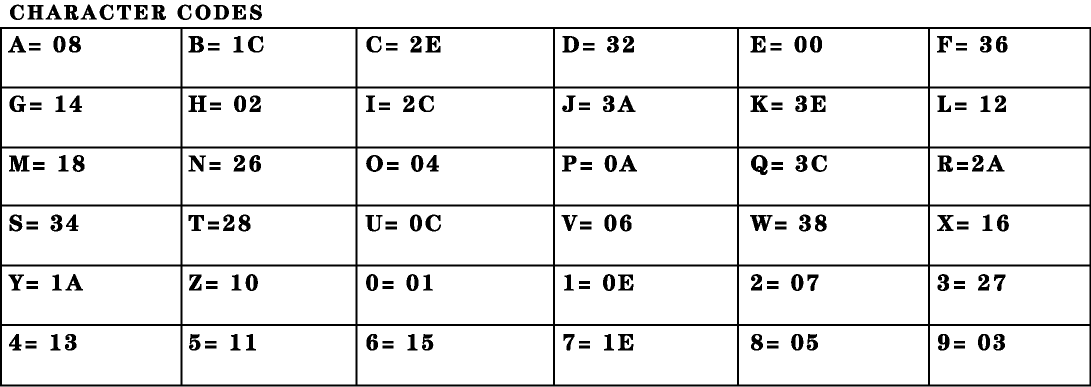
**LOJIK DEVRE TASARIMI PROJE RAPORU**

**Grup Üyeleri**

1358130030 Anıl Ertürk

**Proje amacı:** Bir sayısal bilgisayarda alfabenin karakterleri (A,….Z), 10 tane onlu sayı (0,1,2…9) 6 bitlik bir kod kullanılarak kodlanabilir(düşük anlamlı 4 bit sağ hex karakteri ifade eder 24 =16. Yüksek anlamlı 2 bit sol hex karakterimizin ilk iki bitini ifade eder. En sol iki biti sıfırdır). Bu kod tablosu (hex karşılıkları) aşağıda görülmektedir.



Projenin amacı 6 bitlik bir giriş kodunun bir alfabe karakteri, bir rakam veya geçersiz bir kod olup olmadıgını bulan bir lojik devre tasarlamaktır. Devrenin çıkışları F1, F2 ve F3 olarak alınırsa, aşağıda çıkış fonksiyonları verilmiştir.

F1 = 1 , F2=0 , F3=0 ise, Alfabetik Karakter Kodu (A,B,…Z)

F1 = 0 , F2=1 , F3=0 ise, Nümerik karakter (Rakamlar, 0,1,…9)

F1 = K , F2=K , F3=1 ise, Tanımsız karakter

K: keyfi 0/1

**Doğruluk tablosu:** 6 girişimiz var. 3 çıkışımız var. 26 = 64 olduğu için 64 farklı inputumuz olabilir. Bu 64 inputu ve karşılık geldiği 3 olaydan biri hangisi ise önce doğruluk tablomuza yazıyoruz.

