



GEBZE TEKNİK ÜNİVERSİTESİ  
ELEKTRONİK MÜHENDİSLİĞİ

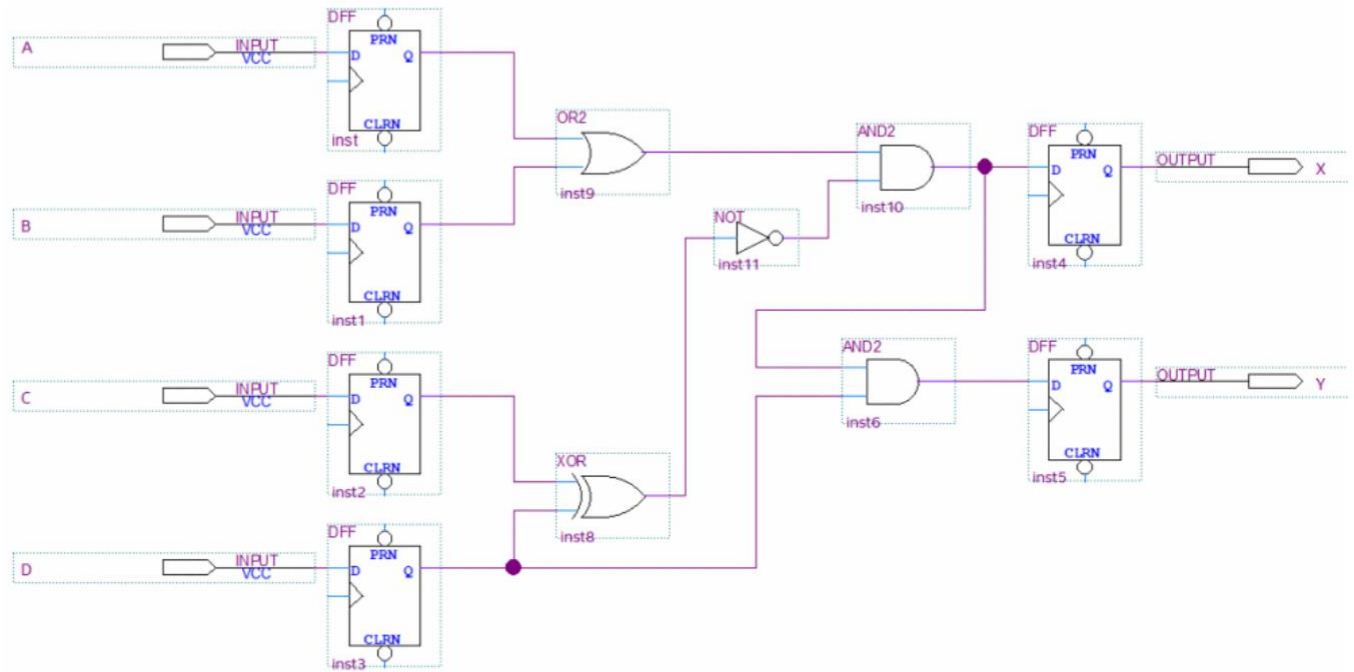
ELM234  
LOJİK DEVRELER VE TASARIM

ÖDEV 2

Alican Bayındır  
200102002087

**Problem 1. Dinamik Disiplin ve Pipeline** [4 + 10 + 4 + 6 + 12 = 36 puan]

- A. Şekil 1 de verilen devrenin bütün clock girişlerini aynı clocka bağlandığını varsayalım. Tablo 1 de verilen değerlere göre devrenin maximum frekansını bulunuz. Setup ve Hold zamanlamalarına uymuyorsa ona göre gerekli yere ekstra lojik ekleyip, devreyi tekrar analiz ediniz.  $t_{ccq} = 30\text{ps}$  ve  $t_{pcq} = 60\text{ps}$ .
- B. A şıkkındaki devrenizi HDL kullanarak tasarlayın, bir testbench devresi oluşturun ve Modelsim de simüle edin. Simülasyonda propagation zaman atamalarını yapmayı unutmayın. Quartus'da bu devreyi sentezleyerek bu devrede ulaşabileceğiniz max frekansını bulun



Şekil 1

- C. Şekil 1 de verilen devreyi Clock Skew = 15ps olduğunu varsayarak tekrar analiz ediniz. maximum çalışma frekansını bulunuz. Setup ve Hold zamanlamalarına uymuyorsa ona göre gerekli yere ekstra lojik ekleyip, devreyi tekrar analiz ediniz.
- D. Şekil 1 de verilen devrenin **C şıkkındaki verilen clock skew değerine ve çözümünüze göre**, latency ve throughputunu hesaplayınız. (Eğer C şıkkını çözemedi iseniz bunu açıkça belirtip clock skew değerini bu soru için 10ps alabilirsiniz) Max frekansını arttırmak için en uygun yere pipeline ekleyip, tekrar hesaplamalarınızı yapınız. Throughput, latency, max freq, ve setup ve hold zamanlamalarına uyduğunu gösterin. Pipeline sayısı kendiniz seçebilirsiniz.
- E. D şıkkındaki devrenizi HDL kullanarak tasarlayın, bir testbench devresi oluşturun ve Modelsim de simüle edin. Simülasyonda propagation zaman atamalarını yapmayı unutmayın. Quartus'da bu devreyi sentezleyerek bu devrede ulaşabileceğiniz max frekansını bulun. B şıkkındaki devrenizle karşılaştırın.

