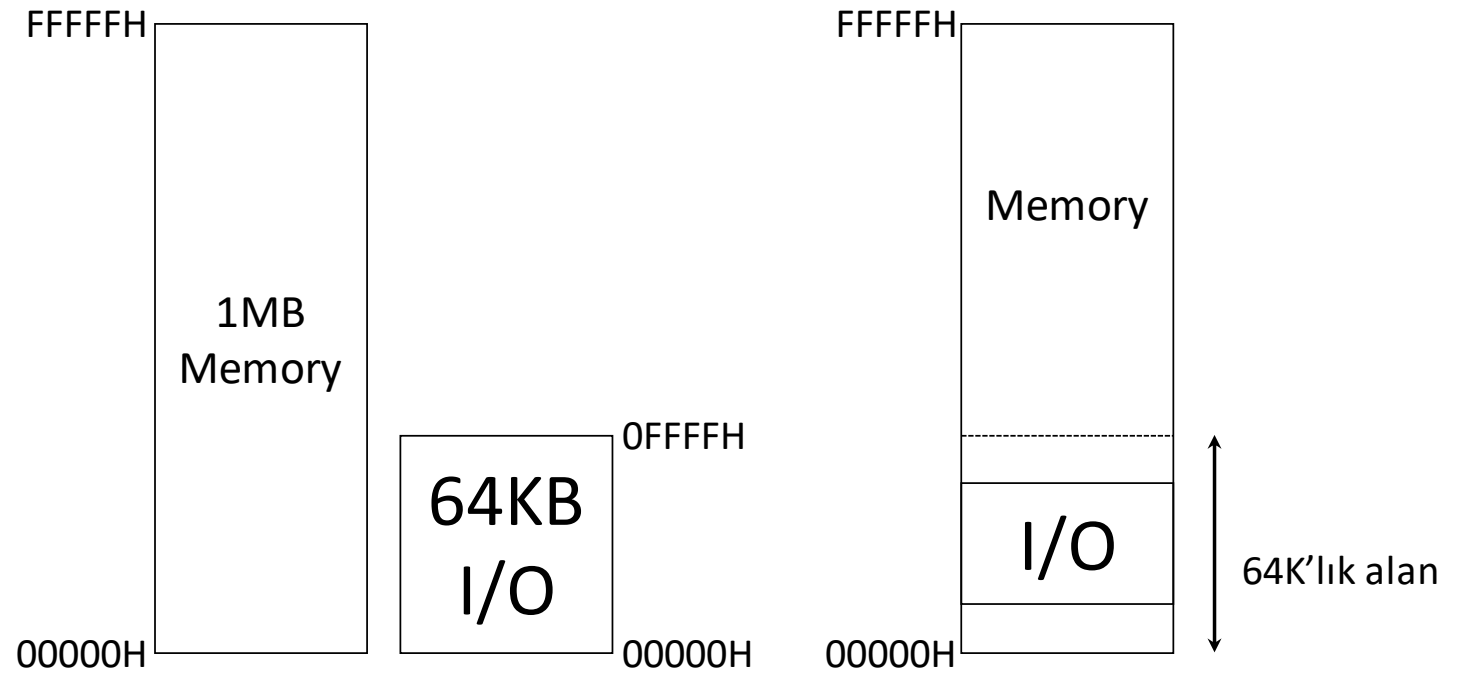


G/Ç Haritalama Yöntemleri

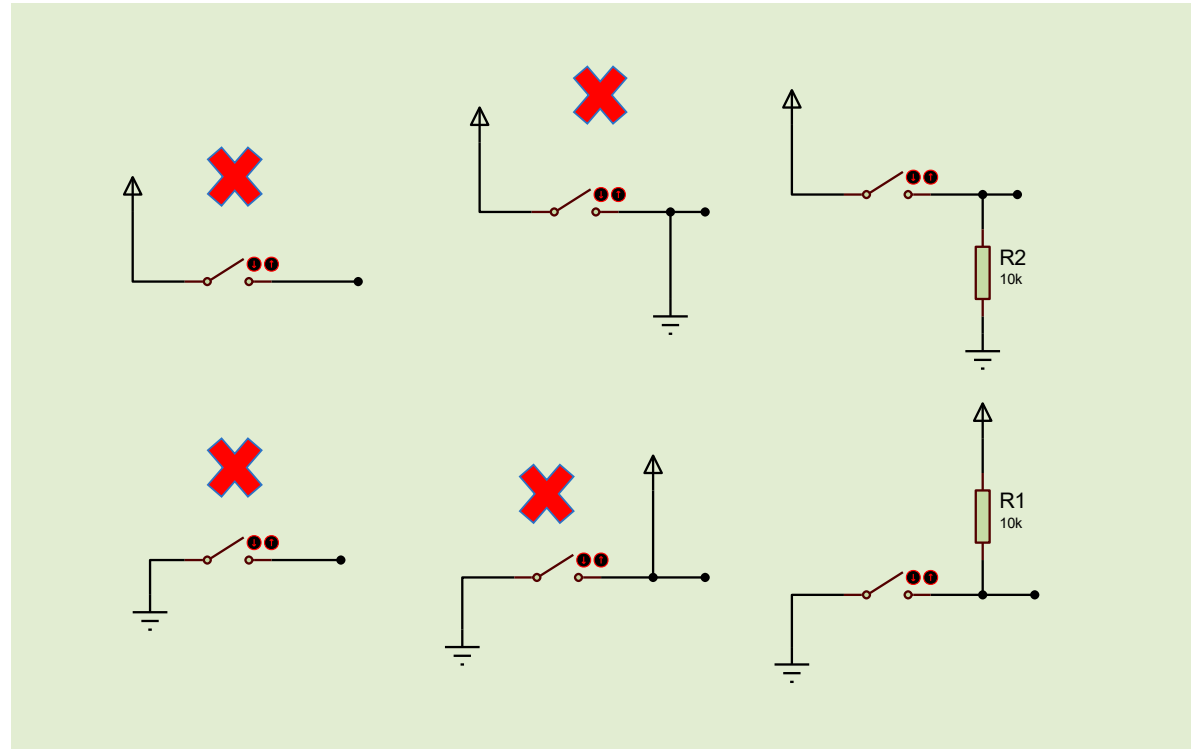
- Isolated I/O Mapping → M/\overline{IO} ucu adres çözümleme için kullanılır
- Memory Mapped I/O → M/\overline{IO} ucu adres çözümleme için kullanılmaz

Isolated I/O – Memory Mapped I/O



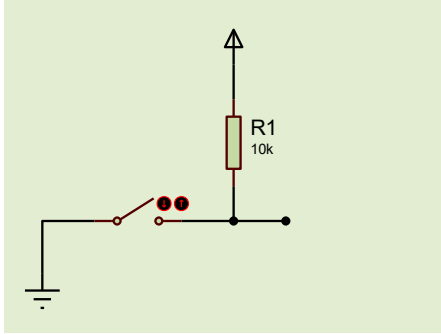
Düğme Arayüzü (Basit Arayüz Devreleri)

