**Ibrahim Abou Zahr**

*Lab 3*

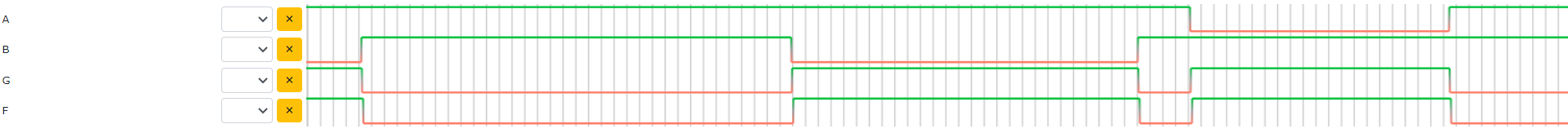
Ex3:

Boolean expression F and Q: (Y+(X+Y)’) + (X + (X+Y)’) 🡪 Using Demorgan’s Law = (Y+(X.’Y’) + (X+ (X’.Y’) = XY + X’.Y’.Y + XY + X.X’.Y’ = XY + X’Y’.

Q = X xnor Y = 🡪 XY + X’Y’ = F **(verified)**.

Ex4:

**Waves:**

****

**Code Used:**

module Lab3(A,B,F,G);

input A,B;

output F,G;

not(P,A);

not(L,B);

and (W,P,B);

and(D,L,A);

or(F,W,D);

xor (G,A,B);

endmodule

**Truth Table:**

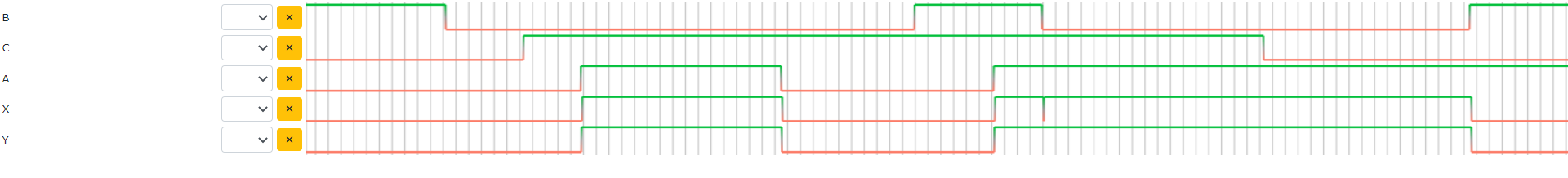
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | A’ | B’ | A’.B | A.B’ | F=(A’.B) + (A.B’) | Q=A xor B |
| T | T | F | F | F | F | F | F |
| T | F | F | T | F | T | T | T |
| F | T | T | F | T | F | T | T |
| F | F | T | T | F | F | F | F |

Boolean Expression:

Q and F 🡪 (A’.B) + (A.B’) = A xor B = Q **(verified)**

Ex5:

**Waves:**



**Code Used:**

module Lab3(A,B,C,X,Y);

input A,B,C;

output X,Y;

not(N1,A);

not(N2,C);

nand(P,N1,N2);

and(P2,A,C,B);

not(N3,B);

and(P3,A,P,N3);

or(X,P2,P3);

not(N4,B);

or(R,C,N4);

and(Y,A,R);

endmodule

**Boolean expression:** X and Y 🡪 Y = (B’+C).A

X = (A.B.C) + ((A’.B’ + AB’C)

X = (A.B.C) + AB’ + AB’C

X = AC(B+B’) + AB’

X = AC + AB’

X = (B’+C).A 🡪 Y **(verified).**