Kapı	$t_{pd}$ (ps)	$t_{cd}$ (ps)
NOT	35	30
BUFFER	40	35
2-giriş AND	50	50
2-giriş XOR	80	80
2-giriş OR	55	55
3-giriş OR	70	70
all others	90	90
SETUP	45	
HOLD	90	

Tablo 1

A)  $T_c = t_{pcq} + t_{pd} + t_{setup}$

$$T_c = 60ps + (80 + 35 + 50 + 50) + 45$$

$$T_c = 320ps$$

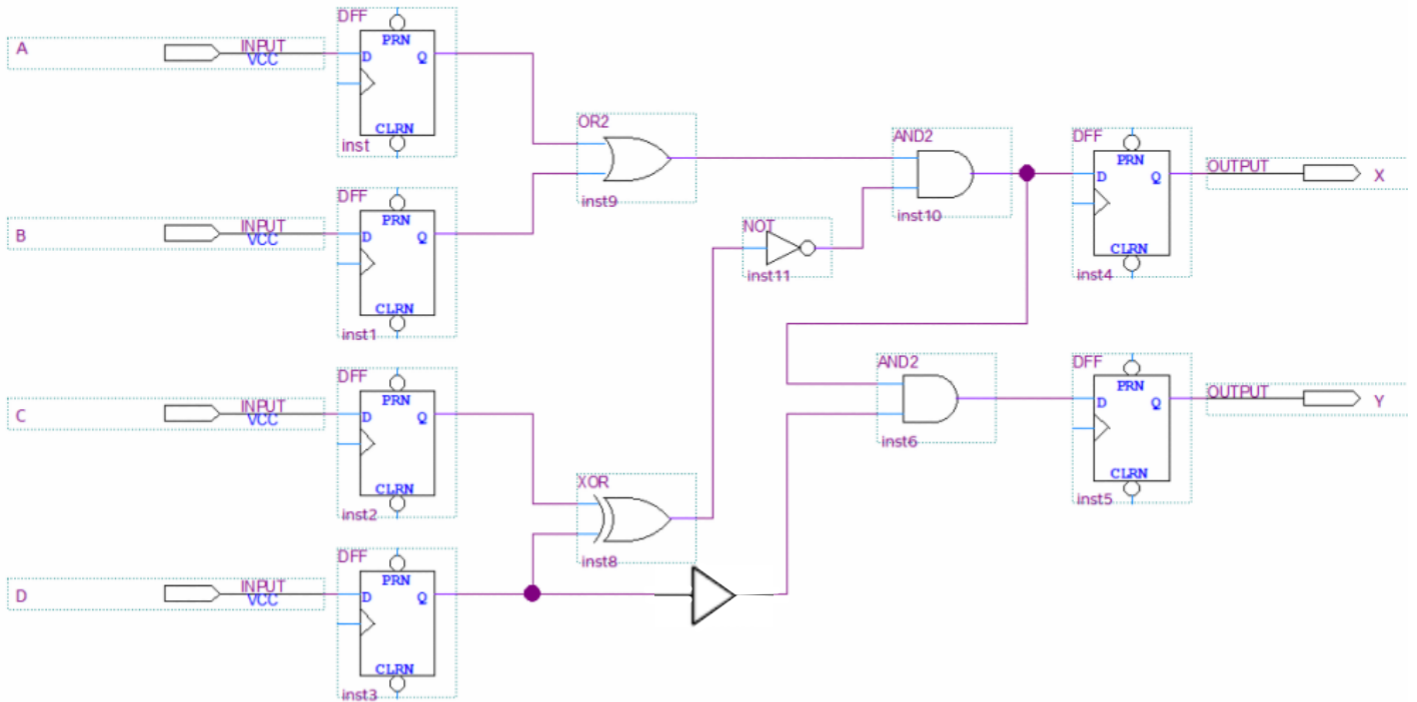
$$t_{hold} < t_{ccp} + t_{cd}$$

$$90 < 30 + 50$$

hold time bu denkleme uymadığı için devreye buffer eklenip, gecikme artırılır  
Buffer ekleyince;

$$90 < 30 + 85 \checkmark$$

$$f = \frac{1}{320} = 3.125 \times 10^{-3} \text{ MHz}$$



c)  $T_c = t_{pcq} + t_{pd} + t_{setup} + t_{skew}$

$$T_c = 60ps + (80 + 35 + 50 + 50) + 45 + 15 = 335ps$$

$$t_{cd} > t_{hold} + t_{skew} - t_{ccq}$$

$$50 > 90 + 15 - 30$$

50 > 75 yine buffer eklersek denklem çözülür

$$50 + 35 > 75$$

$$f = \frac{1}{T_c} = \frac{1}{335}$$

D)