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



Düğme Arayüzü – Kontakt Gürültüsü

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

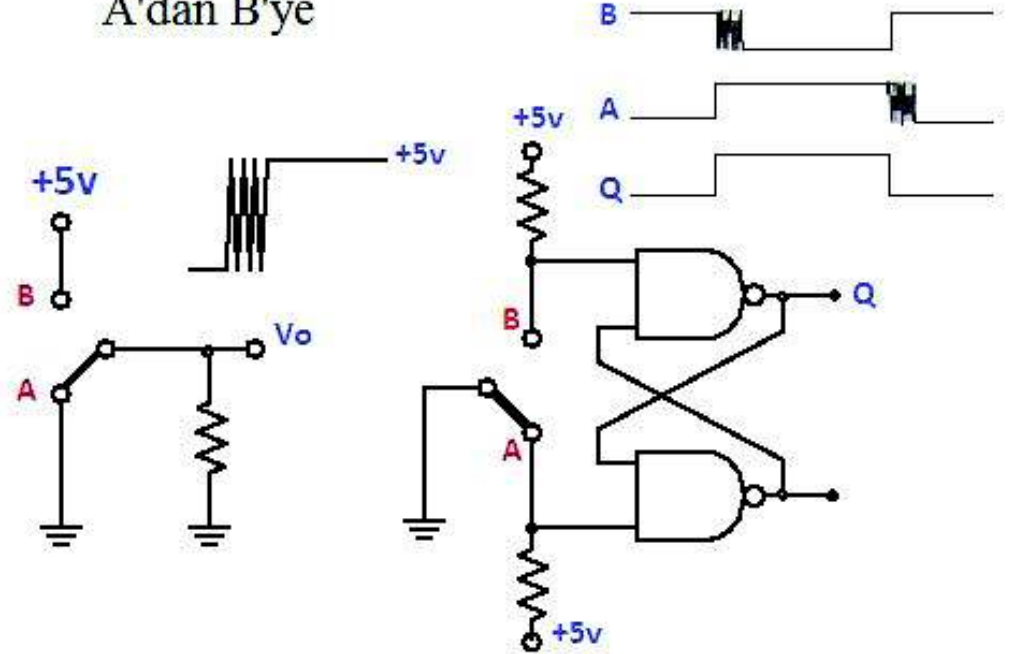


Düğme t anında kapanırsa