Not: Son iki sütun doğruluk tablosunun parçası değil. Sadece insanlar daha kolay kullanabilsin diye eklenmiş notlar. Ayrıca en üstteki satır data değil header satırı.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A5 | A4 | A3 | A2 | A1 | A0 | F1 | F2 | F3 |  |  |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | E | 00 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 01 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | H | 02 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 9 | 03 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | O | 04 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 8 | 05 |
| 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | V | 06 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 2 | 07 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | A | 08 |
| 0 | 0 | 1 | 0 | 0 | 1 | K | K | 1 |  | 09 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | P | 0A |
| 0 | 0 | 1 | 0 | 1 | 1 | K | K | 1 |  | 0B |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | U | 0C |
| 0 | 0 | 1 | 1 | 0 | 1 | K | K | 1 |  | 0D |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0E |
| 0 | 0 | 1 | 1 | 1 | 1 | K | K | 1 |  | 0F |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | Z | 10 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 5 | 11 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | L | 12 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 4 | 13 |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | G | 14 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 6 | 15 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | X | 16 |
| 0 | 1 | 0 | 1 | 1 | 1 | K | K | 1 |  | 17 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | M | 18 |
| 0 | 1 | 1 | 0 | 0 | 1 | K | K | 1 |  | 19 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | Y | 1A |
| 0 | 1 | 1 | 0 | 1 | 1 | K | K | 1 |  | 1B |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | B | 1C |
| 0 | 1 | 1 | 1 | 0 | 1 | K | K | 1 |  | 1D |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 7 | 1E |
| 0 | 1 | 1 | 1 | 1 | 1 | K | K | 1 |  | 1F |
| 1 | 0 | 0 | 0 | 0 | 0 | K | K | 1 |  | 20 |
| 1 | 0 | 0 | 0 | 0 | 1 | K | K | 1 |  | 21 |
| 1 | 0 | 0 | 0 | 1 | 0 | K | K | 1 |  | 22 |
| 1 | 0 | 0 | 0 | 1 | 1 | K | K | 1 |  | 23 |
| 1 | 0 | 0 | 1 | 0 | 0 | K | K | 1 |  | 24 |
| 1 | 0 | 0 | 1 | 0 | 1 | K | K | 1 |  | 25 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | N | 26 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 3 | 27 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | T | 28 |
| 1 | 0 | 1 | 0 | 0 | 1 | K | K | 1 |  | 29 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | R | 2A |
| 1 | 0 | 1 | 0 | 1 | 1 | K | K | 1 |  | 2B |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | I | 2C |
| 1 | 0 | 1 | 1 | 0 | 1 | K | K | 1 |  | 2D |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | C | 2E |
| 1 | 0 | 1 | 1 | 1 | 1 | K | K | 1 |  | 2F |
| 1 | 1 | 0 | 0 | 0 | 0 | K | K | 1 |  | 30 |
| 1 | 1 | 0 | 0 | 0 | 1 | K | K | 1 |  | 31 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | D | 32 |
| 1 | 1 | 0 | 0 | 1 | 1 | K | K | 1 |  | 33 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | S | 34 |
| 1 | 1 | 0 | 1 | 0 | 1 | K | K | 1 |  | 35 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | F | 36 |
| 1 | 1 | 0 | 1 | 1 | 1 | K | K | 1 |  | 37 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | W | 38 |
| 1 | 1 | 1 | 0 | 0 | 1 | K | K | 1 |  | 39 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | J | 3A |
| 1 | 1 | 1 | 0 | 1 | 1 | K | K | 1 |  | 3B |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | Q | 3C |
| 1 | 1 | 1 | 1 | 0 | 1 | K | K | 1 |  | 3D |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | K | 3E |
| 1 | 1 | 1 | 1 | 1 | 1 | K | K | 1 |  | 3F |

**Kagnaugh Maps:** Şimdi doğruluk tablosunun F1, F2, F3 sütunlarınındaki 1 olan kısımların hangi inputlara karşılık geldiğini kullanarak F1, F2, F3 fonksiyonlarının denklem olarak inputların hangi kombinasyonuna karşlılık geldiğini yazmamız gerekiyor. Tabiki tek tek tüm 1 sinyallerini yazmıyoruz. Sadeleştirerek yazıyoruz denklemlerimizi.

Not1: Aşağıdaki tablolarda ilk satır ve ilk sütun data değil header satır ve sütünu.

Not2: 6 Değişkenli k map sadeleştirmesi yaparken S1 S2 ye bitişik, S2 S4 e bitişik, S4 S3 e bitişik S3 S1 e bitişik gibi davranıyoruz. Yani ya tablolar üzerinde tek başına sadeleştirme yapabiliriz, ya bitişik çift tabloları kullanabiliriz yada tüm tabloları birden(4) kullanabiliriz.

**F1 Tabloları**

A5 = 0, A4 = 0 Stack 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 0 | 1 |
| 01 | 1 | 0 | 0 | 1 |
| 11 | 1 | K | K | 0 |
| 10 | 1 | K | K | 1 |

A5 = 0, A4 = 1 Stack 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 0 | 1 |
| 01 | 1 | 0 | K | 1 |
| 11 | 1 | K | K | 0 |
| 10 | 1 | K | K | 1 |

A5 = 1, A4 = 1 Stack 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | K | K | K | 1 |
| 01 | 1 | K | K | 1 |
| 11 | 1 | K | K | 1 |
| 10 | 1 | K | K | 1 |

A5 = 1, A4 = 0 Stack 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | K | K | K | K |
| 01 | K | K | 0 | 1 |
| 11 | 1 | K | K | 1 |
| 10 | 1 | K | K | 1 |

**F2 Tabloları**

A5 = 0, A4 = 0 Stack 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 1 | 0 |
| 01 | 0 | 1 | 1 | 0 |
| 11 | 0 | K | K | 1 |
| 10 | 0 | K | K | 0 |

A5 = 0, A4 = 1 Stack 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 1 | 0 |
| 01 | 0 | 1 | K | 0 |
| 11 | 0 | K | K | 1 |
| 10 | 0 | K | K | 0 |

A5 = 1, A4 = 1 Stack 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | K | K | K | 0 |
| 01 | 0 | K | K | 0 |
| 11 | 0 | K | K | 0 |
| 10 | 0 | K | K | 0 |

A5 = 1, A4 = 0 Stack 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | K | K | K | K |
| 01 | K | K | 1 | 0 |
| 11 | 0 | K | K | 0 |
| 10 | 0 | K | K | 0 |

**F3 Tabloları**

A5 = 0, A4 = 0 Stack 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 0 | 0 |
| 11 | 0 | 1 | 1 | 0 |
| 10 | 0 | 1 | 1 | 0 |

A5 = 0, A4 = 1 Stack 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 1 | 0 |
| 11 | 0 | 1 | 1 | 0 |
| 10 | 0 | 1 | 1 | 0 |

A5 = 1, A4 = 1 Stack 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | 1 | 0 |
| 01 | 0 | 1 | 0 | 1 |
| 11 | 0 | 1 | 1 | 0 |
| 10 | 0 | 1 | 1 | 0 |

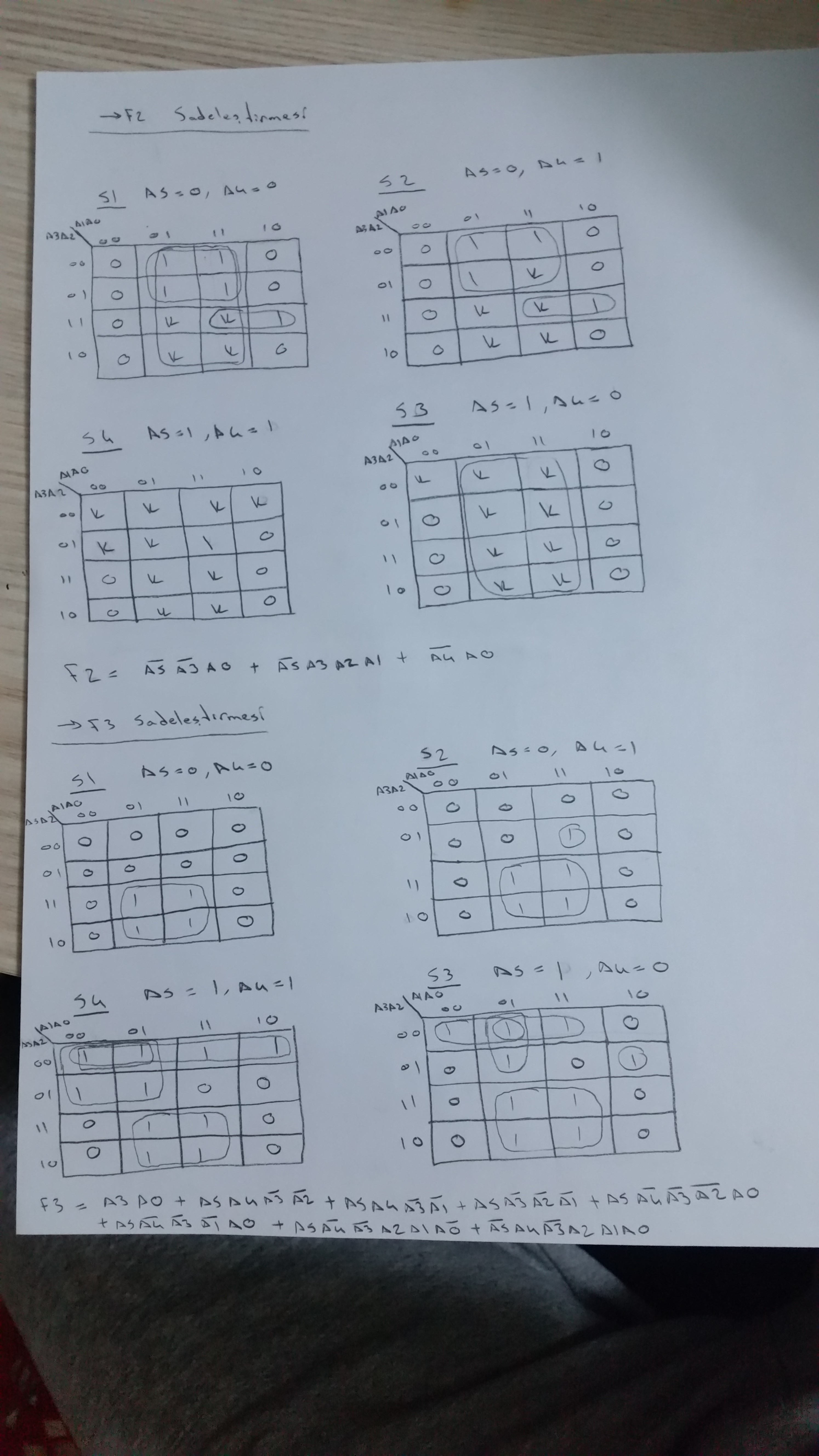
A5 = 1, A4 = 0 Stack 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1A0  A3A2 | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | 1 | 1 |
| 01 | 1 | 1 | 0 | 0 |
| 11 | 0 | 1 | 1 | 0 |
| 10 | 0 | 1 | 1 | 0 |