Latency: Başlangıç ile bitiş arasındaki

Throughput: 1 saatteki ürün sayısı

$$T_c = 335 \text{ ps}$$

$$\text{Latency} = 335 \text{ ps}$$

$$\text{Throughput} = 1/335 \text{ ps}$$

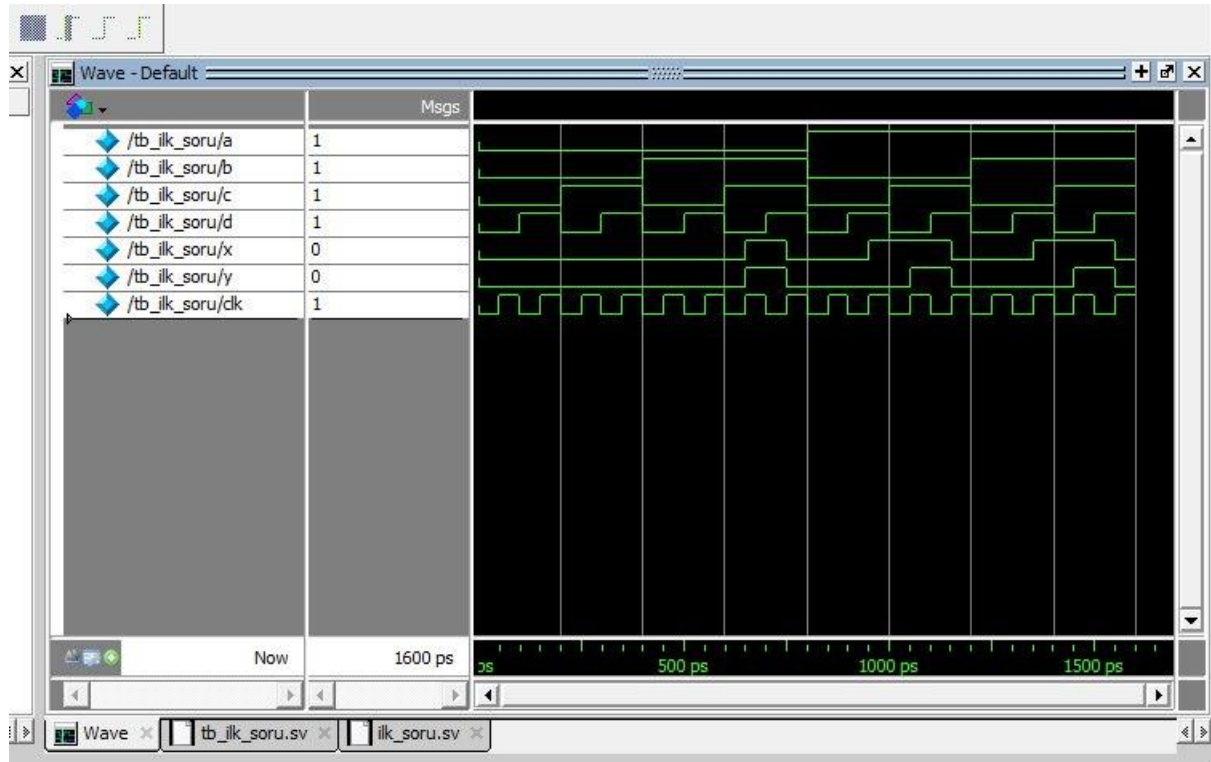
Max frekansı arttırmak için pipeline ayarlanacaktır. "Critical path" de NOT kapısından sonra eklenirse maksimum frekans artmış olur.

$$T_{c1} = 60 + (80 + 35) + 45 + 15 = 235$$

$$T_{c2} = 60 + (50 + 50) + 45 + 15 = 220$$

$$\text{Latency} = 235 + 235 = 470 \text{ ps}$$

$$\text{Throughput} = 1/235$$



Şekil 1 İlk sorunun sinyal analizleri.

```
module ilk_soru(  
    input logic A,B,C,D,CLK,  
    output logic X,Y  
);  
  
    logic f0,f1,f2,f3,n0,n1,n3,p,r;  
  
    assign #80 n0 = (f2^f3);  
    assign #35 n1 = (~n0) ;  
    assign #55 n3 = (f0|f1);  
    assign #50 r = (n3&n1);  
    assign #50 p = (r&f3);  
  
    always_ff @(posedge CLK)  
    begin  
        f0 <= A;  
        f1 <= B;  
        f2 <= C;  
        f3 <= D;  
        X <= r;  
        Y <= p;  
    End  
endmodule
```

```

module tb_ilk_soru ();

    logic a,b,c,d;
    logic x,y;
    logic clk;

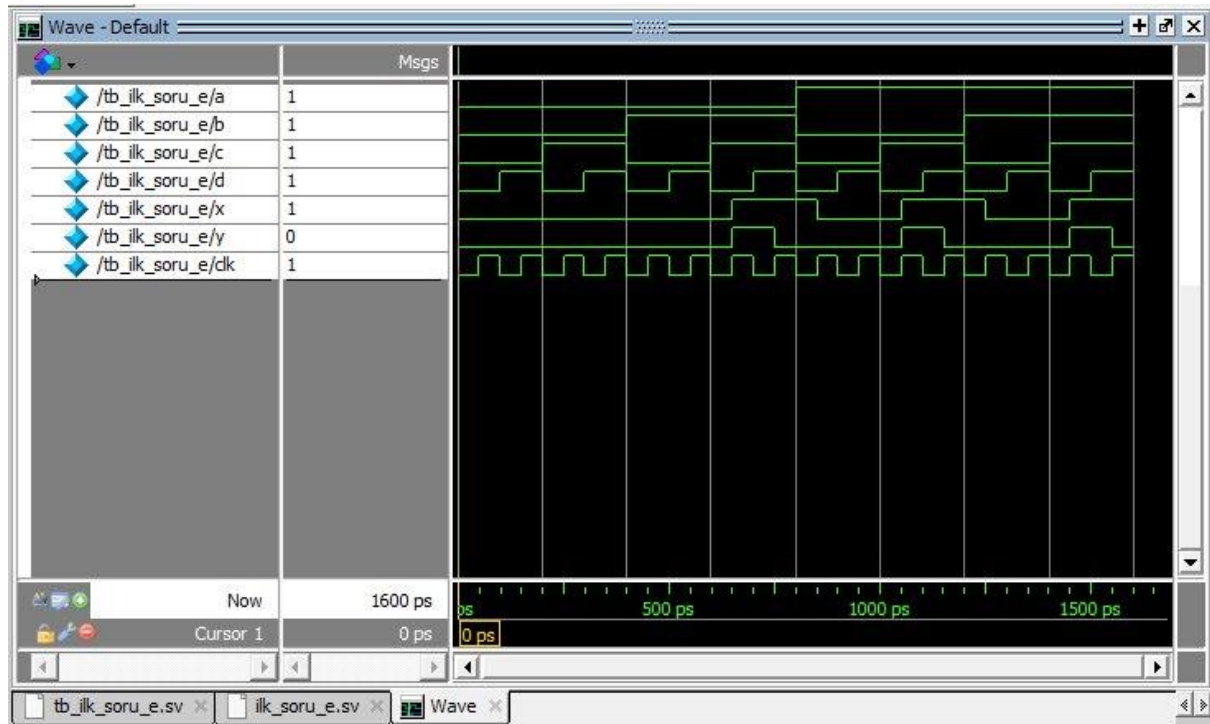
    ilk_soru dut0(.A(a), .B(b), .C(c), .D(d), .X(x), .Y(y), .CLK(clk));

    always begin
        clk = 0; #50; clk=1; #50;
    end

    initial begin
        a=0; b=0; c=0; d=0; x=0; y=0;    #100;
        a=0; b=0; c=0; d=1;              #100;
        a=0; b=0; c=1; d=0;              #100;
        a=0; b=0; c=1; d=1;              #100;
        a=0; b=1; c=0; d=0;              #100;
        a=0; b=1; c=0; d=1;              #100;
        a=0; b=1; c=1; d=0;              #100;
        a=0; b=1; c=1; d=1;              #100;
        a=1; b=0; c=0; d=0;              #100;
        a=1; b=0; c=0; d=1;              #100;
        a=1; b=0; c=1; d=0;              #100;
        a=1; b=0; c=1; d=1;              #100;
        a=1; b=1; c=0; d=0;              #100;
        a=1; b=1; c=0; d=1;              #100;
        a=1; b=1; c=1; d=0;              #100;
        a=1; b=1; c=1; d=1;              #100;
        $stop;
    End

endmodule

```



Şekil 2 İlk soru E şıkının cevabı, yapılan işlem sonrası X ve Y sinyallerinde değişim görüldü.

```
module ilk_soru_e(  
    input logic A,B,C,D,CLK,  
    output logic X,Y);  
  
    logic f0,f1,f2,f3,k,m,n,t,z,k1,n1,h1;  
  
    assign #80 m = (f2^f3);  
    assign #35 n = (~m) ;  
    assign #55 k = (f0|f1);  
    assign #50 z = (k1&n1);  
    assign #50 t = (z&h1);  
    assign #35 h1 = f3;  
  
    always_ff @(posedge CLK)  
  
begin  
    f0 <= A;  
    f1 <= B;  
    f2 <= C;  
    f3 <= D;  
    k1 <= k;  
    n1 <= n;  
    X <= z;  
    Y <= t;  
End  
  
endmodule
```

```

module tb_ilk_soru_e ();

    logic a,b,c,d;
    logic x,y;
    logic clk;

    ilk_soru_e dut0(.A(a), .B(b), .C(c), .D(d), .X(x), .Y(y),
.CLK(clk));

always begin
    clk = 0; #50; clk=1; #50;
end

initial begin
    a=0; b=0; c=0; d=0; x=0; y=0; #100;
    a=0; b=0; c=0; d=1; #100;
    a=0; b=0; c=1; d=0; #100;
    a=0; b=0; c=1; d=1; #100;
    a=0; b=1; c=0; d=0; #100;
    a=0; b=1; c=0; d=1; #100;
    a=0; b=1; c=1; d=0; #100;
    a=0; b=1; c=1; d=1; #100;
    a=1; b=0; c=0; d=0; #100;
    a=1; b=0; c=0; d=1; #100;
    a=1; b=0; c=1; d=0; #100;
    a=1; b=0; c=1; d=1; #100;
    a=1; b=1; c=0; d=0; #100;
    a=1; b=1; c=0; d=1; #100;
    a=1; b=1; c=1; d=0; #100;
    a=1; b=1; c=1; d=1; #100;
    $stop;
End

endmodule

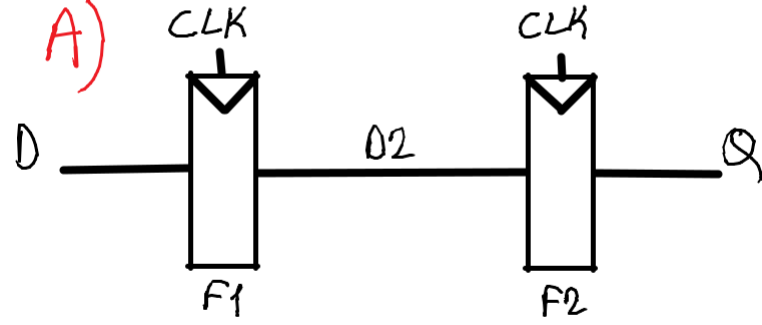
```



## Problem 2. Synchronizer [5 + 5 = 10 puan]

Dışarı bağlanan bir butonu örneklemek için iki FF li bir synchronizer tasarladınız. Gerekliyse Setup ve Hold zamanlarını Tablo 1 deki değerlere göre alabilirsiniz.  $T_0 = 15$  ps,  $t_{hao} = 25$  ps

- A. 10 yıllık bir MTFB (Mean Time Between Failures) için devrenin maximum çalışabileceği frekansı bulunuz.  
B. Frekansı artırmak için **bir FF daha** eklediniz. Aynı MTFB için yeni frekans değerini hesaplayınız.



Kapı	$t_{pd}$ (ps)	$t_{cd}$ (ps)
NOT	35	30
BUFFER	40	35
2-giriş AND	50	50
2-giriş XOR	80	80
2-giriş OR	55	55
3-giriş OR	70	70
all others	90	90
SETUP	45	
HOLD	90	

$$P(\text{failure}) = \frac{T_0}{T_c} e^{-\frac{T_c - t_{setup}}{T}}$$

$$MTFB = \frac{1}{P(\text{failure})/\text{sec}}$$

$$MTFB = \frac{T_c e^{\frac{T_c - t_{setup}}{T}}}{N T_0}$$

$$1 \text{ y.} \rightarrow \pi \times 10^7 \text{ s}$$

$$10 \text{ y.} \rightarrow \pi \times 10^8 \text{ s}$$

$$n=1$$

$$\pi \times 10^8 = \frac{T_c e^{\frac{T_c - 45 \times 10^{-12}}{25 \times 10^{-12}}}}{1 (15 \times 10^{-12})} = 4.49152 \times 10^{-10}$$

B) Bir flip flop daha devreye eklenirse P.P olur

$$MTFB = \frac{1}{P^2/\text{sec}} = 2.22182 \times 10^{-10}$$

### Problem 3. NZVC Hesapları [5 + 10 + 15 = 30 puan]

A. Tablo 2 de verilen 16-bit operasyonlara göre sonuçları ve NZVC bayraklarını hesaplayınız.

	operasyon
a	0x1F32 - 0xFF02
b	0x0010 - 0x1F01
c	0xF801 + 0x7000
d	0xFE02 - 0x00DF
e	0x3D20 + 0xF5D0

Tablo 2

B. ALU devresi sonuçlarını gösteren **bu NZVC bayrakları baz alarak** unsigned HS, LS, HI, ve LO çıkışlarını gösteren devrelerin boolean denklemlerini bulun. Devre girişlerinizi 16-bit olarak alın.

- HS =  $A \geq B$  (A, B ye eşit veya daha büyük. High/Same)
- LS =  $A \leq B$  (A, B ye eşit veya daha küçük. Low/Same)
- HI =  $A > B$  (A, B den daha büyük. High)
- LO =  $A < B$  (A, B den daha küçük. Low)

C. HDL kullanarak tasarlayın, bir testbench devresi oluşturun ve Modelsim de simüle edin. Bunun için A,B girişlerinden NZVC bayraklarını generate eden bir devre tasarlayıp, testbench devrenizde bu A, B girişlerine değer gönderin.

A.a)

0x	<del>1</del>	<del>F</del>	<del>3</del>	<del>2</del>
Binary	0001	1111	0011	0010
0x	<del>F</del>	<del>F</del>	<del>0</del>	<del>2</del>
Binary	1111	1111	0000	0010
Two's Complement	0000	0000	1111	1110
Result	0010	0000	0011	0000

N	0
Z	0
V	0
C	0

A.b)

0x	<del>0</del>	<del>0</del>	<del>1</del>	<del>0</del>
Binary	0000	0000	0001	0000
0x	<del>1</del>	<del>F</del>	<del>0</del>	<del>1</del>
Binary	0001	1111	0000	0001
Two's Complement	1110	0000	1111	1111
Result	1110	0001	0000	1111

N	1
Z	0
V	0
C	0

A.c)

<del>Ox</del>	<del>F</del>	<del>8</del>	<del>0</del>	<del>1</del>
Binary	1111	1000	0000	0001
<del>Ox</del>	<del>7</del>	<del>0</del>	<del>0</del>	<del>0</del>
Binary	0111	0000	0000	0000
Result	1011	0100	0000	0001

N	1
Z	0
V	0
C	0

A.d)

<del>Ox</del>	<del>F</del>	<del>E</del>	<del>0</del>	<del>2</del>
Binary	1111	1110	0000	0010
<del>Ox</del>	<del>0</del>	<del>0</del>	<del>0</del>	<del>F</del>
Binary	0000	0000	1101	1111
Two's Complement	1111	1111	0010	0001
Result	1111	1101	0010	0011

N	1
Z	0
V	0
C	1

A.e)

<del>Ox</del>	<del>B</del>	<del>0</del>	<del>2</del>	<del>0</del>
Binary	0011	1101	0010	0000
<del>Ox</del>	<del>F</del>	<del>5</del>	<del>0</del>	<del>0</del>
Binary	1111	0101	1101	0000
Result	0011	0010	1111	0000

N	0
Z	0
V	0
C	1

```

module alu(input logic n, z, v, c,
           output logic hs, ls, hi, lo
);
    assign hs = c;
    assign ls = z | ~c;
    assign hi = ~(z | ~c);
    assign lo = ~c;
endmodule

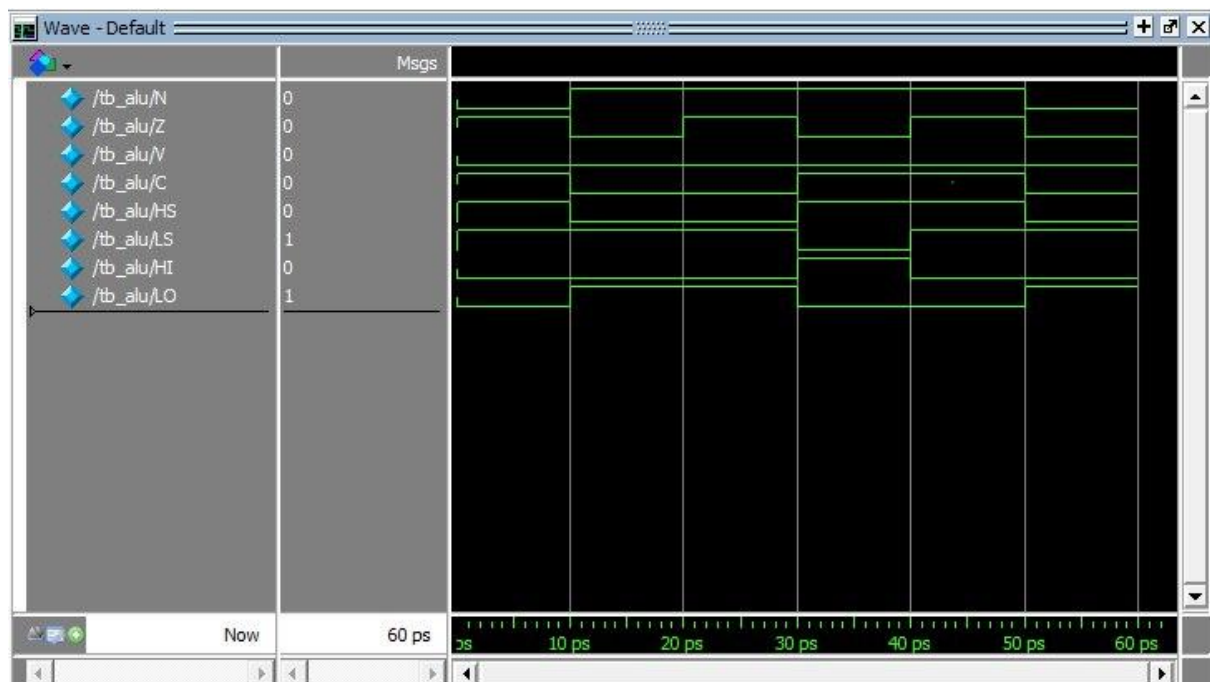
```

```

module tb_alu ();

    logic N, Z, V, C;
    logic HS, LS, HI, LO;
    alu dut0(.n(N), .z(Z), .v(V), .c(C), .hs(HS), .ls(LS), .hi(HI),
    .lo(LO));
    initial begin
        HI = 0; HS = 0; LS = 0; LO = 0;
        Z = 1; C = 1; N = 0; V = 0;      #10
        Z = 0; C = 0; N = 1;             #10
        Z = 1;                           #10
        Z = 0; C = 1;                     #10
        Z = 1;                           #10
        Z = 0; C = 0; N = 0; V = 0;      #10
    $stop;
    end
endmodule

```



Şekil 3 ALU Devresi çıktıları

#### Problem 4. Multiplier tasarımı [12 + 12 = 24 puan]

- A. AND kapıları ve full adder lar kullanarak iki unsigned 5-bitlik binary sayıyı çarpan bir devre tasarlayınız. Bu devrenin propagation ve contamination gecikmelerini Tablo 1 de verilen değerlere göre hesaplayınız.
- B. HDL kullanarak bu devreyi tasarlayın, bir testbench devresi oluşturun ve Modelsim de simüle edin. Doğrulamayı otomatik olarak \* operatörü kullanan ayrı bir devre ile karşılaştırın.
- Quartus'da bu devreyi sentezleyerek bu devrede ulaşabileceğiniz max frekansını bulun. Bunun için giriş ve çıkışlara register bağlayabilirsiniz.
  - Devrenizi, \* operatörü kullanan ayrı bir devre ile sonuçlarınızı karşılaştırın. (i.e. assign c = a \* b;), max freq, utilization.

Sorunun A şıkkı en son sayfalardadır.



Şekil 4 İki devre ile birlikte aynı sayılar aynı testbench'e koyulduğu zaman ikisinde de çıkacak sonuç.

```
/* Kontrolün yapılması için gerekli olan multiplier kodu*/  
  
module multiplier_c  
    (input logic [4:0] a, b,  
     output logic [9:0] q);  
  
    assign q= a * b;  
  
endmodule
```

```

module multiplier
    (input [4:0] a, b,
     output [9:0] q);

    wire [4:0] n0;
    wire [5:0] n1;
    wire [6:0] n2;
    wire [7:0] n3;
    wire [8:0] n4;
    wire [9:0] s1,s2,s3,s4;

    assign n0 = {5{a[0]}} & b[4:0];
    assign n1 = {5{a[1]}} & b[4:0];
    assign n2 = {5{a[2]}} & b[4:0];
    assign n3 = {5{a[3]}} & b[4:0];
    assign n4 = {5{a[4]}} & b[4:0];

    assign s1 = n0 + (n1 << 1);
    assign s2 = s1 + (n2 << 2);
    assign s3 = s2 + (n3 << 3);
    assign s4 = s3 + (n4 << 4);

    assign q = s4;

endmodule

```

Yukarıdaki iki kodun testbench'i.

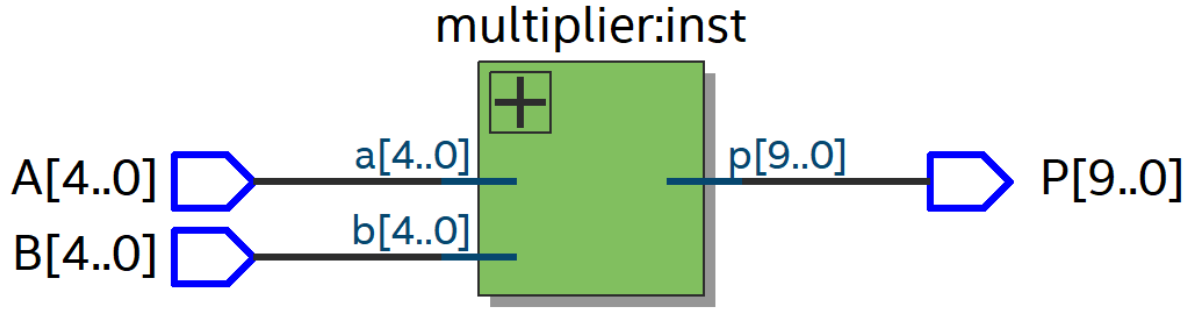
```

module tb_multiplier();
    reg[4:0] A,B;
    wire[9:0] R;

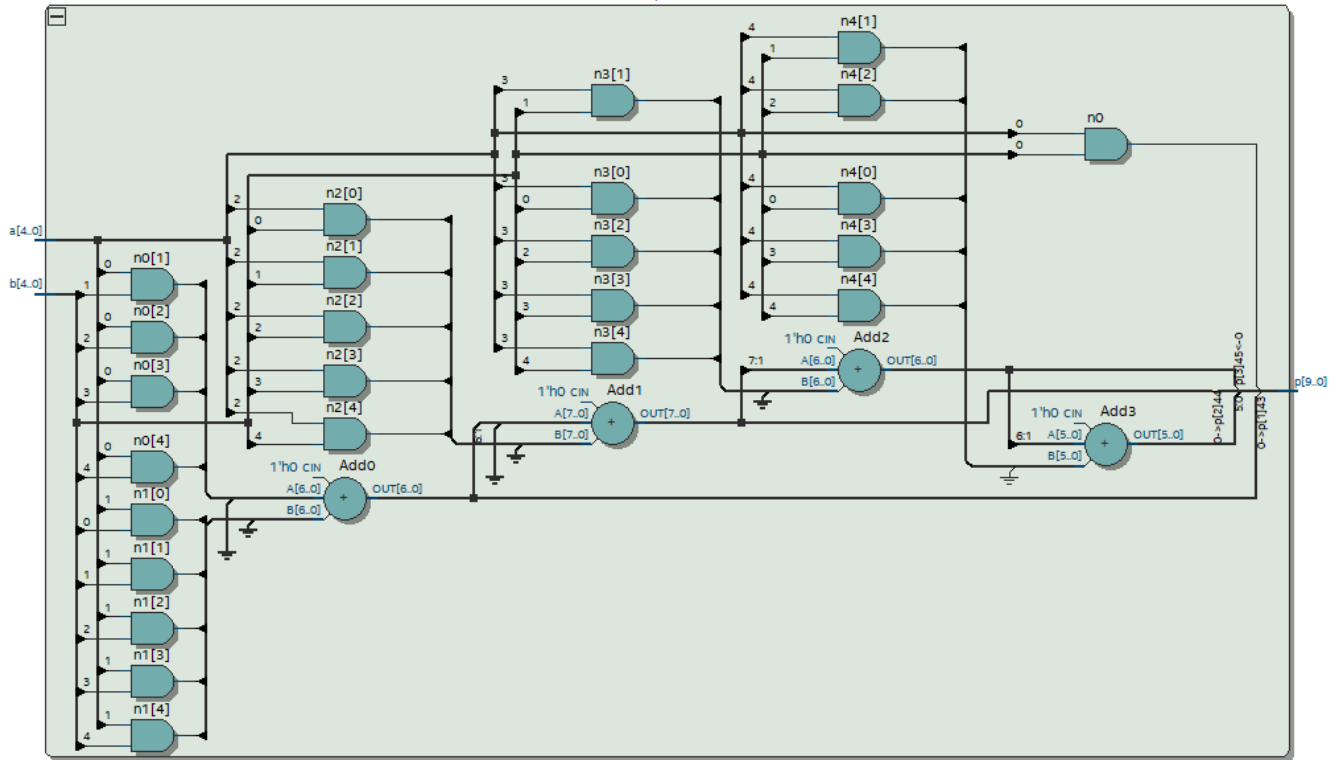
    multiplier_c mm1(.a(A), .b(B), .q(R));

    initial begin
        A = 5'b11011;    B = 5'd3;        #50;
        A = 23;           B = 4;           #50;
        A = 5'b11111;    B = 5'b11111;    #50;
        $stop;
    end
endmodule

```



Şekil 5 Devrenin Quartus'ta sentezlenmiş hali.



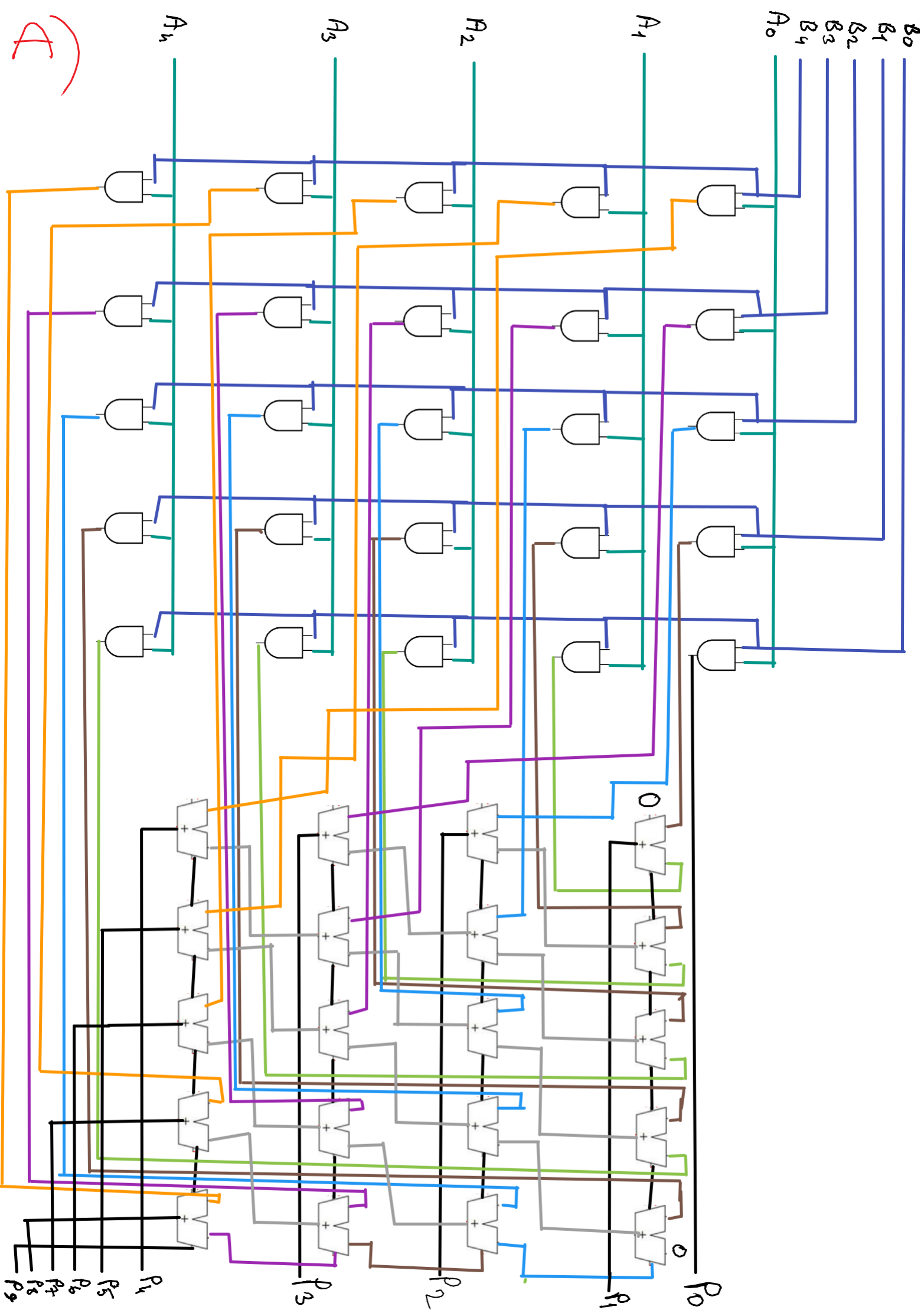
Şekil 6 Multiplier:inst'in iç görünümü (Quartusta)

