Mantık Kapılarıyla Boolean Fonksiyonları - 2

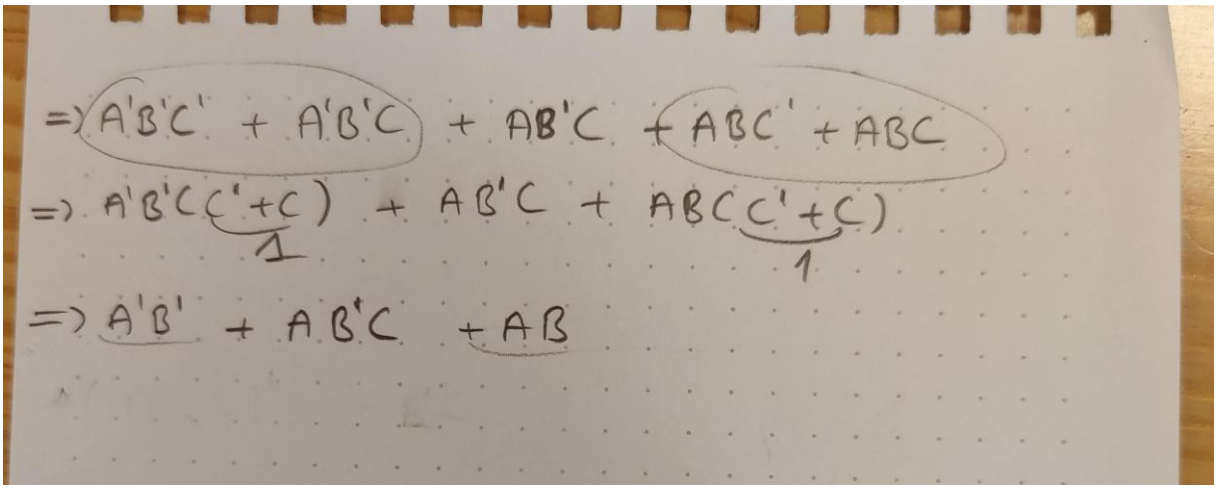
Aşağıda verilen örnekler, Vivado programı üzerinden System Verilog dili yardımıyla logic devreleri tasarlanarak simüle edilecektir.

2. **ÖRNEK: F(A,B,C) = Σ m(0,1,5,6,7)**

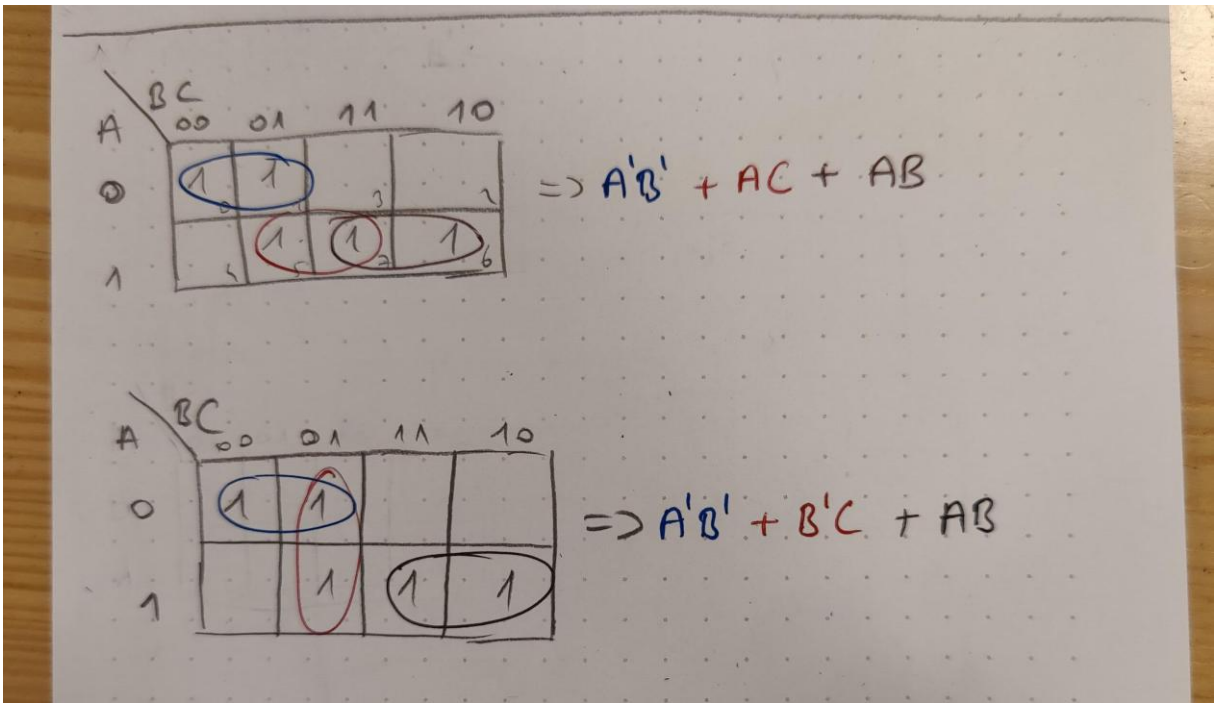
Öncelikle doğruluk tablosunu çıkartmakla işe başlayalım:

