East West University

Department of CSE

Project Report

Project 1: 2's Complementor

Submitted To

Musharrat Khan

Senior Lecturer,
Department of Computer Science and Engineering
East West University

Submitted By

Asma Akhter
ID: 2019-3-60-033
Course Name: Digital Logic Design
Course Code: CSE 345
Section: 3

Date of Submission: 11/01/2022

1.Problem Statement:

In this problem there is a 4 bit 2's complementor. ABCD are 4-bit inputs and WXYZ are 4-bit outputs. I have to find out 4 bit 2's complement outputs (W X Y Z) from 4-bit inputs (A B C D), where 2's complement of 0000 is 0000.

2.Design Details:

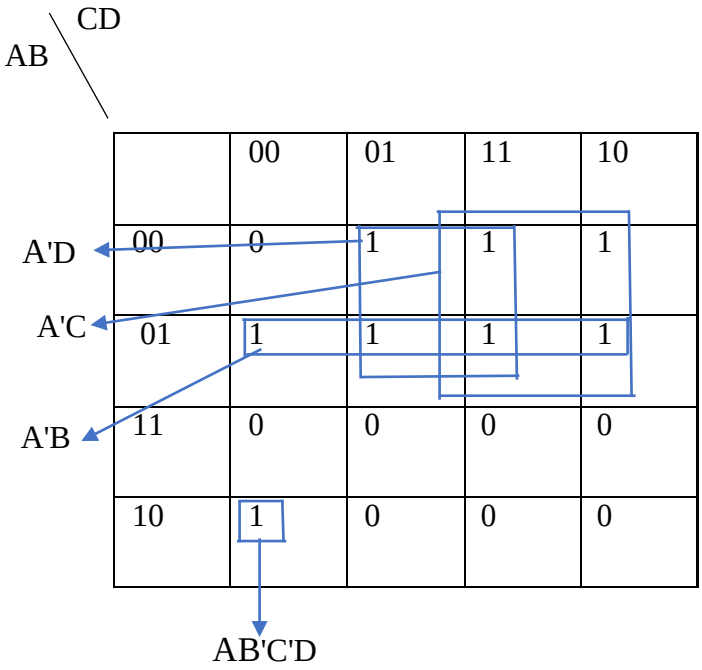
To complete this project, first I will construct a truth table. Truth table outputs are 4 bit 2's complementor of inputs. Then I will draw the k-map of each output. Now I will draw logic diagram by Boolean expression obtained from k-map and then I will do the simulation. Lastly, I will write two types behavioral Verilog code (procedural model and continuous assign statement) for the logic diagram and then I will do the simulation. In the end we will see that the results of logic diagram simulation, two types behavioral Verilog code simulation and the truth table output are same. This will complete the project.

Truth Table :

Inputs				Outputs			
A	**B**	**C**	**D**	**W**	**X**	**Y**	**Z**
0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1
0	0	1	0	1	1	1	0
0	0	1	1	1	1	0	1
0	1	0	0	1	1	0	0
0	1	0	1	1	0	1	1
0	1	1	0	1	0	1	0
0	1	1	1	1	0	0	1
1	0	0	0	1	0	0	0
1	0	0	1	0	1	1	1
1	0	1	0	0	1	1	0
1	0	1	1	0	1	0	1
1	1	0	0	0	1	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	0	1	0
1	1	1	1	0	0	0	1

K-map:

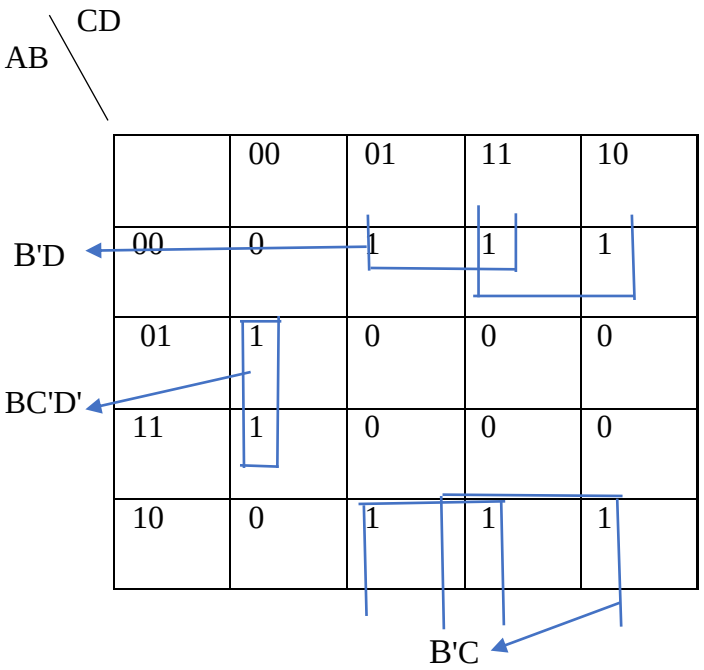
K-Map for W-



Boolean Expression for W-

W = A'B+A'D+A'C+AB'C'D'

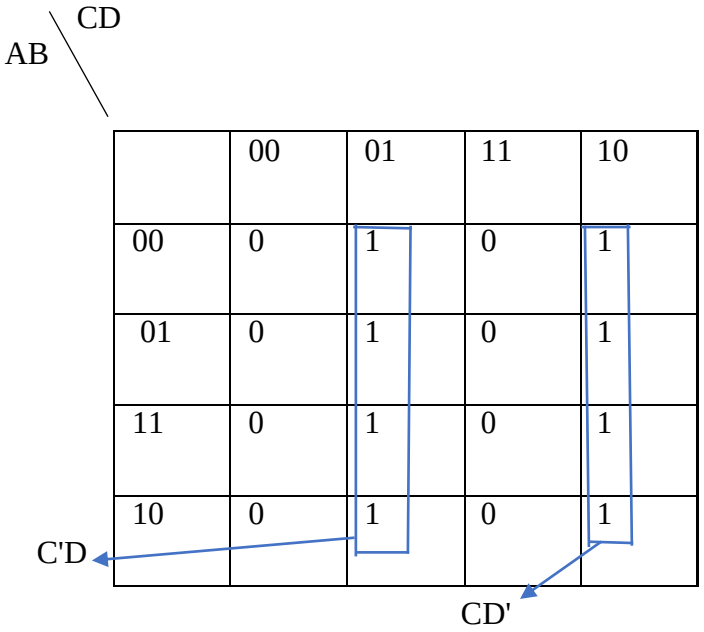
K-Map for X-



Boolean Expression for X-

X = B'D+B'C+BC'D'

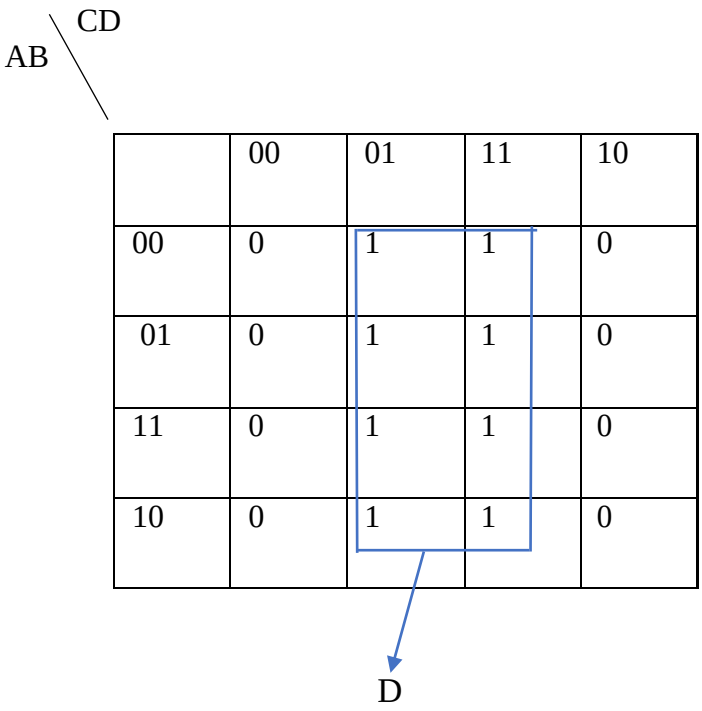
K-Map for Y-



Boolean Expression for Y-

Y = C'D+CD'