Slow 1200mV 85C Model Fmax Summary

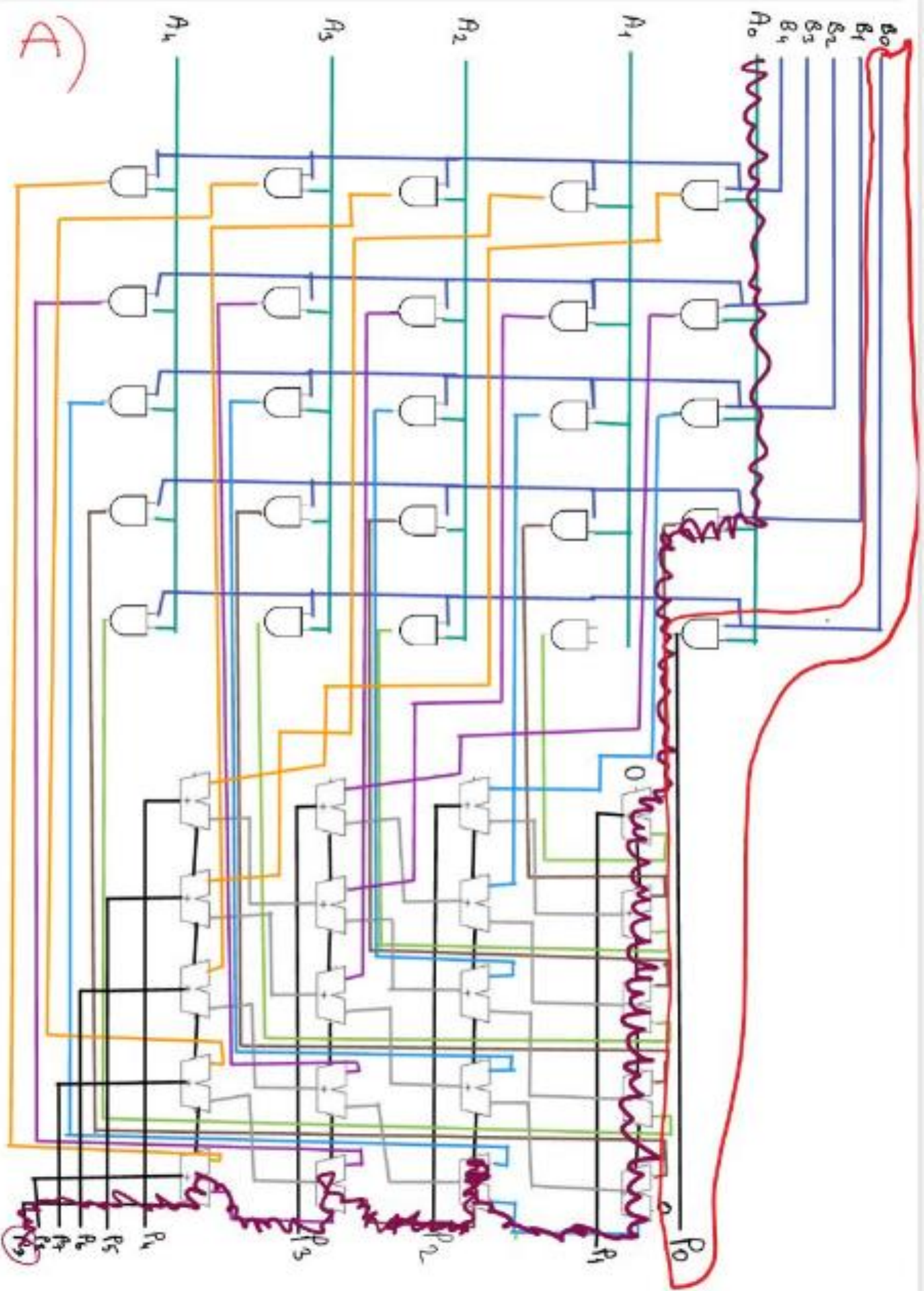
No paths to report.


Şekil 7 Quartus Timing analysis'de karşılaştığım hata.

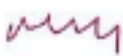
Quartusta Fmax'ı bulmak için uğraşırken yukarıdaki gibi bir ekran ile karşılaştım. Sebebinin tam olarak bilmiyorum ancak devre üzerinde clock bağlamadığım için olduğunu tahmin ediyorum. Clock bağlayarak devreyi yapmaya çalıştığımda komple bozulduğu için CLK'lı versiyonu koymadım buraya.







 short path

 critical path

short path sadece 1 and var;

50 ps

critical path 1 and 8 full adder var

$$50 + 8 \cdot 90 = 770 \text{ ps}$$

Short path B0 - P0 = 50 ps;

Critical path A0 - P9 = 770 ps bulunmuştur.