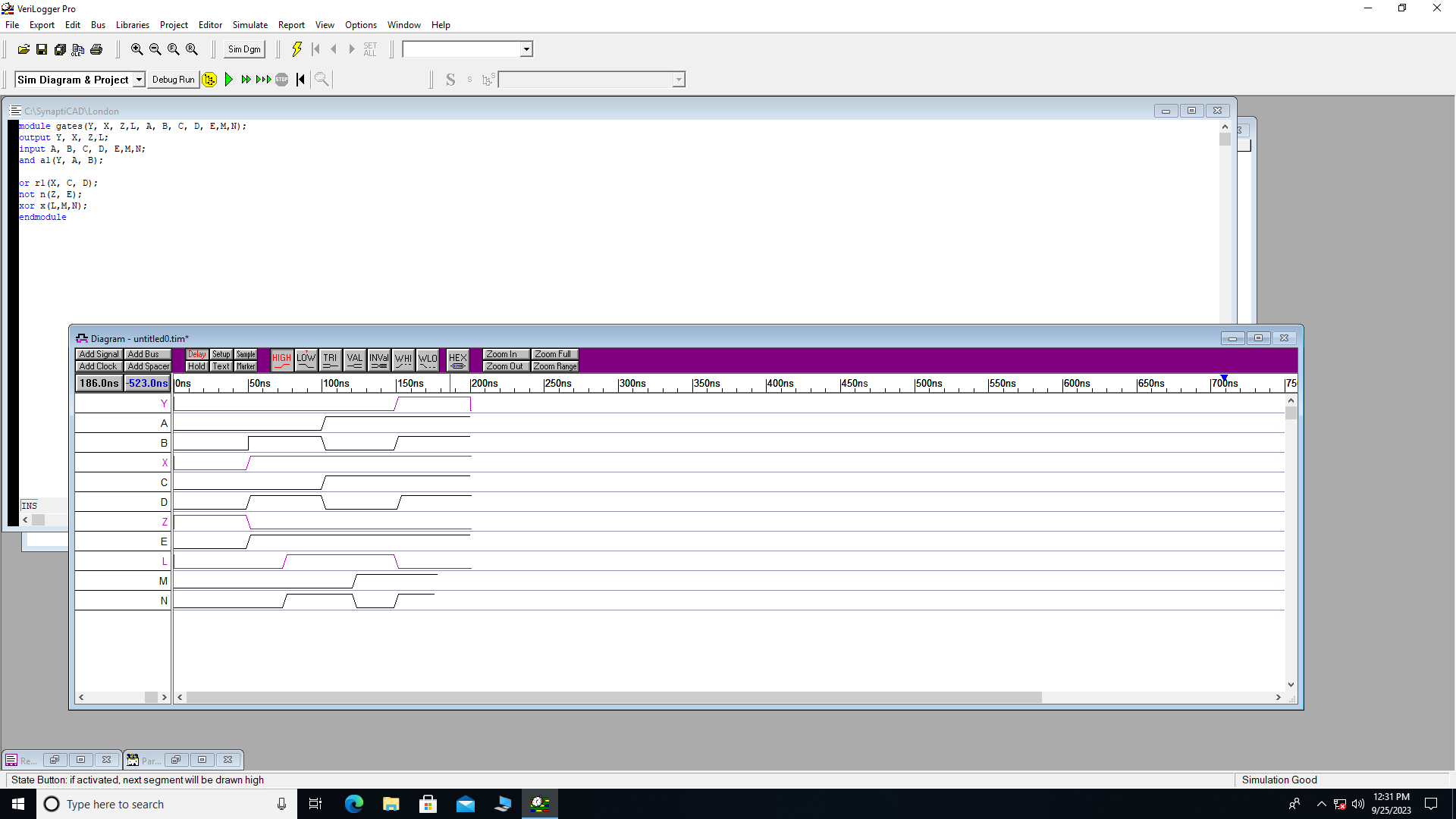
and a1 (Y, A, B);

or r (X, C, D);

not n (Z, E);

xor x1 (L, M, N);

endmodule



1. **Distributive Law**

**L.H.S:**

module Dist(Y, A, B, C);

output Y;

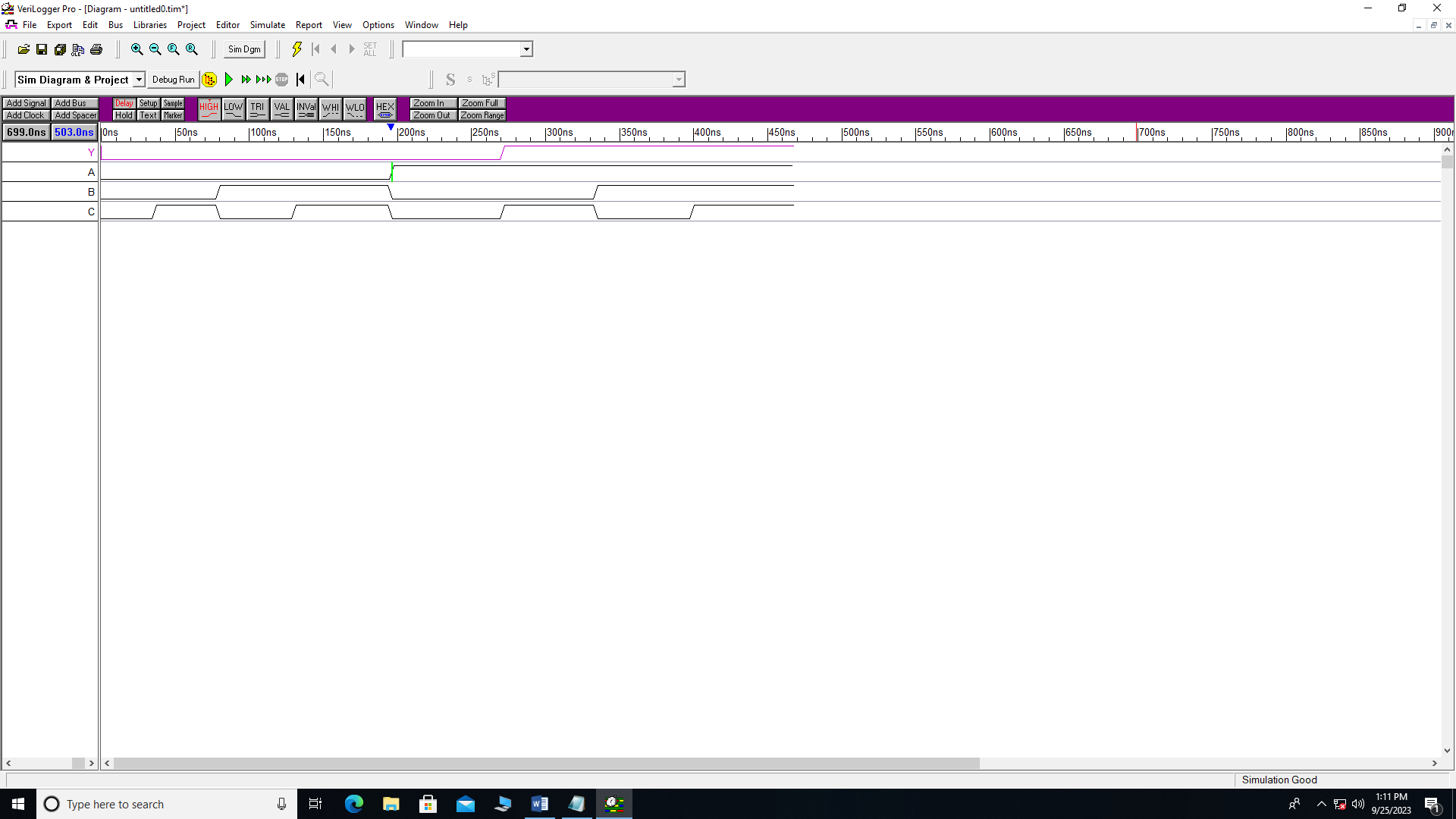
input A, B, C;

wire W1;

or r(W1, B, C);

and a(Y, W1, A);

endmodule

****

**R.H.S:**

module Dist(Y, A, B, C);

output Y;

input A, B, C;

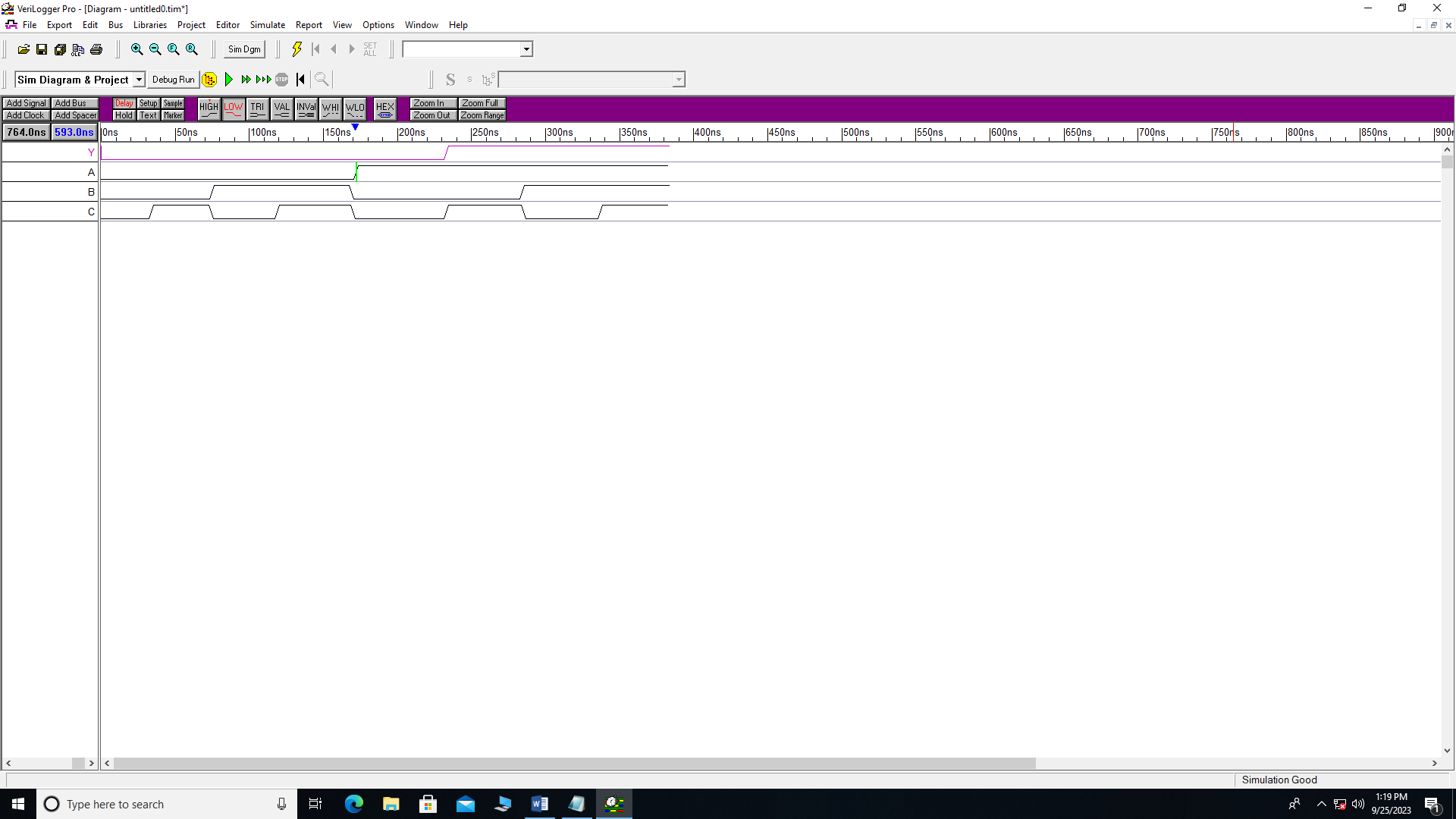
wire W1, W2;

and a1(W1, A, B);

and a(W2, A, C);

or r (Y, W1, W2);

endmodule



1. **Distributive Law**

module Dist(Yr, Yl, A, B, C, D);

output Yr, Yl;

wire w1, w2, w3, w4, w5, w6;

input A, B, C, D;

or x6 (w1, A, B);

or x7 (w2, C, D);

and x8 (Yl, w1, w2);

and x9 (w3, A, C);

and x10 (w4, A, D);

and x11 (w5, B, C);

and x12 (w6, B, D);

or x13 (Yr, w3 ,w4, w5, w6);

endmodule