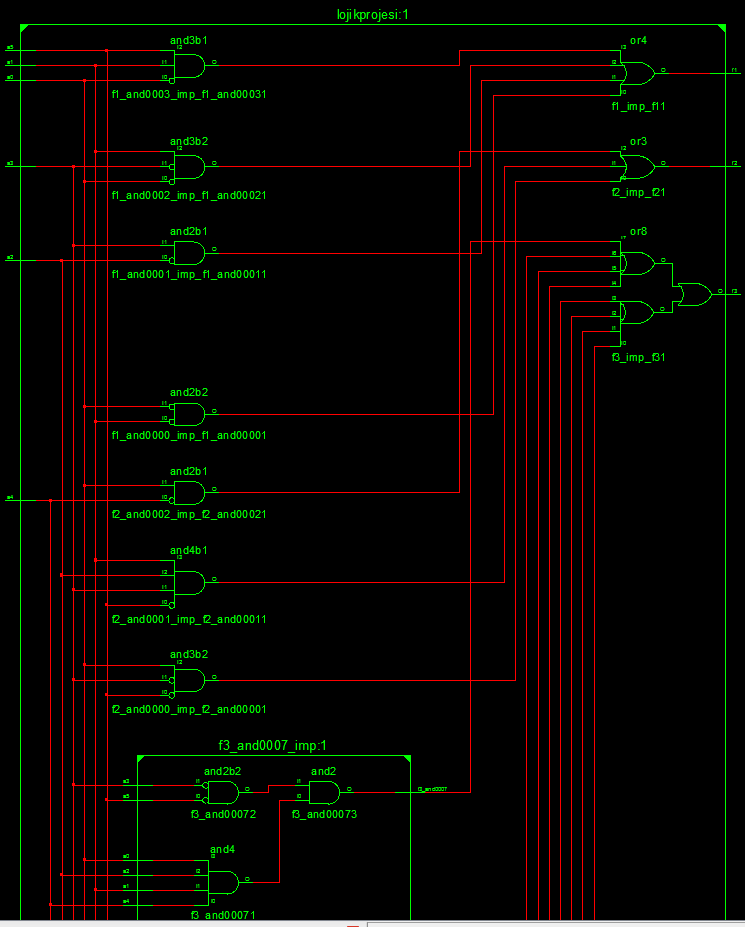
**Sadeleştirme:** Kağıt kalem ile daha iyi göstereceğim için ovalleri kağıt üzerinde yaptım bu işlemleri. Aşağıda görebildiğiniz gibi F1, F2, F3 için ayrı ayrı 6 değişkenli sadeleştirme için 4 tablo çiziyoruz. Ve bulabildiğimiz en küçük(verimli) denklemi bulmaya çalışıyoruz.

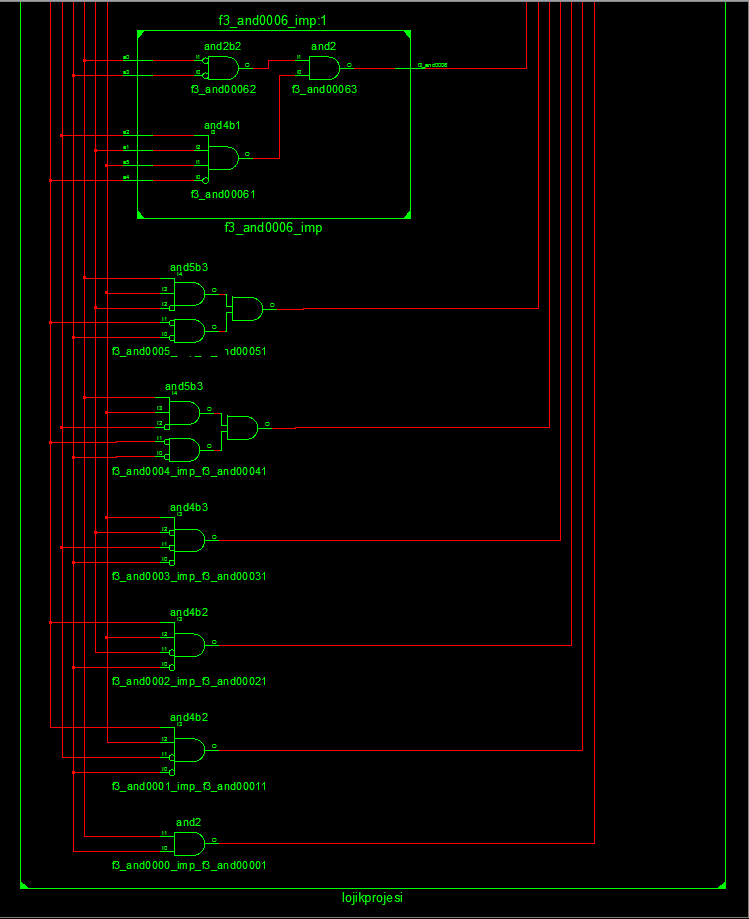


F3 için K olmaması onun denkleminin çok daha büyük olmasına sebep oldu.



**Devre:** Istedigimiz programi kullanabildigimiz icin devreyi Xilinx ISE design studio 14.7 programiyla kurdum.



****

**Devrenin kodu**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity lojikprojesi is

Port (

a5,a4,a3,a2,a1,a0 : in STD\_LOGIC;

f1,f2,f3 : out STD\_LOGIC

);

end lojikprojesi;

architecture dataflow of lojikprojesi is

begin

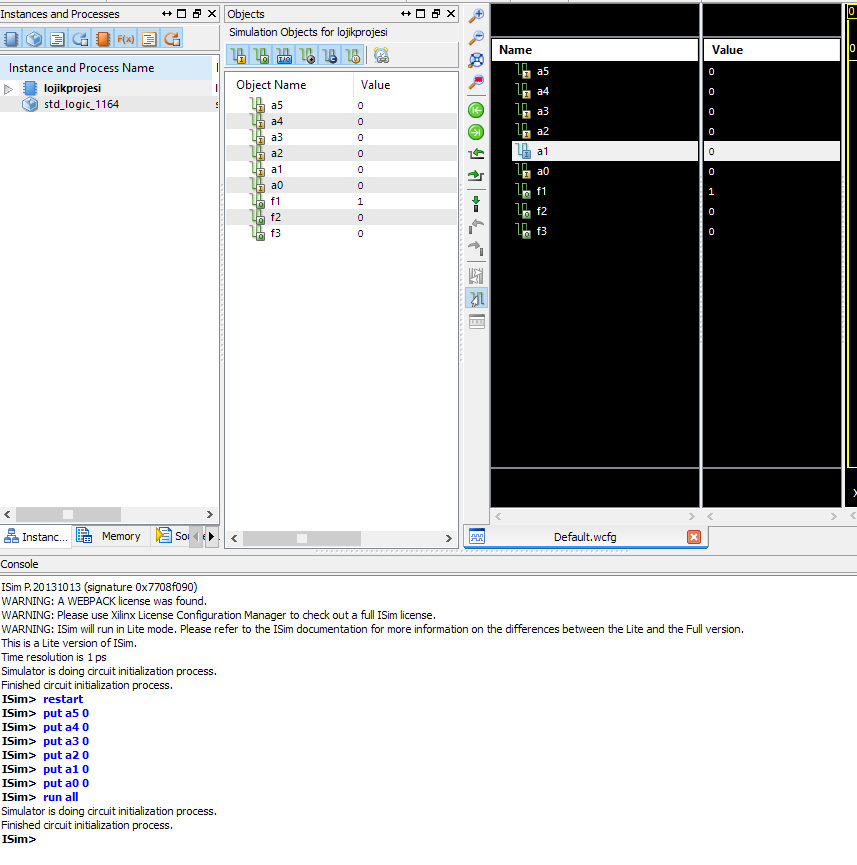
f1 <= (not a1 and not a0) or (a3 and not a2) or (not a3 and a1 and not a0) or (a5 and a1 and not a0);

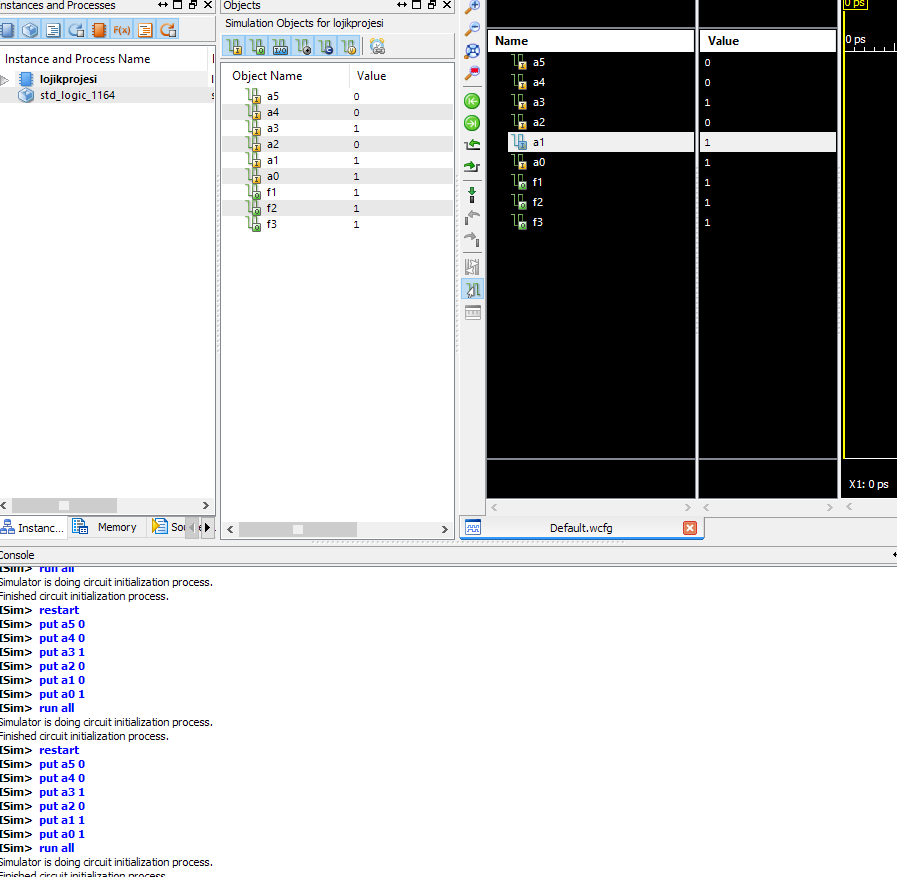
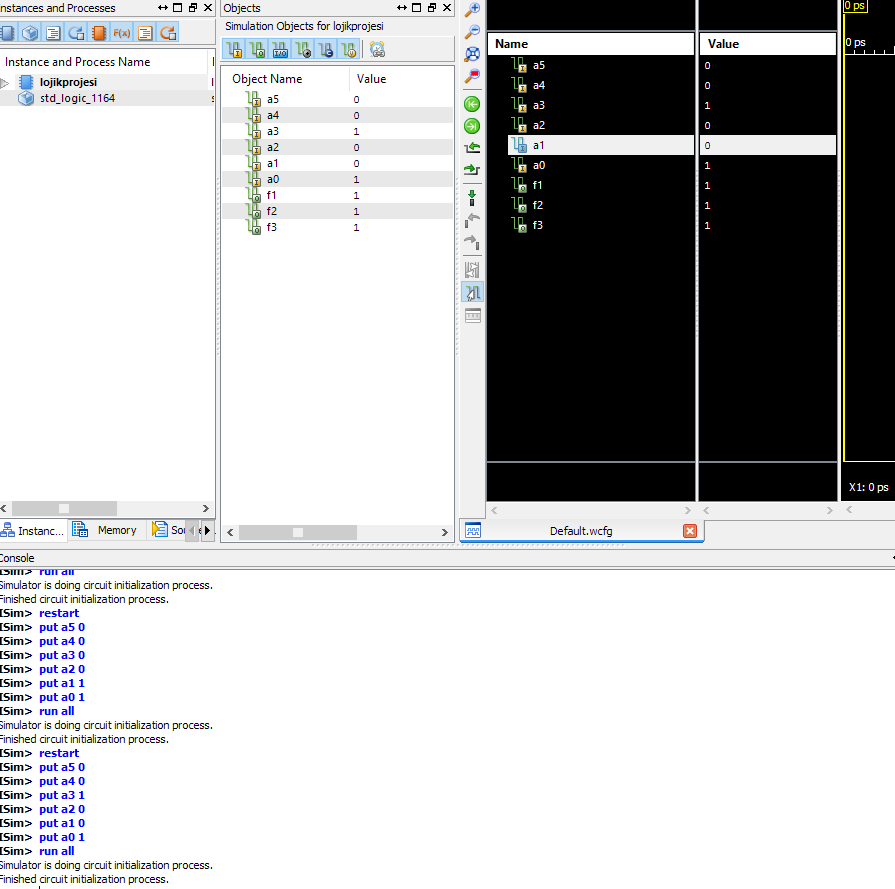
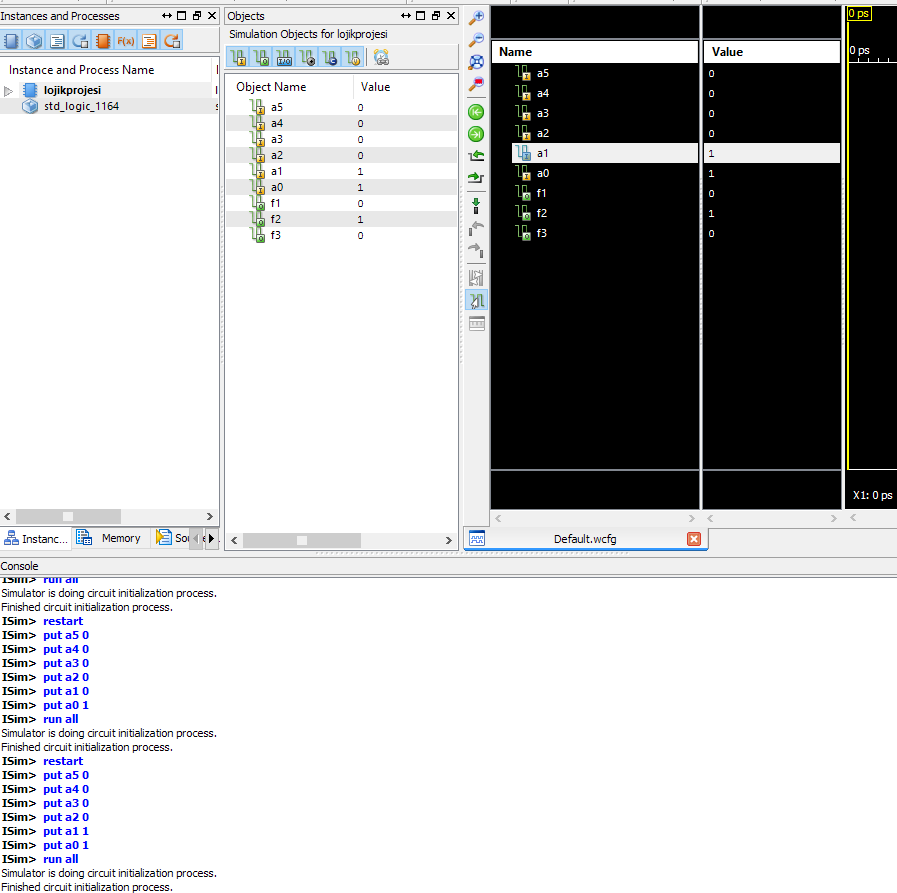
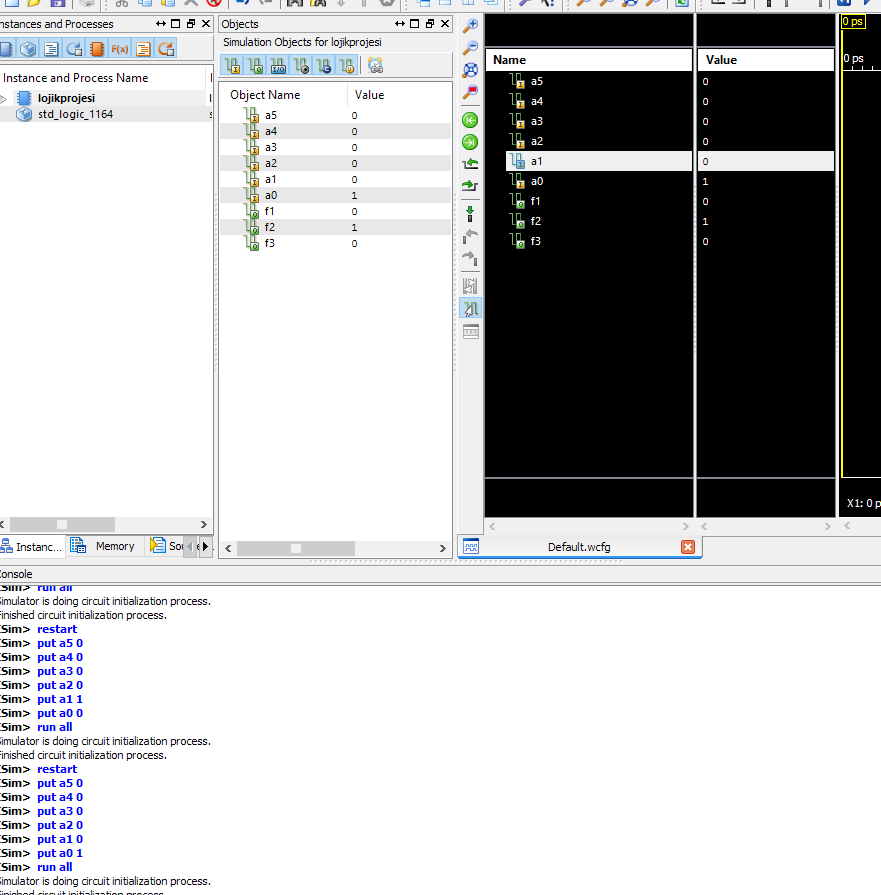
f2 <= (not a5 and not a3 and a0) or (not a5 and a3 and a2 and a1) or (not a4 and a0);

f3 <= (a3 and a0) or (a5 and a4 and not a3 and not a2) or (a5 and a4 and not a3 and not a1) or (a5 and not a3 and not a2 and not a1) or (a5 and not a4 and not a3 and not a2 and a0) or (a5 and not a4 and not a3 and not a1 and a0) or (a5 and not a4 and not a3 and a2 and a1 and not a0) or (not a5 and a4 and not a3 and a2 and a1 and a0);

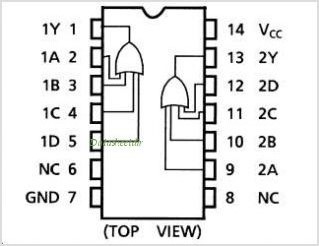
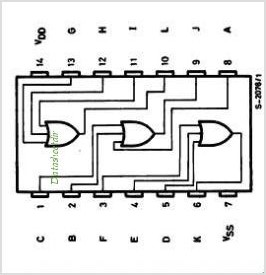
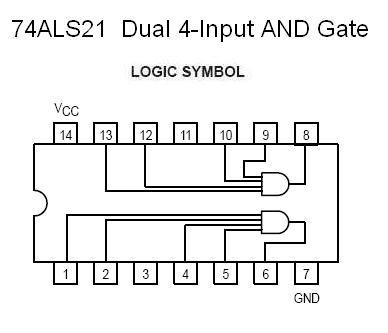
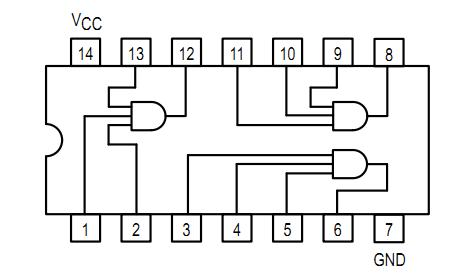
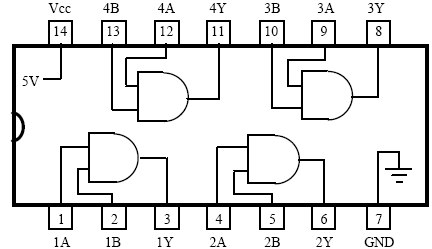
end dataflow;

**Waveform örnekleri** İlk görüntüde 00 ve 02 hex kodlarını giriyoruz ve E ve H harfleri yani f1 çıkışları alıyoruz. İkinci görüntüde 01 ve 03 hex kodlarını giriyoruz ve 0 ve 9 sayılarını yani f2 çıkışları alıyoruz. Üçüncü örnekte 09 ve 0B hex kodlarını giriyoruz ve wrong input yani f3 çıkışı alıyoruz. Sonraki üç sayfada göreceğiniz üzere ilk iki görüntüde f1 sonraki iki görüntüde f2 sonraki iki görüntüde f3 sonucu aldık.

****



**Devre katalog:** Not kapısı, 2-3-4 girişli And kapısı, 2-3-4 girişli Or kapısı.



**Devre maliyeti:**

Not kapisi sayisi: 28(kullanilan), 30(toplam)

Not entegre sayısı(74LS04): 5(5 x 6 = 30 gate)

2 girişli and kapısı sayısı: 12(kullanilan), 12(toplam)

2 girişli and entegre sayisi(74LS08): 3(3 x 4 = 12 gate)

3 girişli and kapısı sayısı: 5(kullanilan), 6(toplam)

3 girişli and entegre sayisi(74LS11): 2(2 x 3 = 6 gate)

4 girişli and kapısı sayısı: 6(kullanilan), 6(toplam)

4 girişli and entegre sayisi(74ALS21): 3(3 x 2 = 6 gate)

2 girişli or kapısı sayısı: 1(kullanilan), 4(toplam)

2 girişli or entegre sayisi(74LS32): 1(1 x 4 = 4 gate)

3 girişli or kapısı sayısı: 1(kullanilan), 3(toplam)

3 girişli or entegre sayisi(HCF407): 1(1 x 3 = 3 gate)

4 girişli or kapısı sayısı: 3(kullanilan), 4(toplam)

4 girişli or entegre sayisi(TC74HC4072AF): 2(2 x 2 = 4 gate)