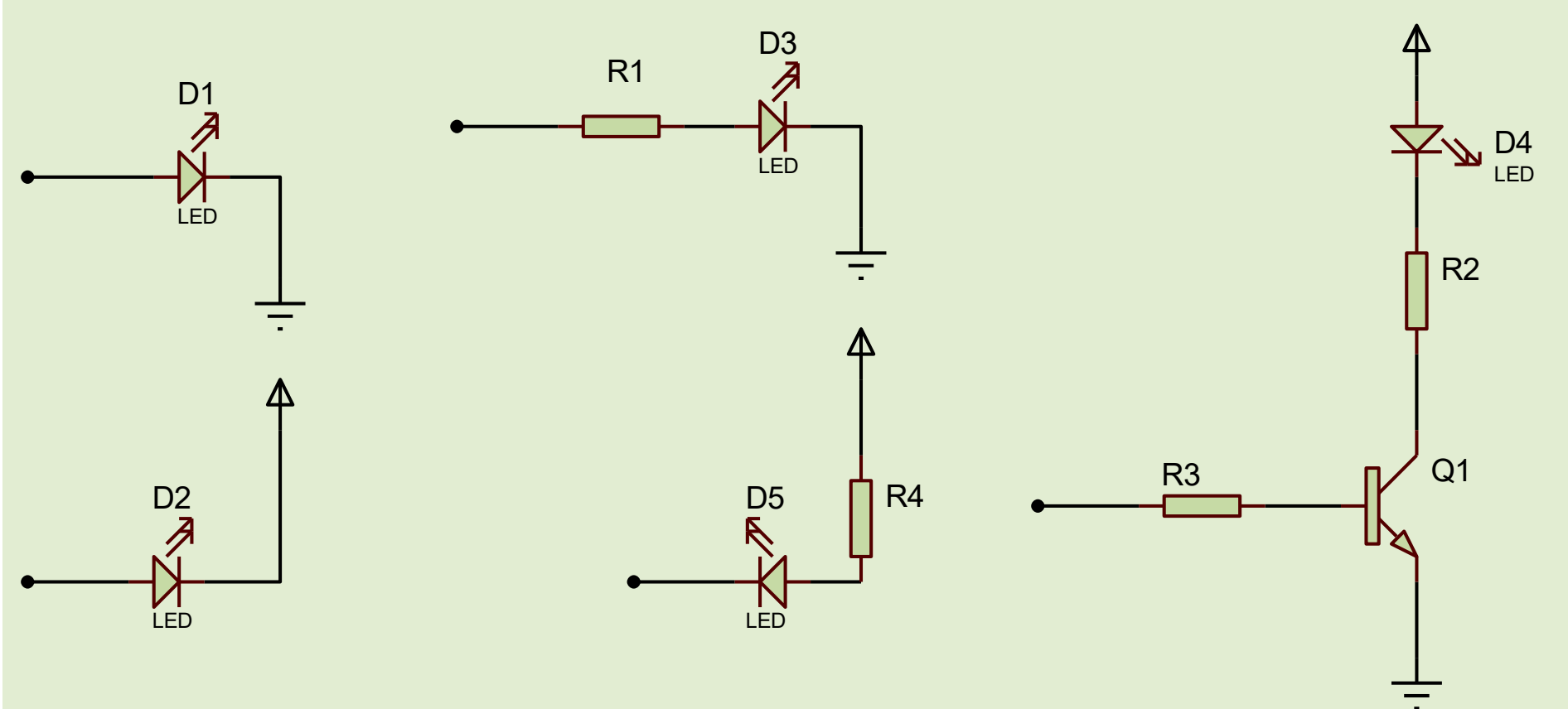


Hardware Debounce Switch

A'dan B'ye

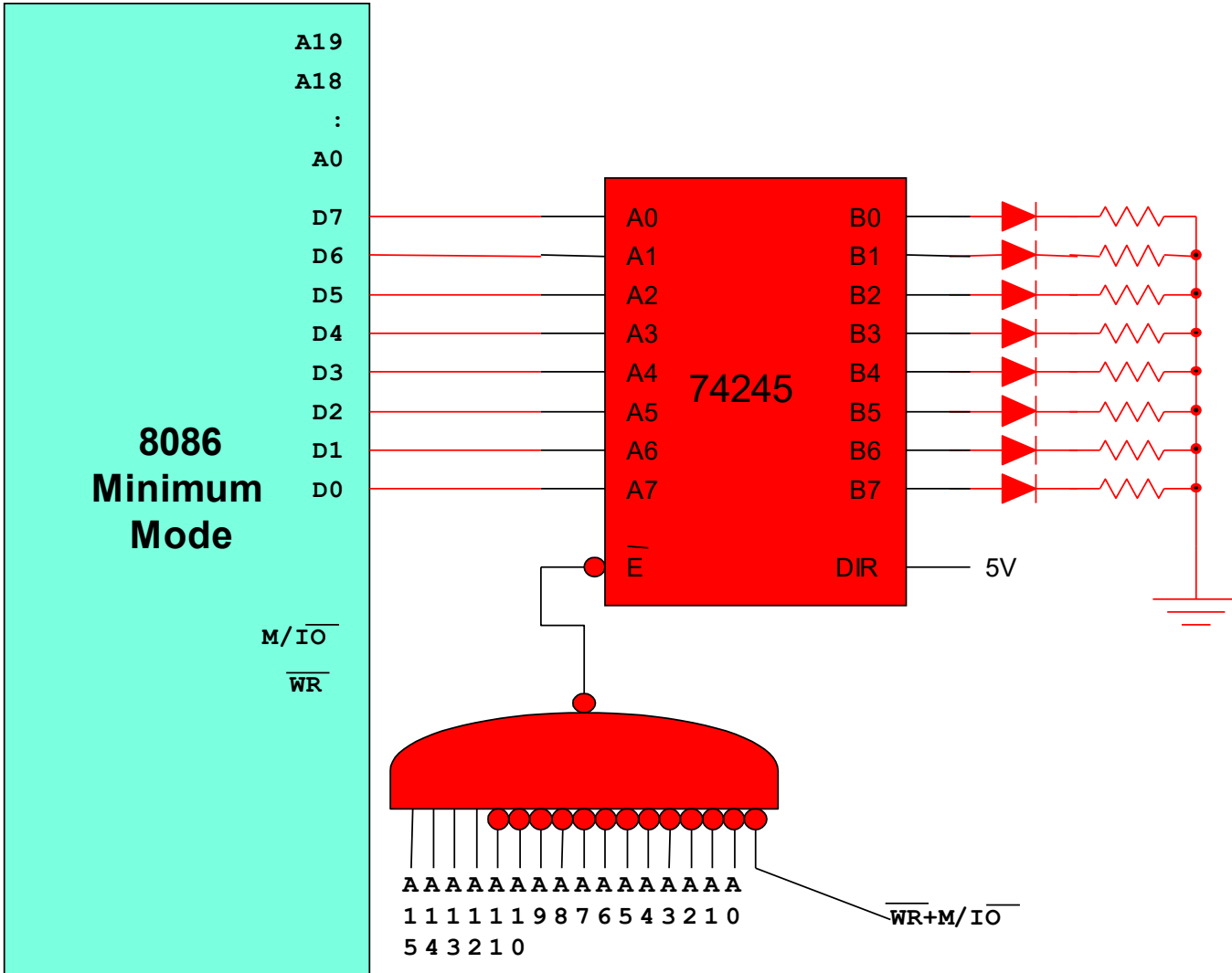


LED Arayüzü (Basit Arayüz Devreleri)



Basit Çıkış Birimi

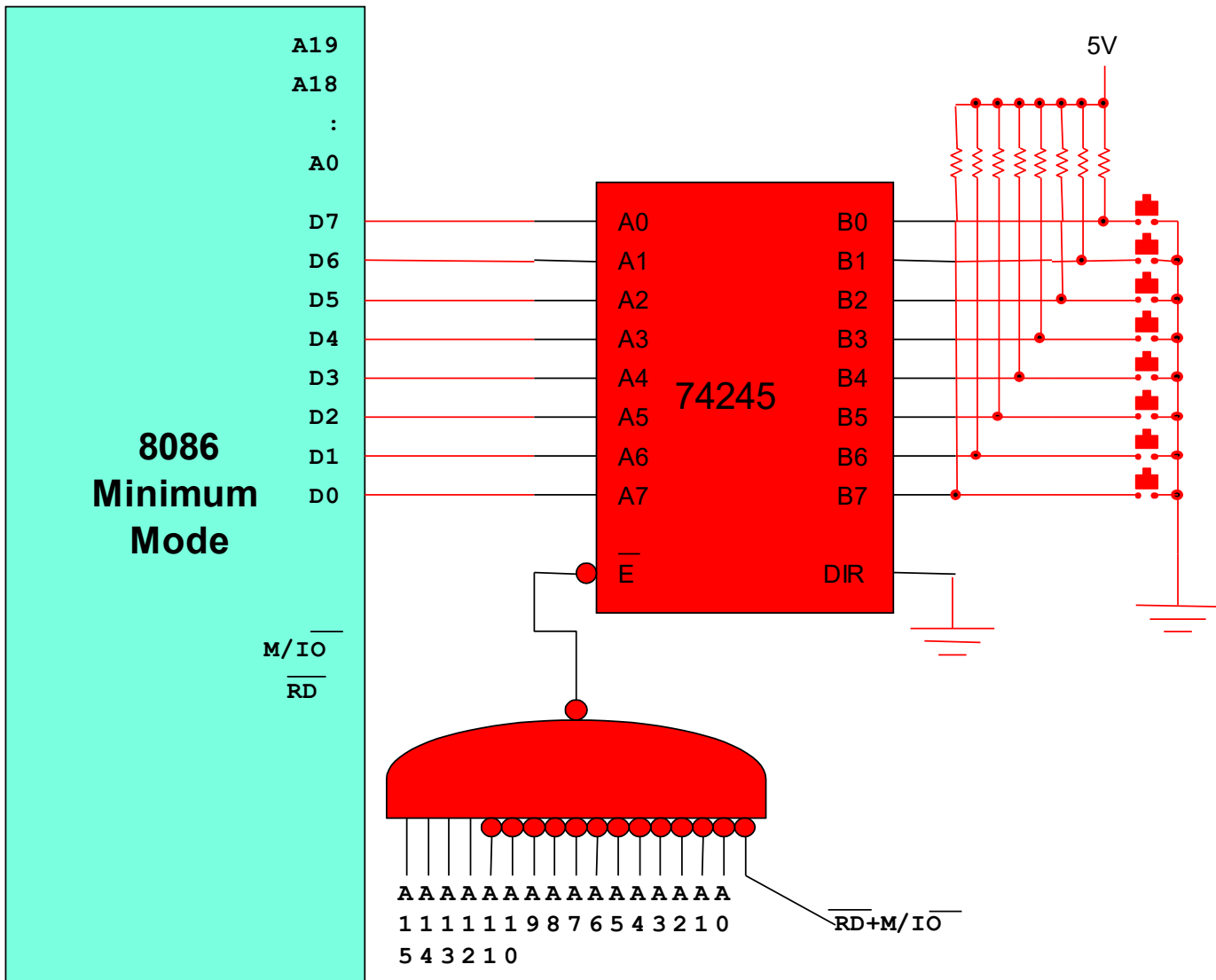
- 8 LED kullanarak 0F000H adresine yerleştirilmiş basit bir çıkış biriminin tasarlanması → donanım + adres çözümleme
- LED'lerde (on,off,on,off...) şeklinde patern oluşturma → I/O programlama



```

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

```



```

:
mov dx, 0F000h
in  al, dx
:

```

```

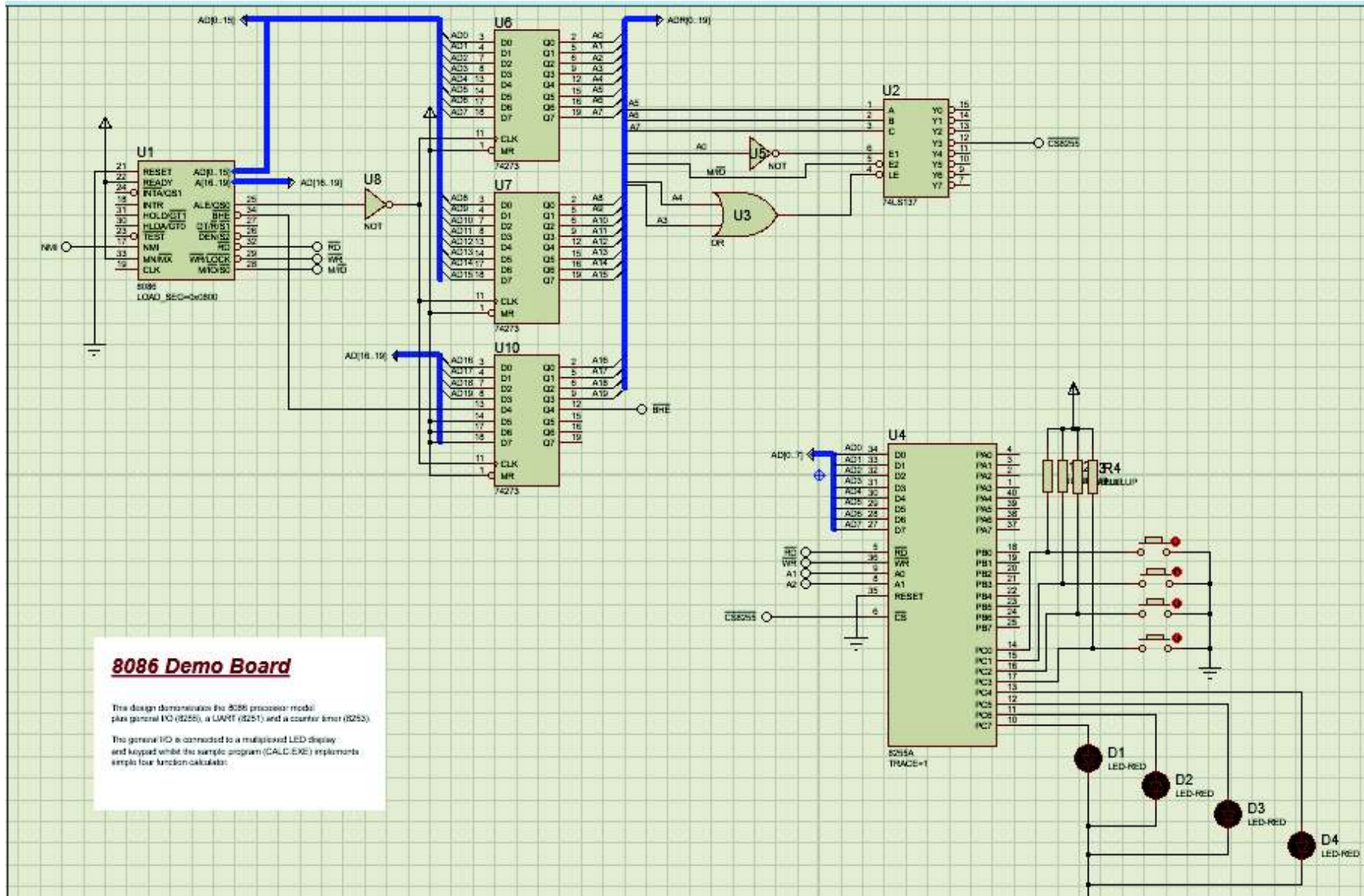
mov dx, 0F000h
L1: in  al, dx
    cmp al, 0FFh
    je  L1
:

```

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

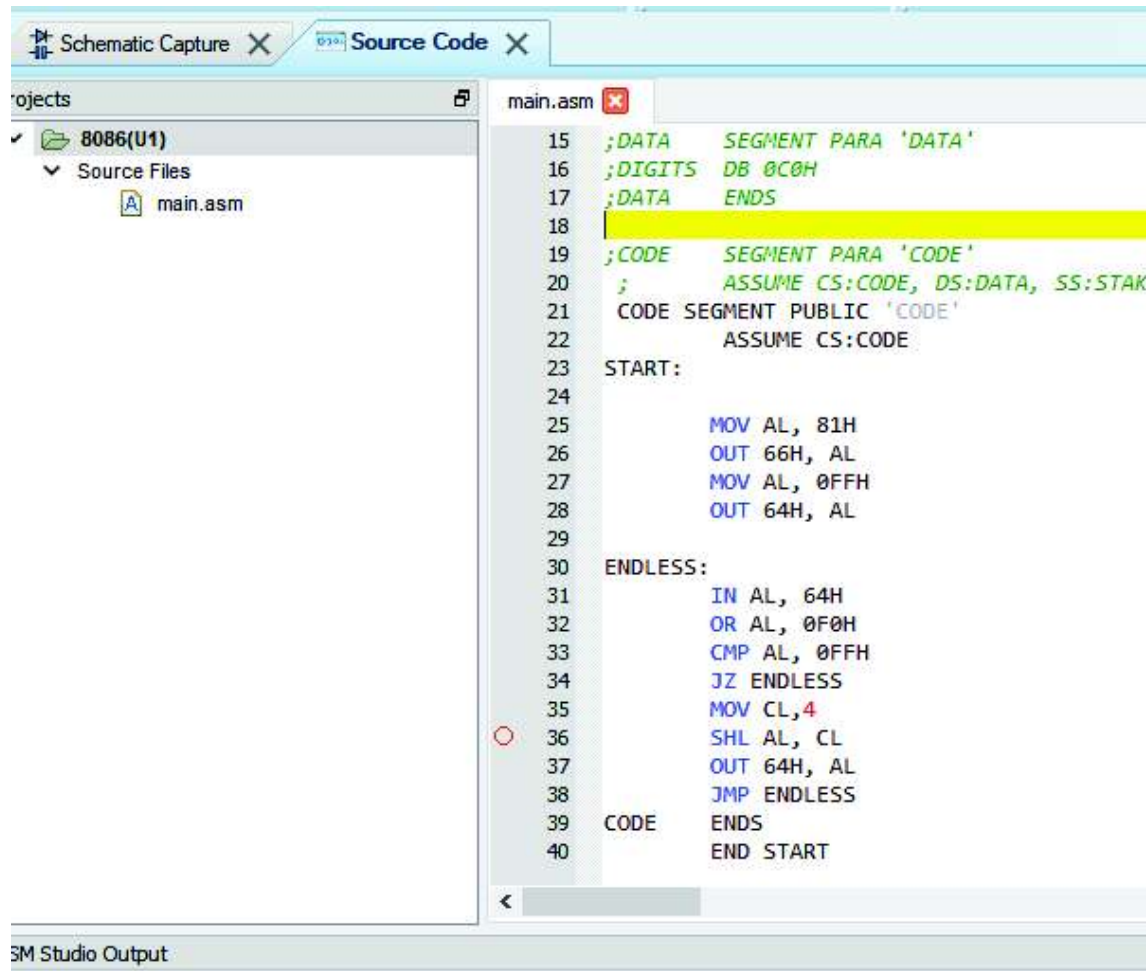


8086 Demo Board

This design demonstrates the 8086 processor model plus general I/O (8255), a UART (8251) and a counter timer (8253).

The general I/O is connected to a multiplexed LED display and keypad whilst the sample program (CALC.EXE) implements simple four function calculator.

Örnek-1: 4 Buton ve 4 LED sürücü devresi



The screenshot shows the MASM Studio Source Code window for a project named '8086(U1)'. The file 'main.asm' is open, displaying the following assembly code:

```
15 ;DATA SEGMENT PARA 'DATA'
16 ;DIGITS DB 0C0H
17 ;DATA ENDS
18
19 ;CODE SEGMENT PARA 'CODE'
20 ; ASSUME CS:CODE, DS:DATA, SS:STAK
21 CODE SEGMENT PUBLIC 'CODE'
22 ASSUME CS:CODE
23 START:
24
25     MOV AL, 81H
26     OUT 66H, AL
27     MOV AL, 0FFH
28     OUT 64H, AL
29
30     ENDLESS:
31         IN AL, 64H
32         OR AL, 0F0H
33         CMP AL, 0FFH
34         JZ ENDLESS
35         MOV CL, 4
36         SHL AL, CL
37         OUT 64H, AL
38         JMP ENDLESS
39     CODE ENDS
40     END START
```

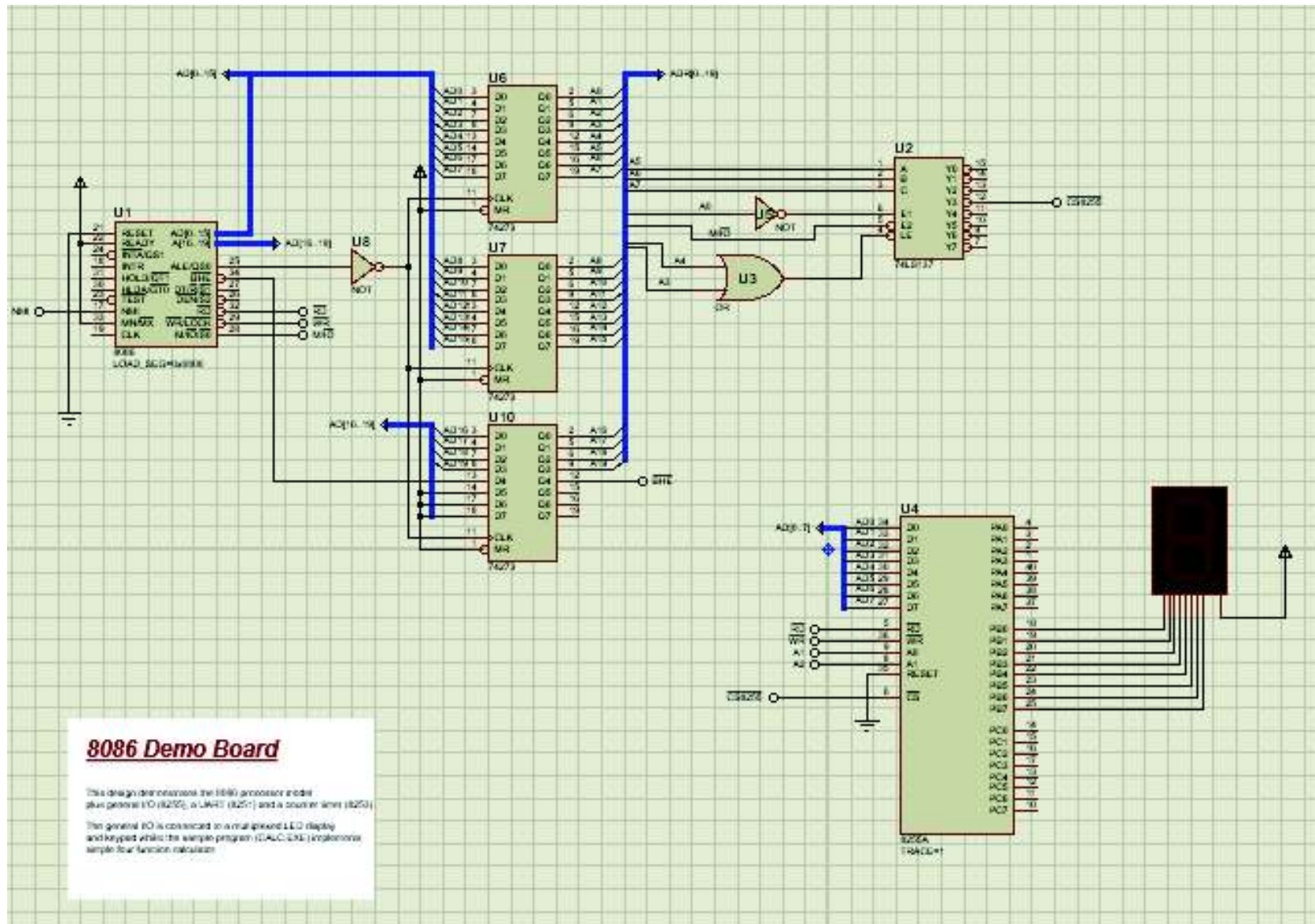
ASM Studio Output

Compiling: main.asm
nk16.exe /CODEVIEW main.obj,Debug.exe,nul.map,...

Microsoft (R) Segmented Executable Linker Version 5.60.339 Dec 5 1994
Copyright (C) Microsoft Corp 1984-1993. All rights reserved.

.LNK : warning L4021: no stack segment
Compiled successfully.

Örnek-1: 7 Parçalı Gösterge Devresi



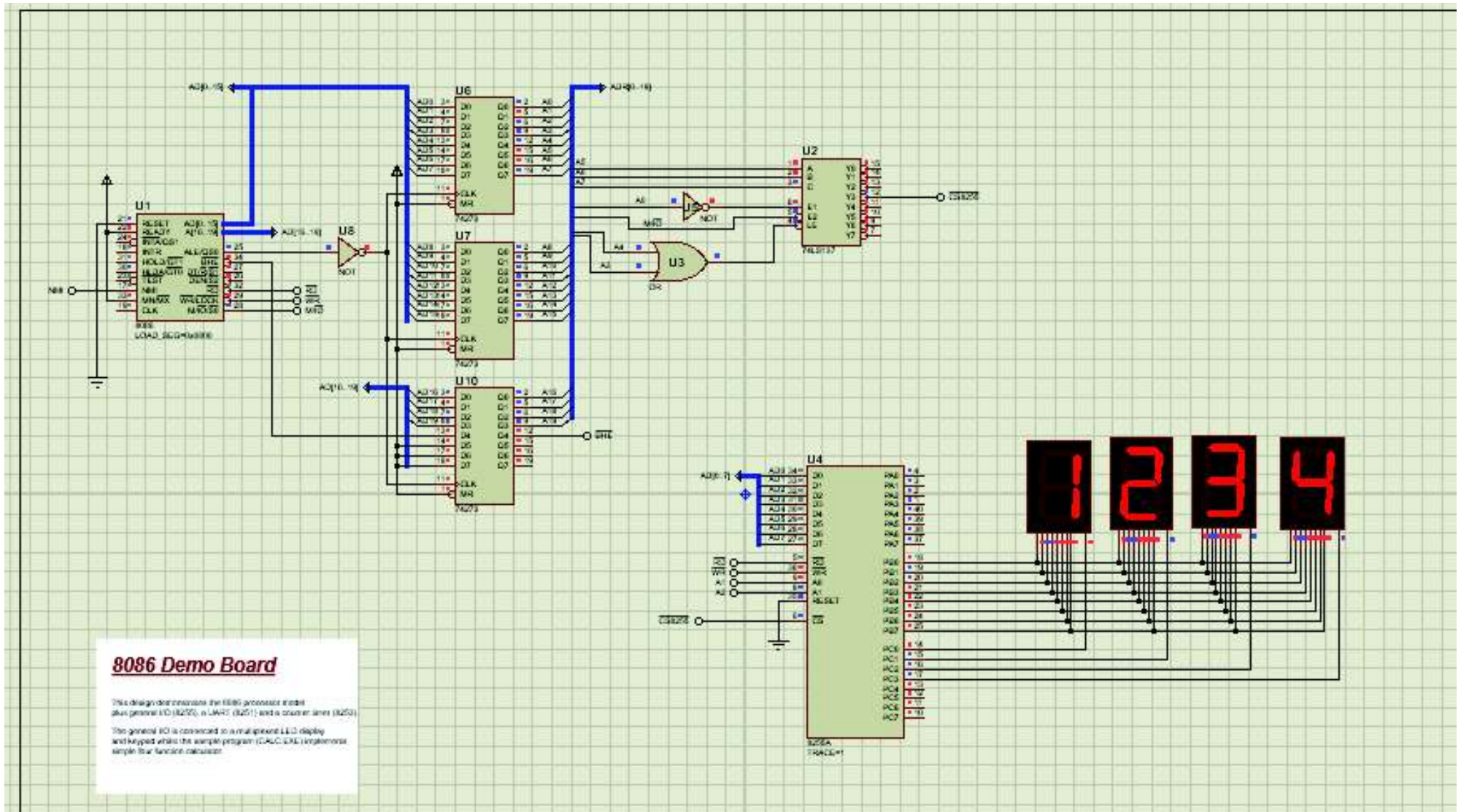
Örnek-1: 7 Parçalı Gösterge Devresi

```
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
```