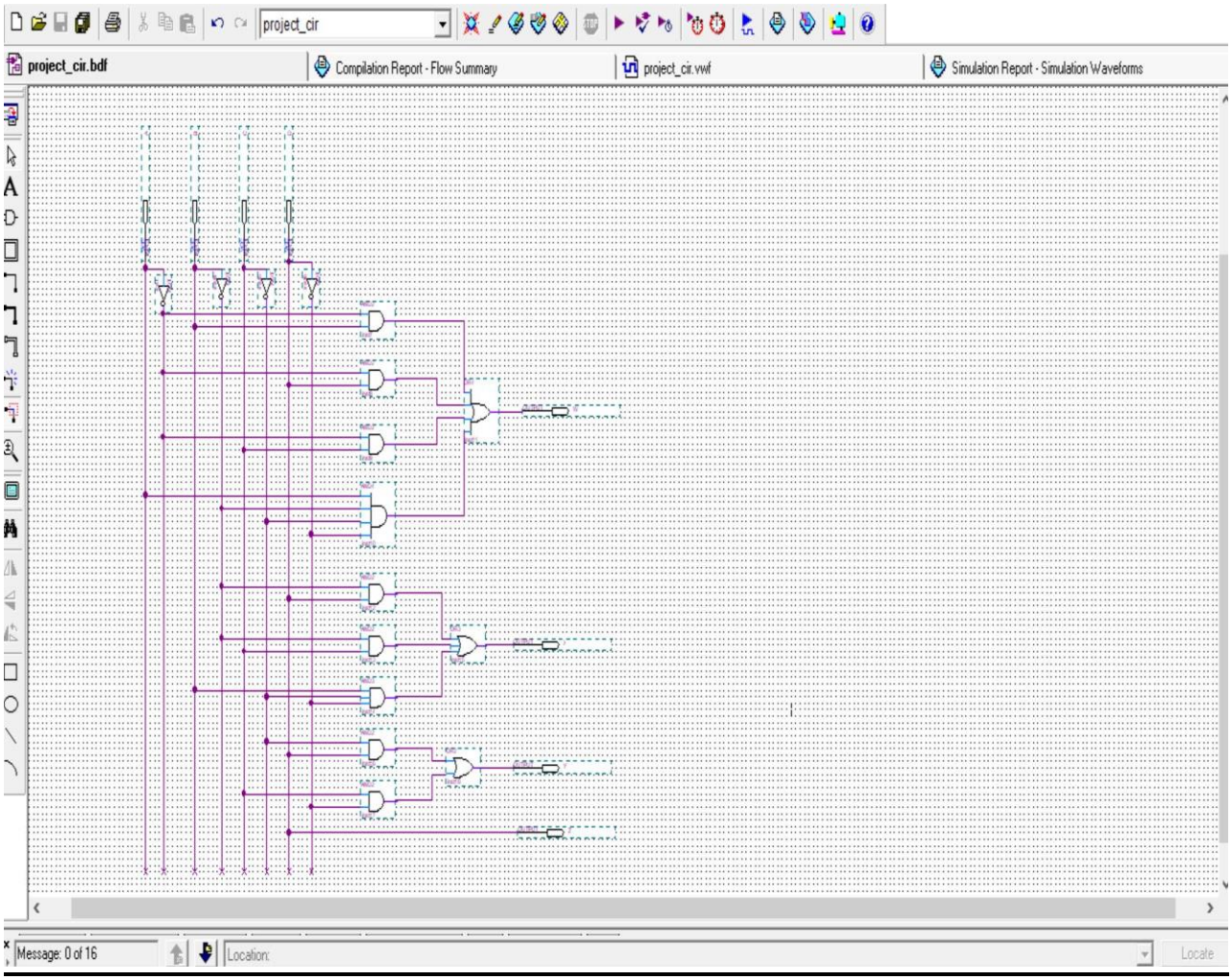
K-Map for Z-



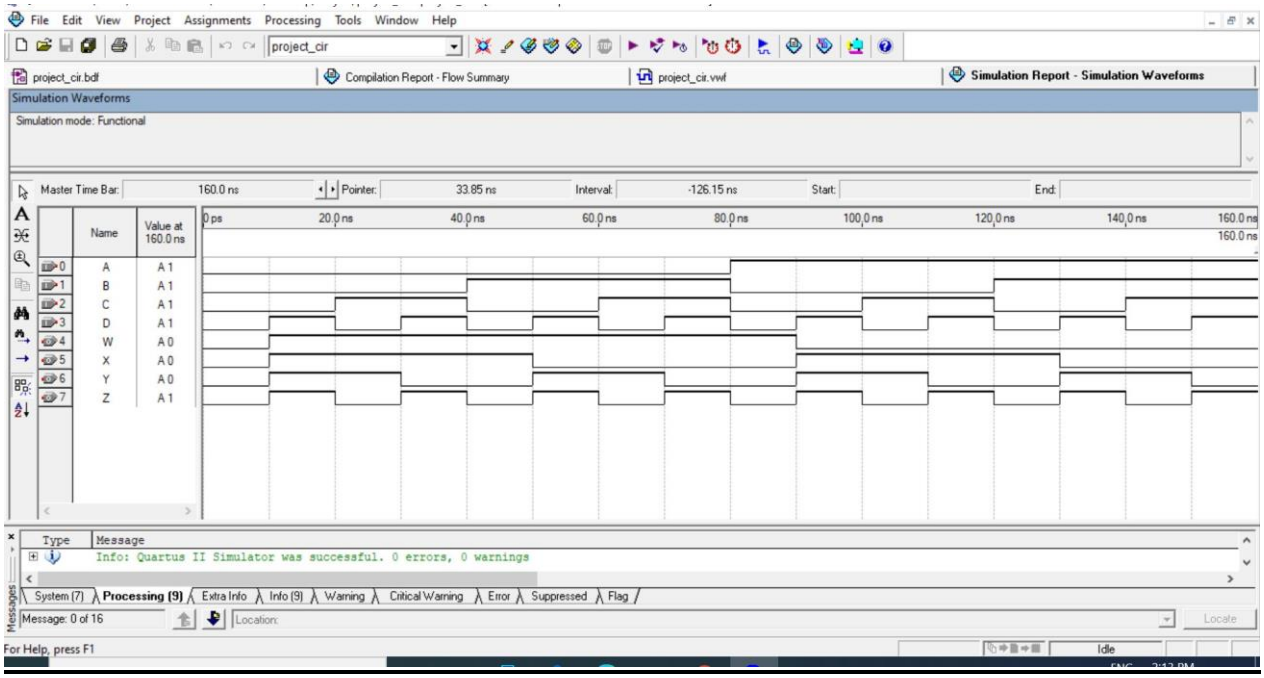
Boolean Expression for Z-

Z = D

3.Circuit Diagram:



Simulation Result:



4. Behavioral Verilog Code and Simulation Result:

Procedural model:

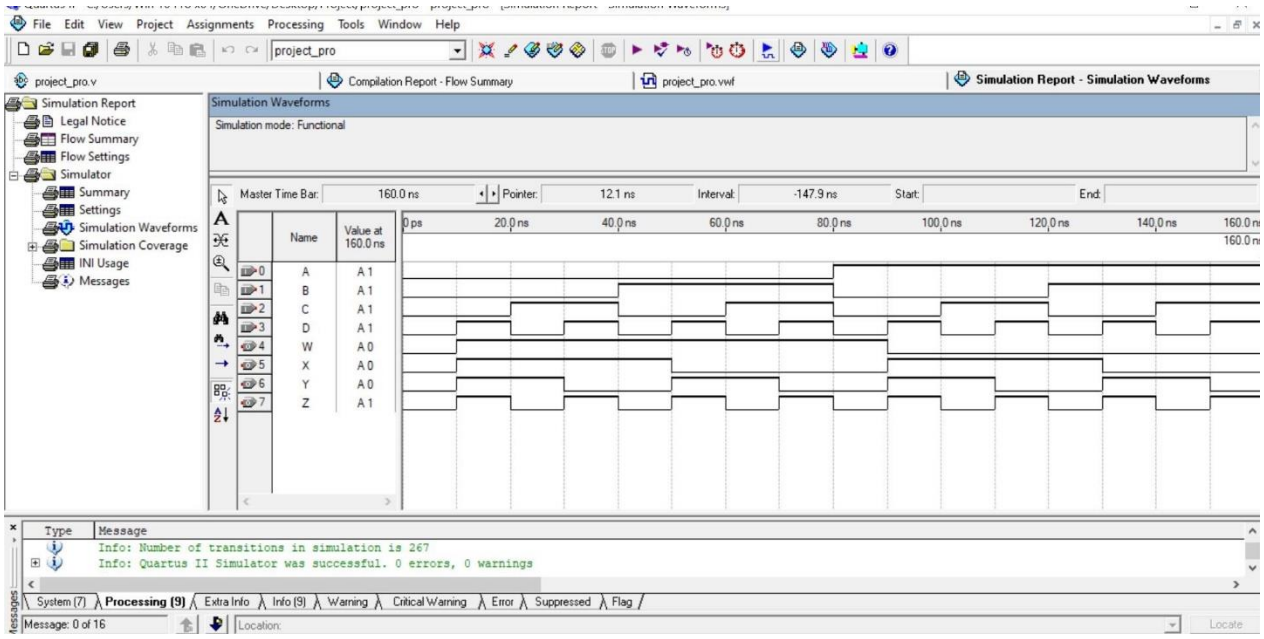
```
module project_pro ( input A, B, C, D, output reg W, X, Y, Z );
always@( A, B, C, D ) begin
W=0;
X=0;
Y=0;
Z=0;
if ( ~A & ~B & ~C & D ) W=1;
if ( ~A & ~B & C & ~D ) W=1;
if ( ~A & ~B & C & D )  W=1;
```

```
if ( ~A & B & ~C & ~D ) W=1;
if ( ~A & B & ~C & D )  W=1;
if ( ~A & B & C & ~D )  W=1;
if ( ~A & B & C & D )   W=1;
if ( A & ~B & ~C & ~D)  W=1;
if ( ~A & ~B & ~C & D ) X=1;
if ( ~A & ~B & C & ~D ) X=1;
if ( ~A & ~B & C & D )  X=1;
if ( ~A & B & ~C & ~D ) X=1;
if ( A & ~B & ~C & D )  X=1;
if ( A & ~B & C & ~D )  X=1;
if ( A & ~B & C & D )   X=1;
if ( A & B & ~C & ~D )  X=1;
if ( ~A & ~B & ~C & D ) Y=1;
if ( ~A & ~B & C & ~D ) Y=1;
if ( ~A & B & ~C & D )  Y=1;
if ( ~A & B & C & ~D )  Y=1;
if ( A & ~B & ~C & D )  Y=1;
if ( A & ~B & C & ~D )  Y=1;
if ( A & B & ~C & D )   Y=1;
if ( A & B & C & ~D )   Y=1;
if ( ~A & ~B & ~C & D ) Z=1;
if ( ~A & ~B & C & D )  Z=1;
if ( ~A & B & ~C & D )  Z=1;
if ( ~A & B & C & D )   Z=1;
if ( A & ~B & ~C & D )  Z=1;
if ( A & ~B & C & D )   Z=1;
if ( A & B & ~C & D )   Z=1;
if ( A & B & C & D )    Z=1;
```

end

endmodule

Simulation result:



Continuous assign Statement:

```
module project_con ( input A, B, C, D, output W, X, Y, Z );

assign W = ( ~A & ~B & ~C & D ) | ( ~A & ~B & C & ~D ) | ( ~A & ~B & C & D ) | ( ~A & B & ~C & ~D ) | ( ~A & B & ~C & D )|( ~A & B & C & ~D ) | ( ~A & B & C & D ) | ( A & ~B & ~C & ~D );

assign X = ( ~A & ~B & ~C & D ) | ( ~A & ~B & C & ~D ) | ( ~A & ~B & C & D ) | ( ~A & B & ~C & ~D ) | ( A & ~B & ~C & D ) | ( A & ~B & C & ~D ) | ( A & ~B & C & D ) | ( A & B & ~C & ~D );

assign Y = ( ~A & ~B & ~C & D ) | ( ~A & ~B & C & ~D ) | ( ~A & B & ~C & D ) | ( ~A & B & C & ~D ) | ( A & ~B & ~C & D ) | ( A & ~B & C & ~D ) | ( A & B & ~C & D ) | ( A & B & C & ~D );

assign Z = ( ~A & ~B & ~C & D ) | ( ~A & ~B & C & D ) | ( ~A & B & ~C & D ) | ( ~A & B & C & D ) | ( A & ~B & ~C & D ) | ( A & ~B & C & D ) | ( A & B & ~C & D ) | ( A & B & C & D );

endmodule
```

Simulation Result: