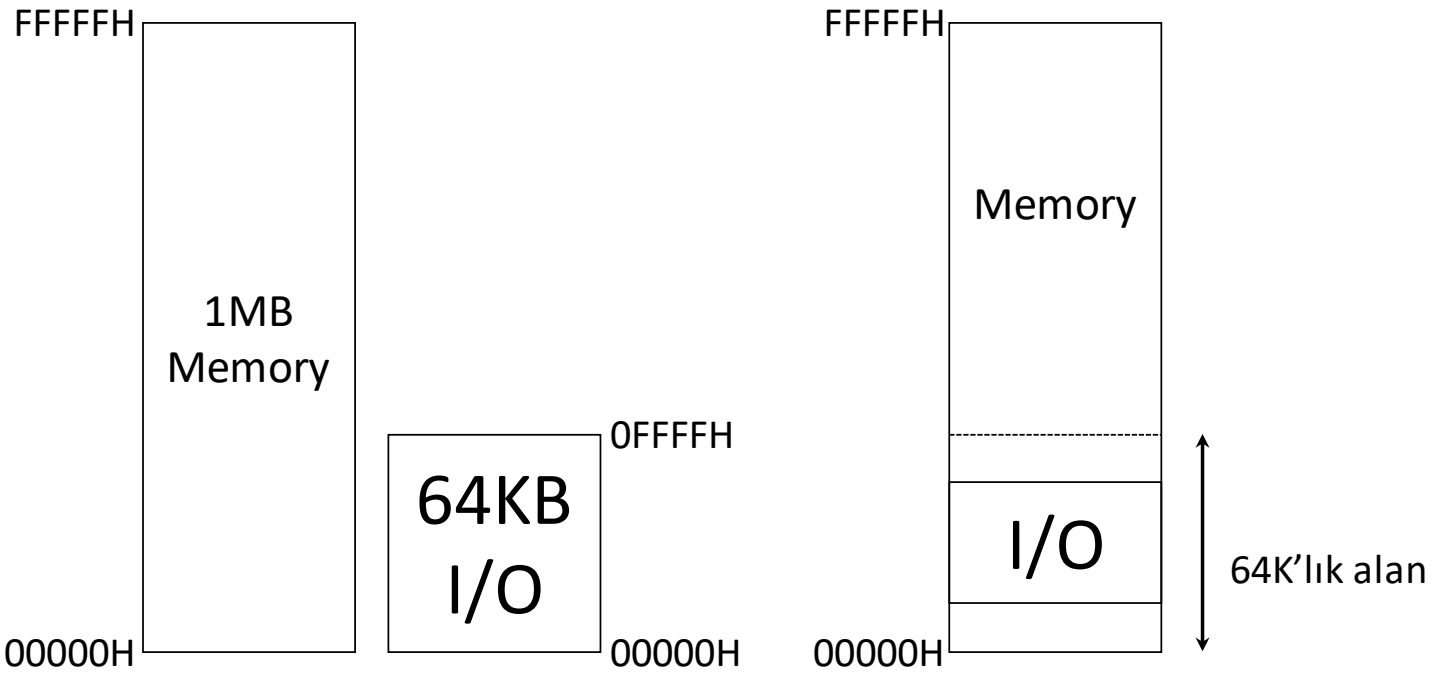
G/Ç Haritalama Yöntemleri

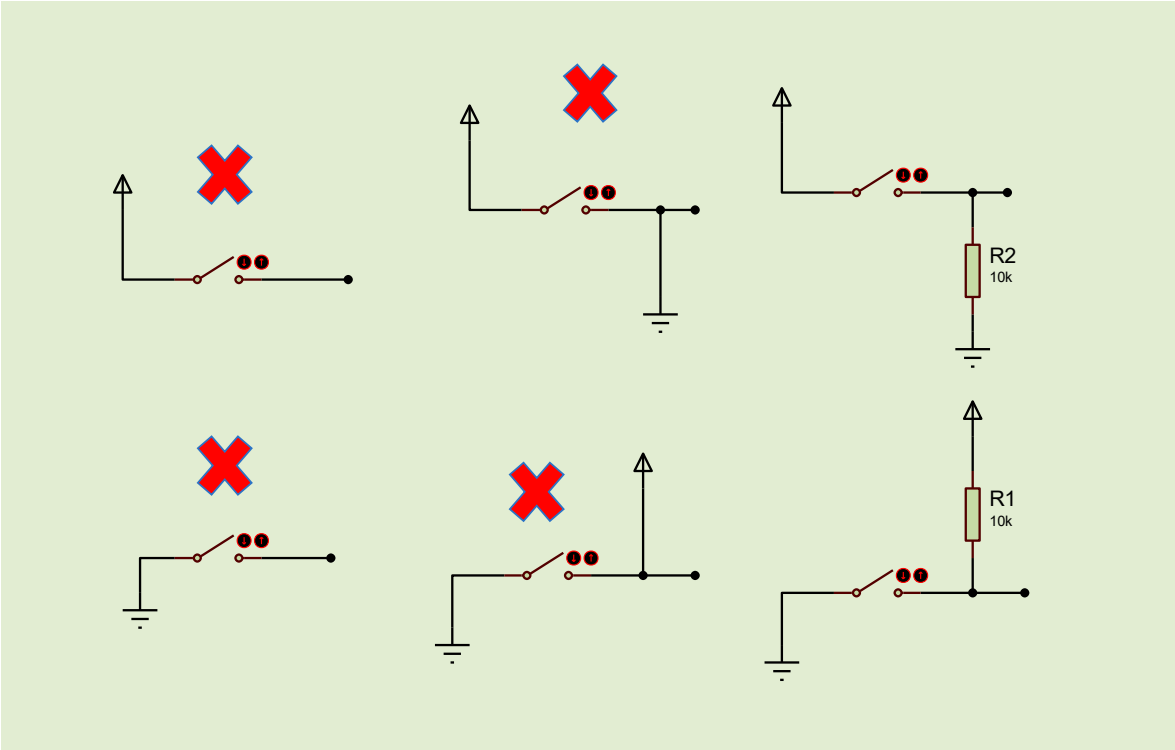
- Isolated I/O Mapping ➔ $M/\overline{IO}$ ucu adres çözümleme için kullanılır
- Memory Mapped I/O ➔ $M/\overline{IO}$ ucu adres çözümleme için kullanılmaz

Isolated I/O – Memory Mapped I/O

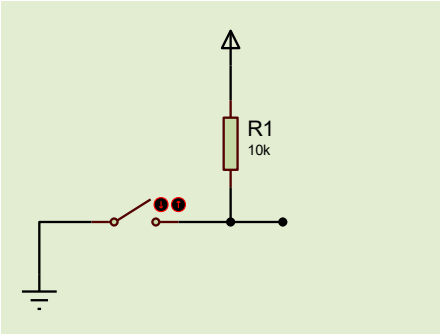


Düğme Arayüzü (Basit Arayüz Devreleri)

- Basit giriş arayüzü olarak kullanılabilir
- Basılmadığı durumda geçerli lojik bir seviye üretmelidir
- Kontakt gürültüne karşı yazılımsal veya donanımsal önlem gereklidir



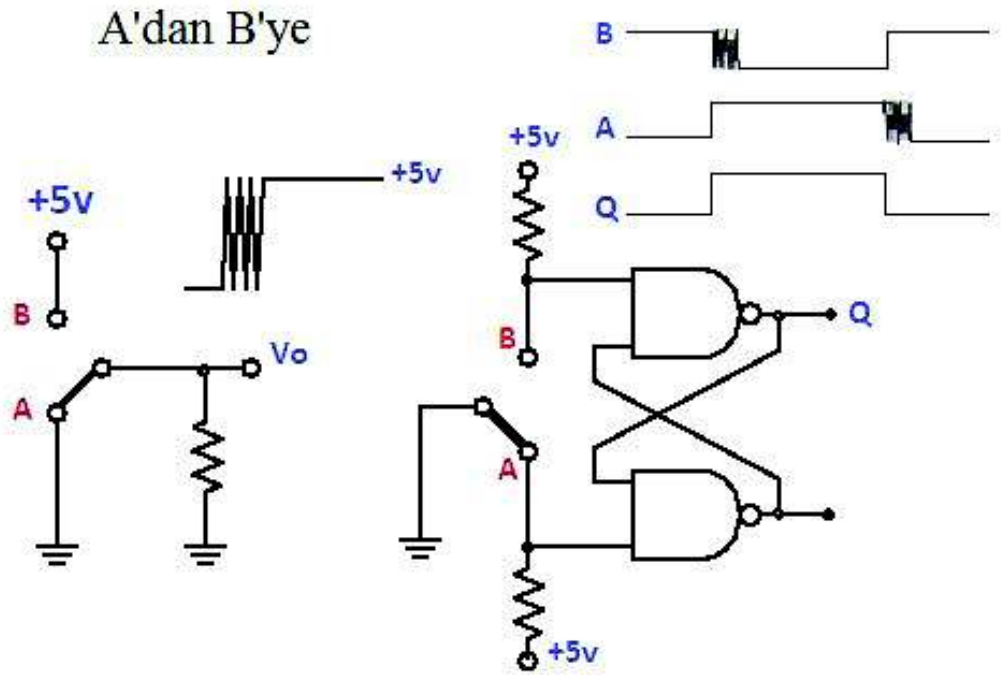
Düğme Arayüzü – Kontakt Gürültüsü



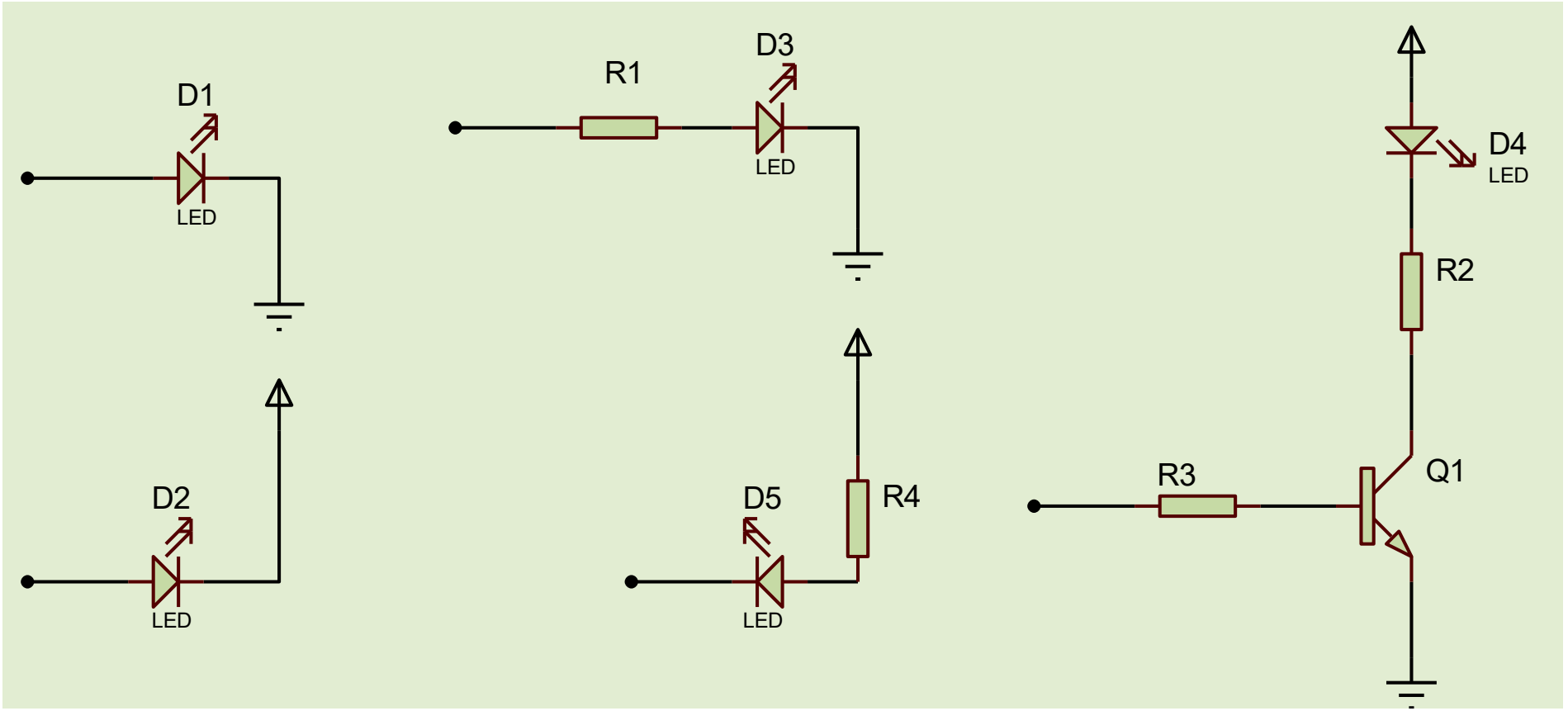
Düğme t anında kapanırsa

- Kontakt gürültüsünü gidermek için yazılımsal veya donanımsal çözümler uygulanmalıdır
- Yazılımsal olarak kontakt gürültüsü giderme : ilk değişim yakalandıktan sonra belirli süre aktif bekleme yapılıp uç tekrar kontrol edilir

Hardware Debounce Switch

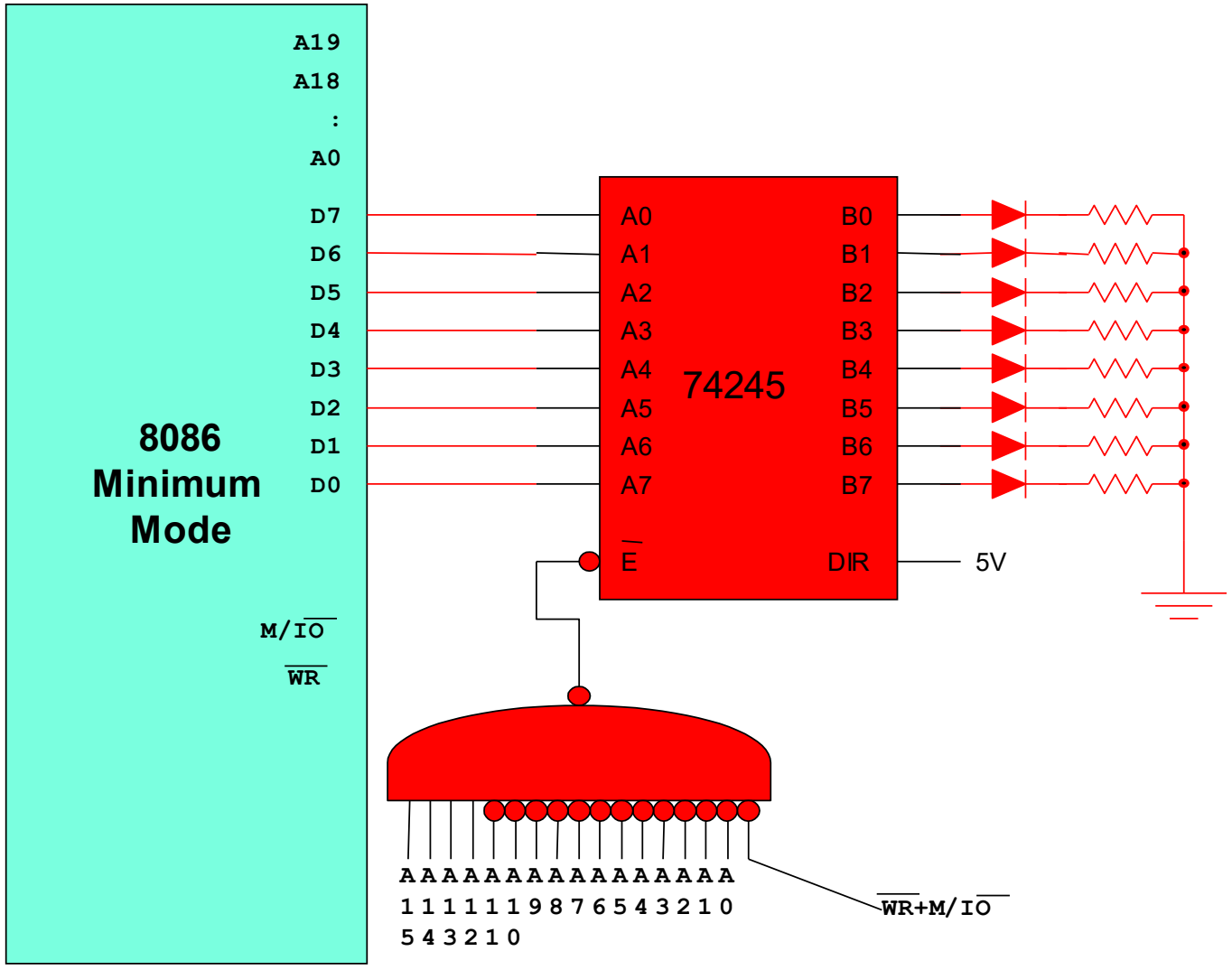


LED Arayüzü (Basit Arayüz Devreleri)

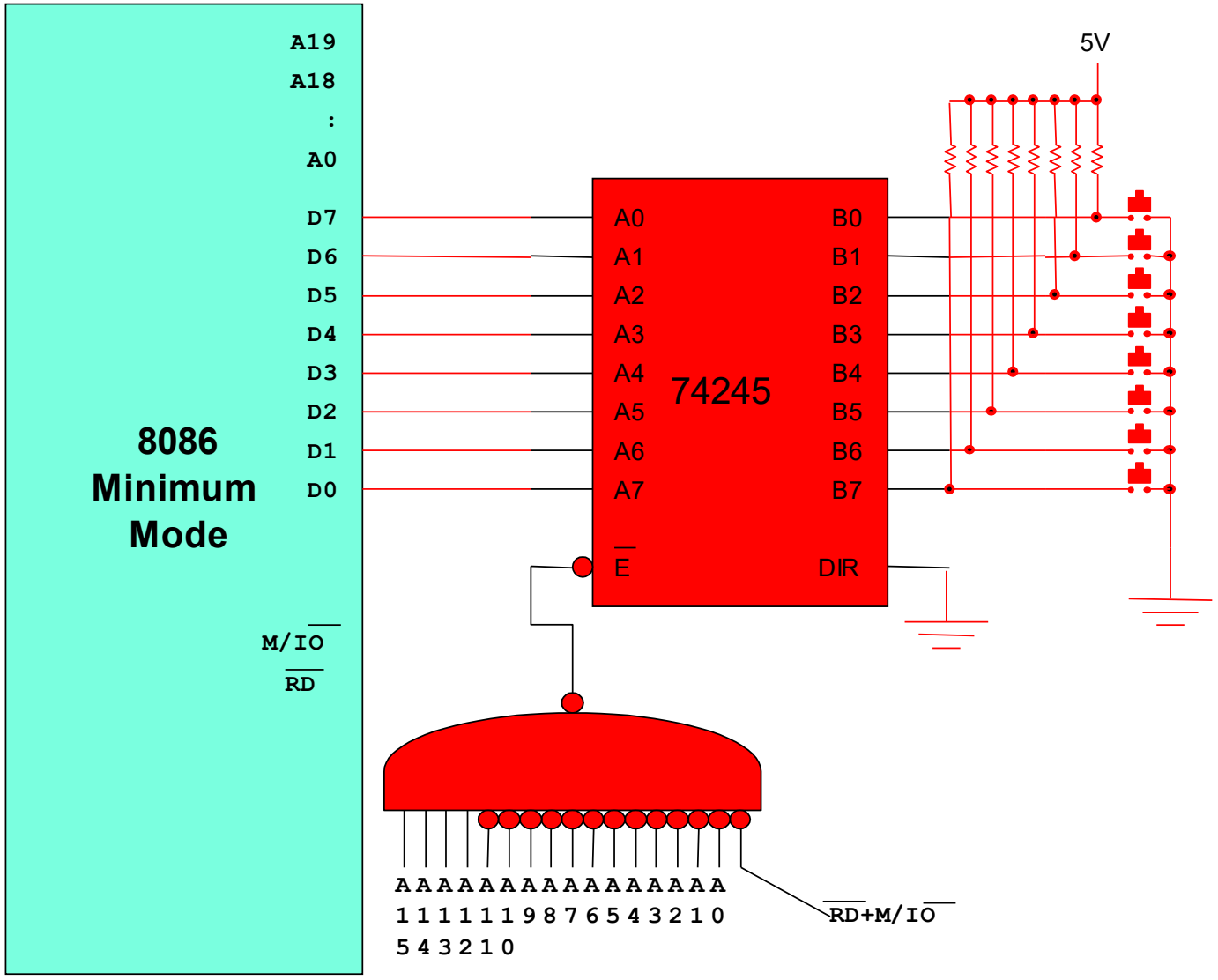


Basit Çıkış Birimi

- 8 LED kullanarak 0F000H adresine yerleştirilmiş basit bir çıkış biriminin tasarlanması ➔ donanım + adres çözümleme
- LED'lerde (on,off,on,off…) şeklinde patern oluşturma ➔ I/O programlama




```
      :
mov al, 55h
mov dx, 0F000h
out dx, al
      :
```



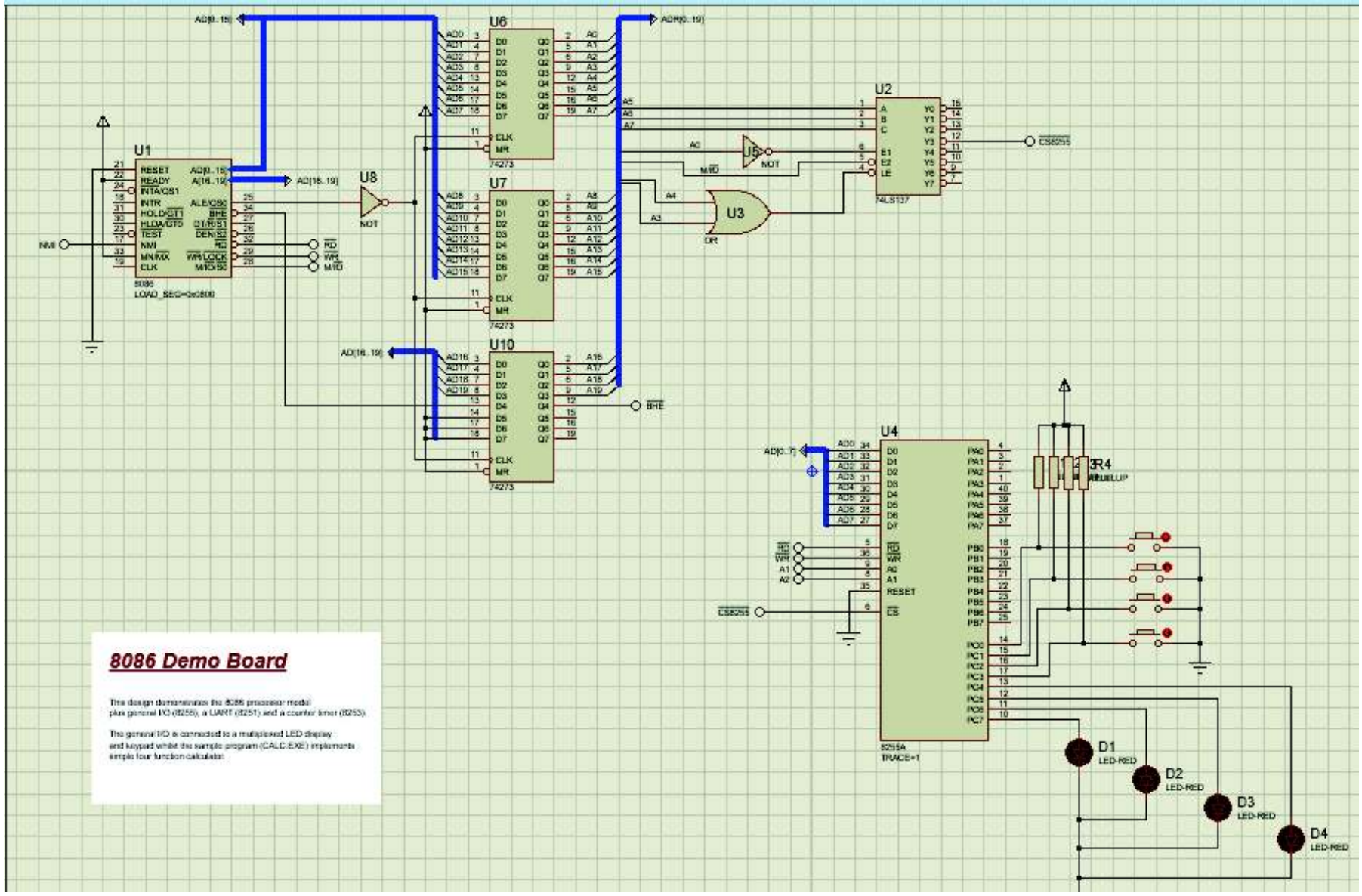

```
        :
mov dx, 0F000h
in  al, dx
        :
```

```
        mov dx,0F000h
L1:     in  al, dx
        cmp al, 0FFh
        je  L1
            :
```

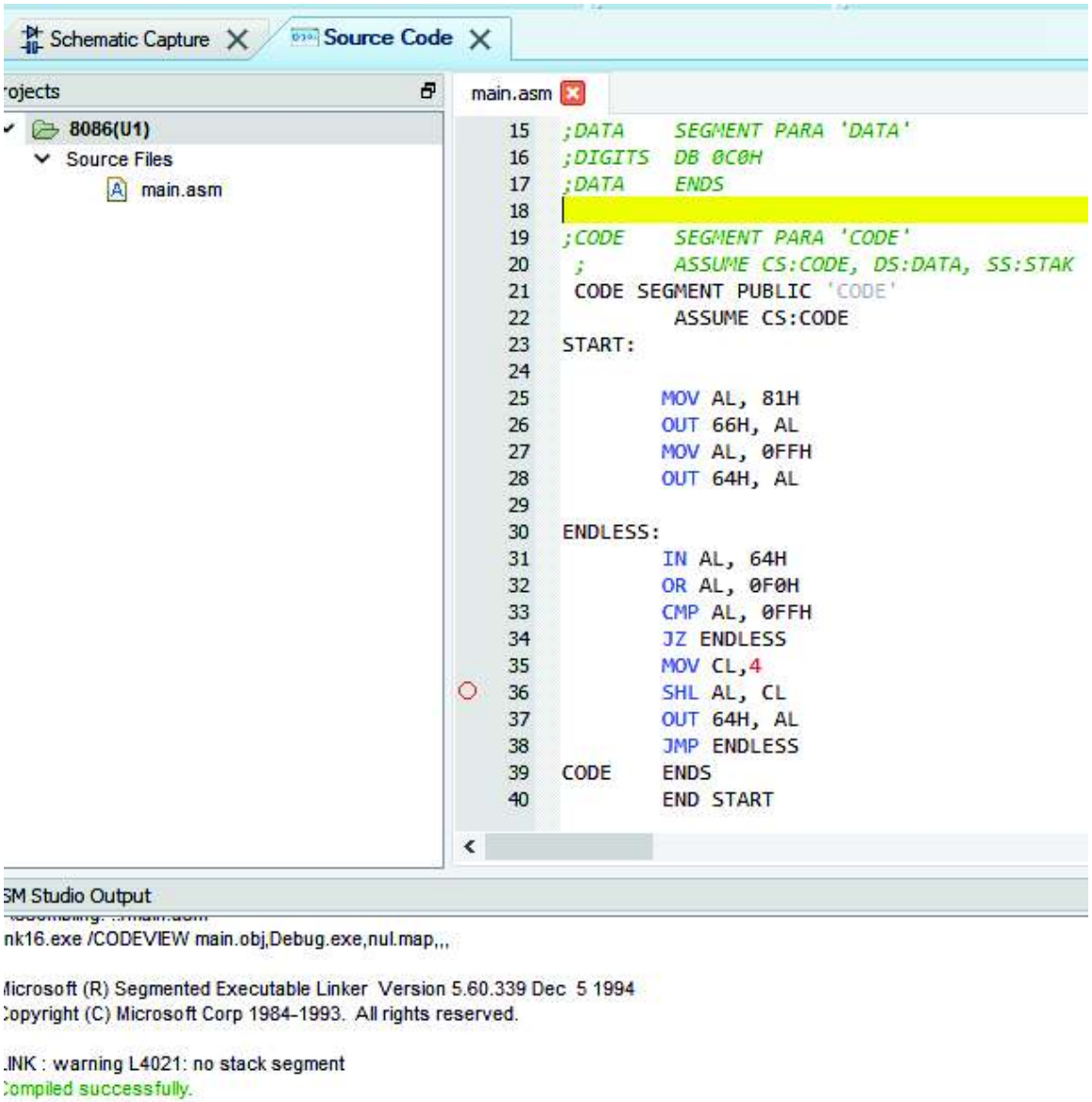
Basit Giriş/Çıkış Birimi

- Aynı adreste giriş ve çıkış birimi yerleştirildi → bu durum problem oluşturur mu?
- Basit giriş birimi / çıkış birimi 0F001H adresine yerleştirmek için ne yapılmalıdır?
- 0F000H adresinden itibaren 16 bitlik bir basit çıkış birimi nasıl tasarlanmalıdır?

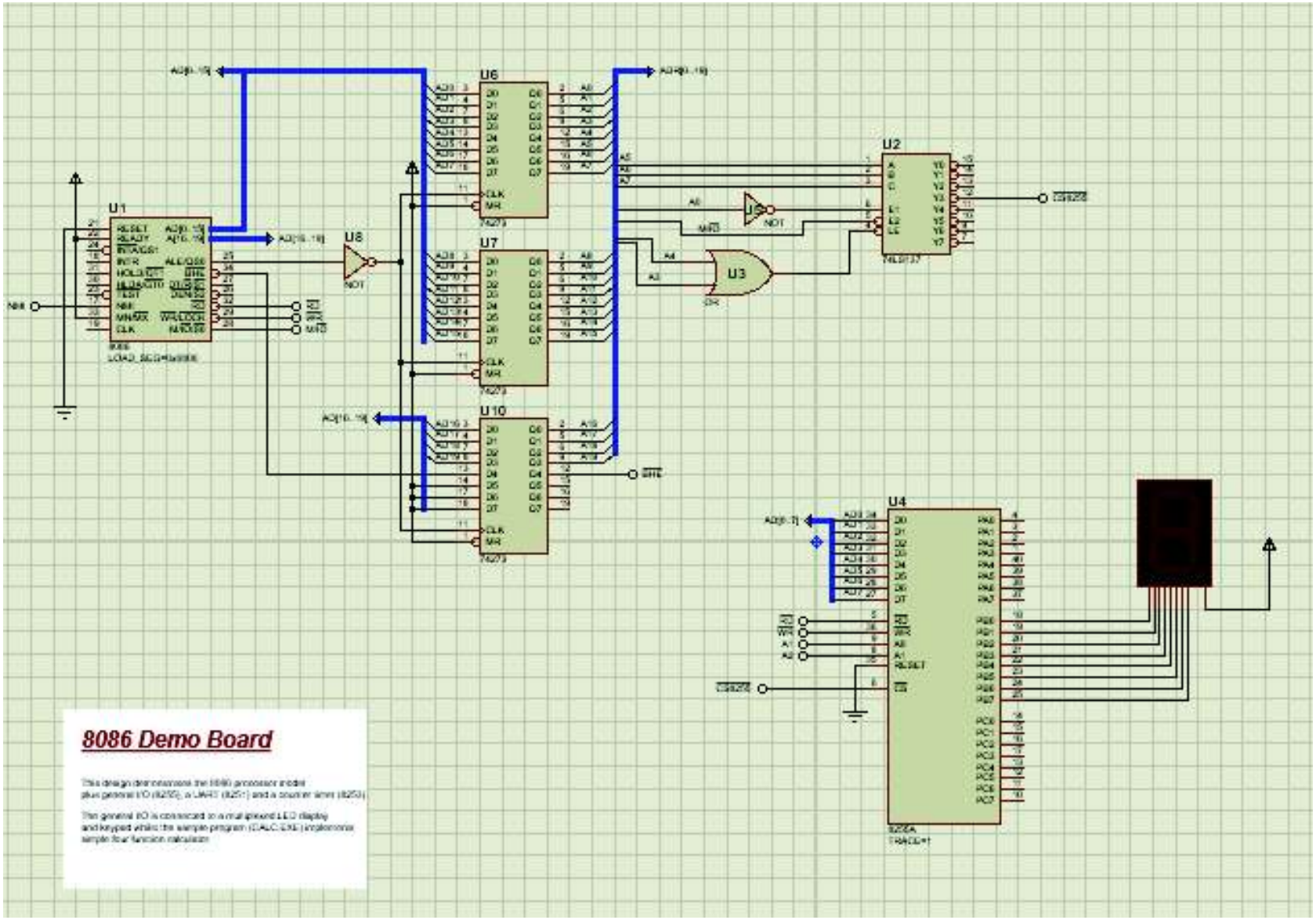
Örnek-1: 4 Buton ve 4 LED sürücü devresi



Örnek-1: 4 Buton ve 4 LED sürücü devresi



Örnek-1: 7 Parçalı Gösterge Devresi



Örnek-1: 7 Parçalı Gösterge Devresi

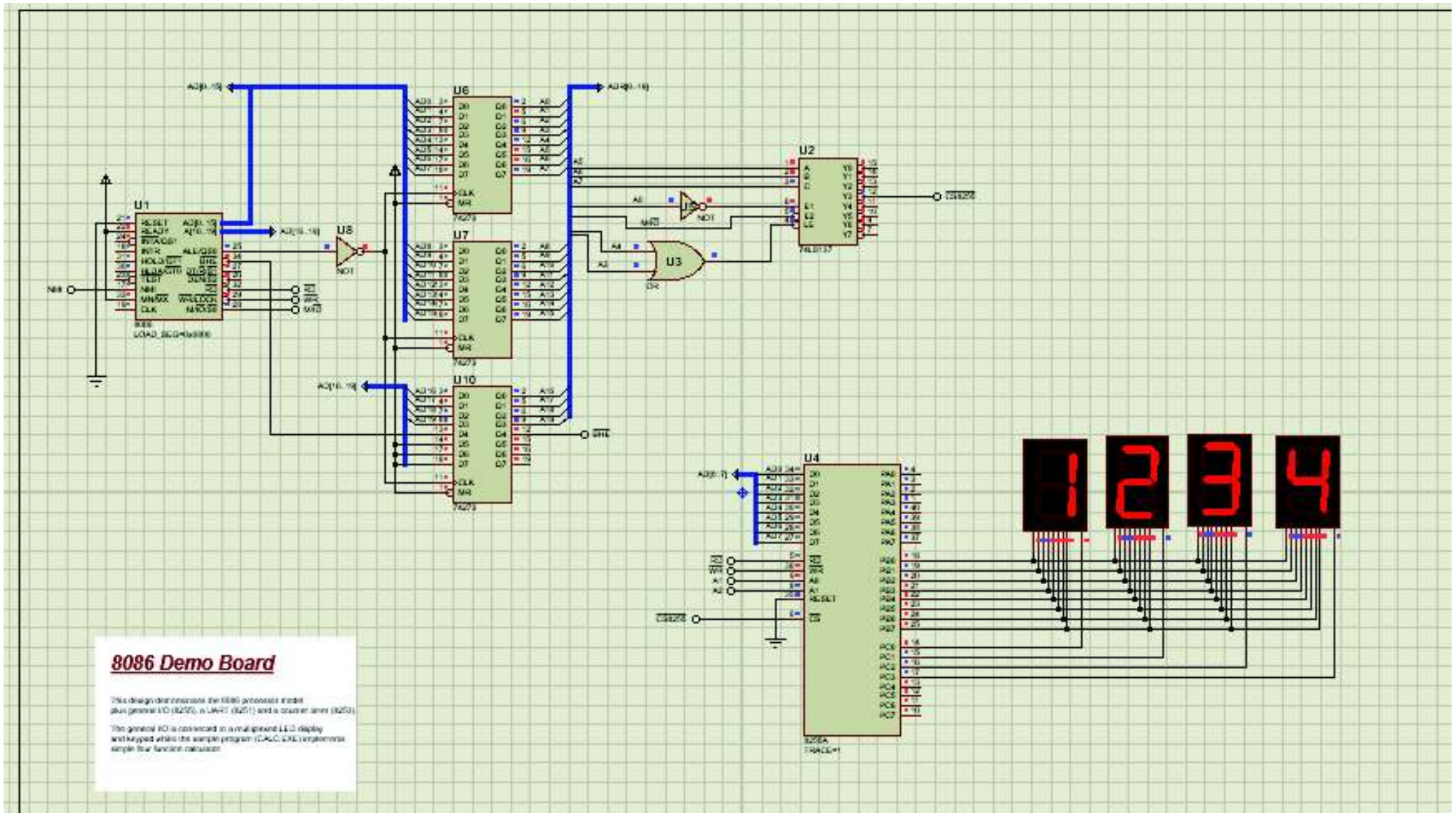
```
STAK    SEGMENT PARA STACK 'STACK'
        DW 20 DUP(?)
STAK    ENDS

DATA    SEGMENT PARA 'DATA'
DIGITS  DB 0C0H, 0F9H, 0A4H
DATA    ENDS

CODE    SEGMENT PARA 'CODE'
        ASSUME CS:CODE, DS:DATA, SS:STAK
START:
        MOV AX, DATA
                MOV DS, AX
        MOV AL, 80H
                OUT 66H, AL
                MOV AL, 0FFH
                OUT 62H, AL
                ;MOV AL, 0FFH
                ;OUT 64H, AL
                MOV AL, DIGITS[2]
                OUT 62H, AL
        ; Write your code here
ENDLESS:

        JMP ENDLESS
CODE    ENDS
        END START
```

Örnek-1: 4 adet 7 Parçalı Gösterge Devresi



Örnek-1: 4 adet 7 Parçalı Gösterge Devresi

```
STAK    SEGMENT PARA STACK 'STACK'
        DW 20 DUP(?)
STAK    ENDS

DATA    SEGMENT PARA 'DATA'
DIGITS  DB 0C0H, 0F9H, 0A4H, 0B0H, 99H, 92H, 82H, 0F8H, 80H, 98H
DATA    ENDS

CODE    SEGMENT PARA 'CODE'
        ASSUME CS:CODE, DS:DATA, SS:STAK

DELAY PROC NEAR
                MOV CX, 0FFH
  L1:
                LOOP L1
                RET
                DELAY ENDP

START:
                 MOV AX, DATA
                MOV DS, AX
                MOV AL, 80H
                OUT 66H, AL
                MOV AL, 0FFH
                OUT 62H, AL
                MOV AL, 0FFH
                OUT 64H, AL
                MOV AL, DIGITS[5]
                OUT 62H, AL

ENDLESS:
                MOV AL, 0F1H
                OUT 64H, AL
                MOV AL, DIGITS[1]
                OUT 62H, AL

                CALL DELAY
                MOV AL, 0F2H
                OUT 64H, AL
                MOV AL, DIGITS[2]
                OUT 62H, AL

                CALL DELAY

                MOV AL, 0F4H
                OUT 64H, AL
                MOV AL, DIGITS[3]
                OUT 62H, AL

                CALL DELAY

                MOV AL, 0F8H
                OUT 64H, AL
                MOV AL, DIGITS[4]
                OUT 62H, AL
                CALL DELAY

        JMP ENDLESS

CODE    ENDS
        END START
```

BSR Mod Örneği

- Örnek:
- 80H adresinden itibaren ardışık çift adreslere yerleştirilmiş bir 8255'de
  - PC2'yi lojik 1 olacak şekilde
  - PC6'da ise duty cycle'ı %66 olan bir kare dalga üretecek şekilde

programlayın

BSR Mod Örneği

```
MOV AL, 00000101B
OUT 86H, AL
AGAIN   MOV AL, 0xxx1101
        OUT 86H, AL
        CALL Delay
        CALL Delay
        MOV AL, 0xxx1100
        OUT 86H, AL
        CALL Delay
        JMP AGAIN
```