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

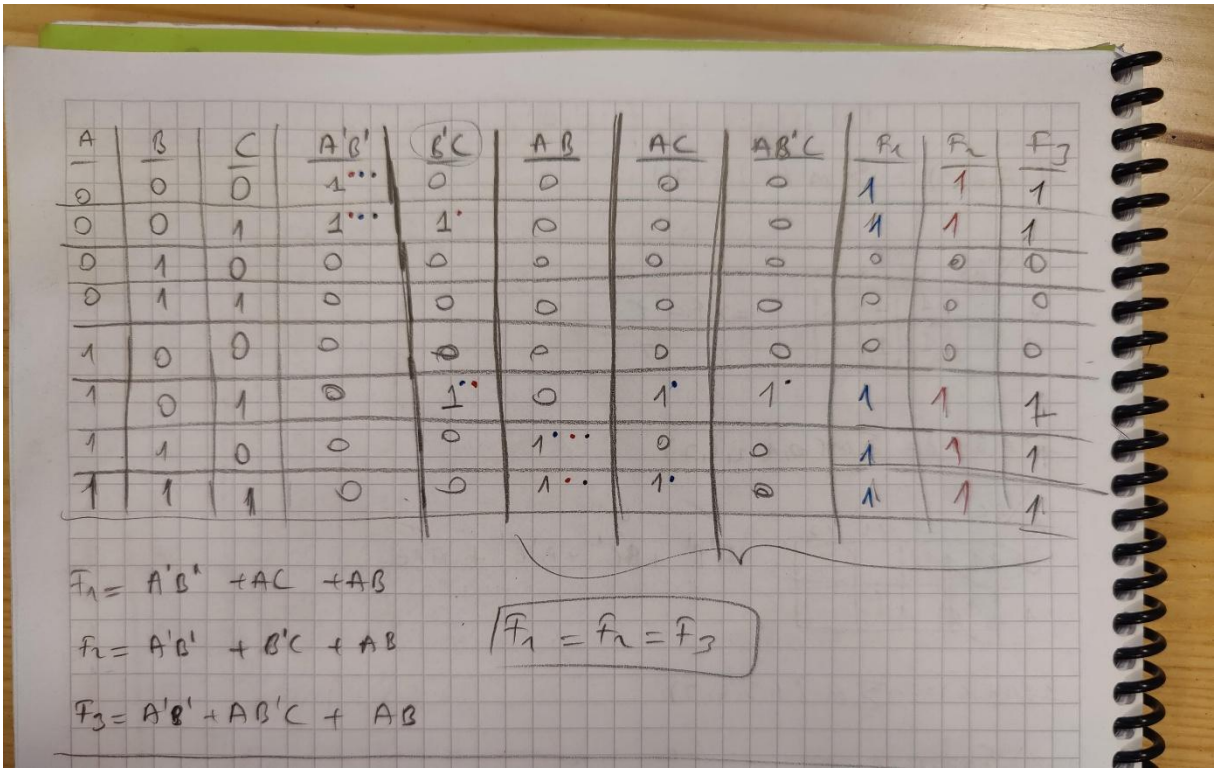
Ardından sadeleştirelim:



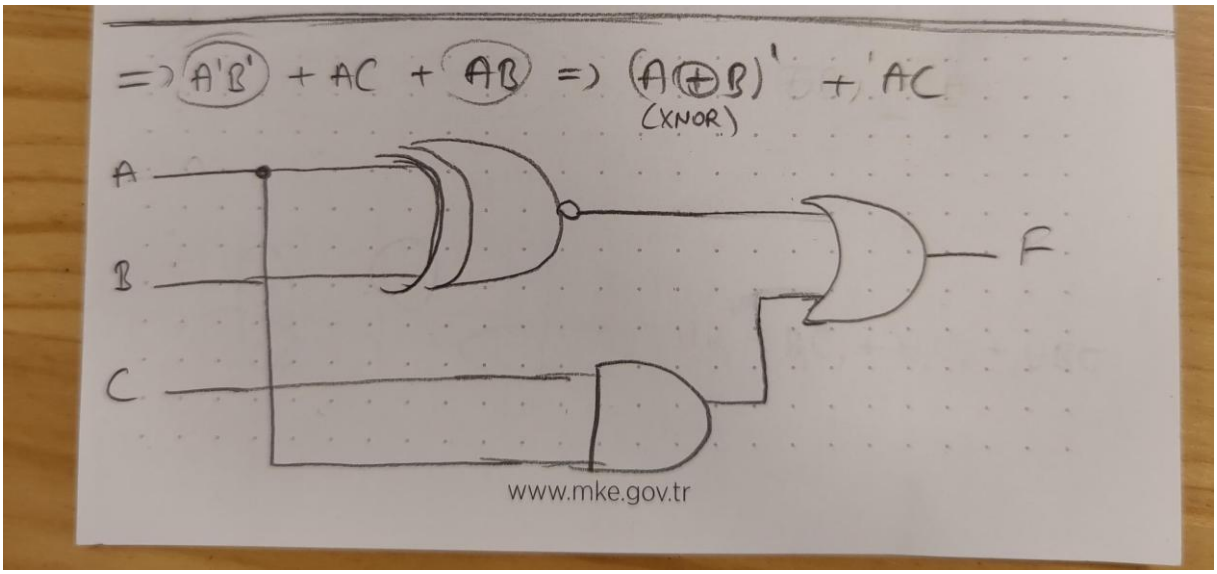
Daha iyi görebilmek adına K-Map kullanarak da sadeleştirme yapalım:



Görüldüğü üzere K-Map ve cebir kullanarak 3 farklı sonuç bulmuşuz gibi duruyor fakat işin özünde bu sonuçların hepsi birbirine eşittir. Şimdi doğruluk tablosuna birkaç ekleme yaparak bulduğumuz 3 ifadenin de birbirine eşit olduklarını gösterelim:



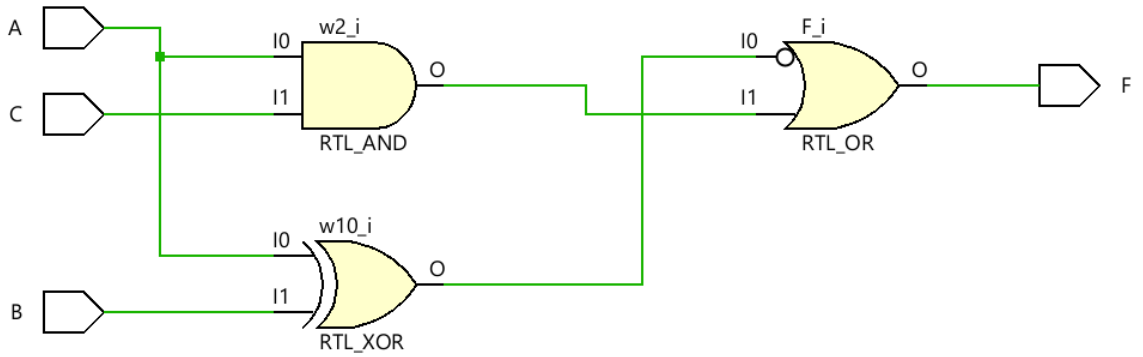
Görüldüğü üzere bulunan bütün fonksiyonlar birbirine eşittir. Birini seçelim ve XNOR kapısı yardımıyla devre tasarımızı yapmaya başlayalım:



Vivado üzerinden kodumuzu yazmaya başlayalım:

```
module calisma2(
    input logic A,
    input logic B,
    input logic C,
    output logic F
    );

    logic w1, w2; // ara sinyaller

    xnor(w1,A,B);
    and(w2,A,C);
    or(F,w1,w2);

    // assign F = (!A && !B) || (A && C) || (A && B);
    // assign F = (A !~ B) | (A & C);

endmodule
```

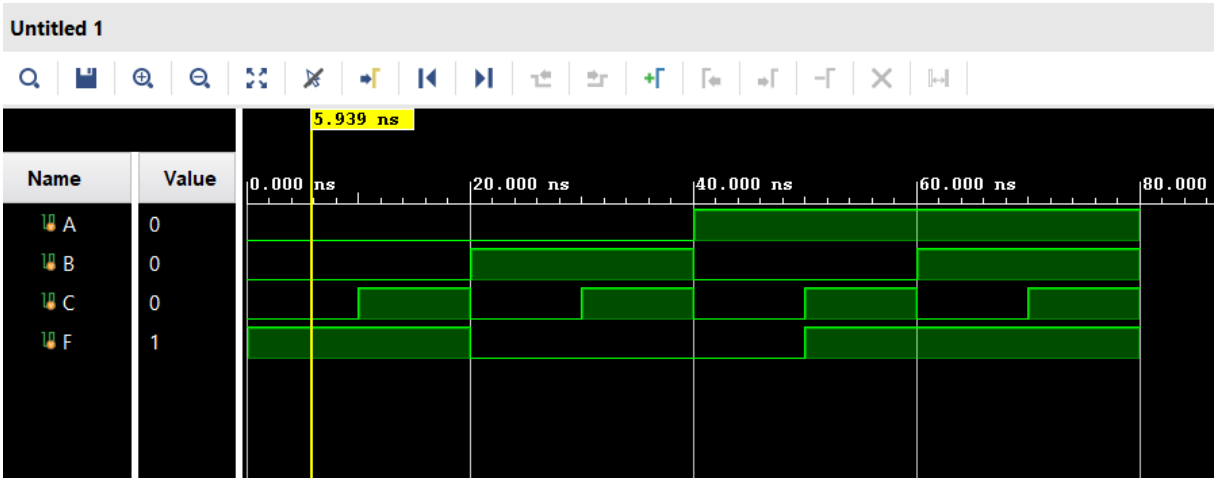
“Open Elaborated Design” dediğimizdeyse design çıktımız şu şekildedir:



Şimdi de simülasyon kodumuzu yazalım:

```
module tb_calisma2();
    logic A; // reg
    logic B; // reg
    logic C; // reg
    logic F; // wire

    calisma2 calisma2_Inst(.*); //hepsini bağladık.

    initial begin
        A = 1'b0; B = 1'b0; C = 1'b0; #10;
        A = 1'b0; B = 1'b0; C = 1'b1; #10;
        A = 1'b0; B = 1'b1; C = 1'b0; #10;
        A = 1'b0; B = 1'b1; C = 1'b1; #10;
        A = 1'b1; B = 1'b0; C = 1'b0; #10;
        A = 1'b1; B = 1'b0; C = 1'b1; #10;
        A = 1'b1; B = 1'b1; C = 1'b0; #10;
        A = 1'b1; B = 1'b1; C = 1'b1; #10;

        // 1 -> 1 bit, b -> binary, 0,1 -> değer
        // #10 -> 10ns bekle
        $stop; // simülasyonu durdurur, olası hataları görmek için işlevlidir.
    end

endmodule
```

“Run Simulation – Run Behavioral Simulation” kısmından simülasyonu çalıştıralım:



İşlemlerimiz bu kadardır.

3. ÖRNEK:

Bir güvenlik şirketi, sizden mesai saatleri dışında şirket binasının ana giriş – çıkış kapısını kontrol altında tutmak için iki alarmlı bir güvenlik sistemi tasarlamanızı istiyor. Bu alarmlardan birisi, sadece şirket kapısı önünde herhangi bir hareket algılanırsa ve kapı açılırsa; diğerinin ise olası sensör hatalarını tespit etmek için (hareketin olup kapının açılmadığı / hareketin olmayıp kapının açıldığı) devreye girmelerini istiyor. Bu sistemi en basit şekilde tasarlayınız.

ÇÖZÜM:

Öncelikle "input" olarak kabul edeceğimiz durumları tespit ederek işe başlayalım:

A: Kapının açık olup olmama durumu

B: Kapı önünde herhangi bir hareketin olup olmama durumu

Şimdi de "output" olarak kabul edeceğimiz durumları tespit edelim:

Alarm_out: Şirket kapısı önünde hareketin algılanıp kapının açılma durumu

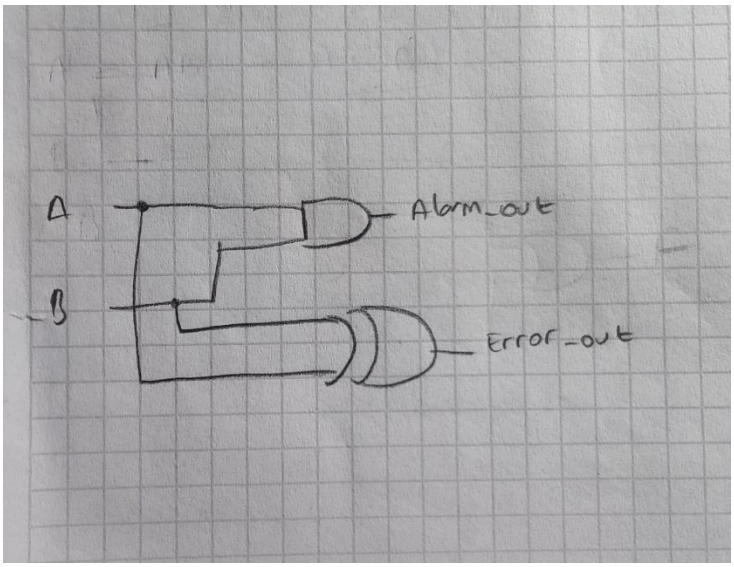
Error_out: Olası sensör hataları

Doğruluk tablosunu çıkaralım:

A	B	Alarm_out	Error_out
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Açık bir şekilde görülüyor ki; Alarm_out = AB, Error_out = A xor B

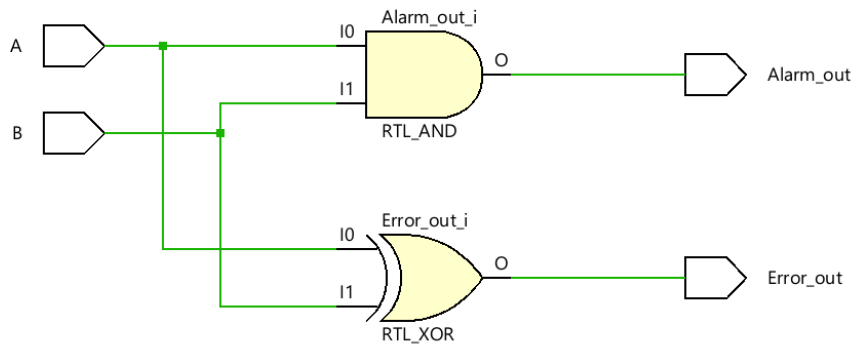
Devremiz şuna benzeyecektir:



Vivado üzerinden design kodumuzu yazmaya başlayalım:

```
module calismaa3(
    input logic A, // kapı sensörü
    input logic B, // hareket sensörü
    output logic Alarm_out, // şirket kapısı önünde hareketin algılanıp
kapının açılma durumu
    output logic Error_out // olası sensör hataları
    );

    assign Alarm_out = A && B; //alarm için and kapısı
    assign Error_out = A ^ B; //hata için xor kapısı

endmodule
```

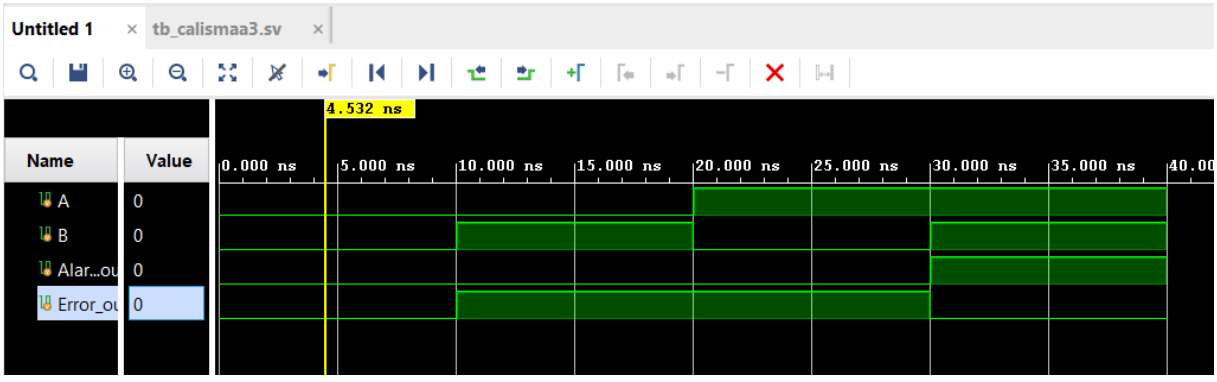
Elaborated Design dediğimizde ise design çıktımız aşağıdaki gibidir:



Simülasyon kodunu yazalım:

```
module tb_calismaa3();
    logic A;
    logic B;
    logic Alarm_out;
    logic Error_out;

    calismaa3 calismaa3_Inst(.*);

    initial begin
        A = 1'b0;
        B = 1'b0;
        #10;

        A = 1'b0;
        B = 1'b1;
        #10;

        A = 1'b1;
        B = 1'b0;
        #10;

        A = 1'b1;
        B = 1'b1;
        #10;

        $stop;
    end
endmodule
```

“Run Simulation – Run Behavioral Simulation” kısmından simülasyonu çalıştıralım:



İşlemlerimiz bu kadardır.