

BİLGİSAYAR KONTROLLÜ BASKI DEVRE ÇİZİCİ

- MiniCiR -

Cevher AK
Ramazan YAĞMUR
Muhammed TEKPINAR

Mersin Üniversitesi
Mühendislik Fakültesi
Elektrik-Elektronik Mühendisliği

LİSANS TEZİ

Tez Danışmanı
Öğr. Gör. Dr. Ali YILDIZ

MERSİN
2006

TEŐEKKÜR

Projenin hazırlanmasında bize destek veren deęerli hocamız Ali YILDIZ'a, alıőmamızın mekanik aksamını saęlamamızda yardımcı olan Mustafa TEKPINAR'a , alıőmamıza fikirleriyle katkıda bulunan Hasan SELEK'e ve manevi desteklerinden ötürü Seda ERMİŐ, Murat PEKEL ve Zafer KİBAR'a sonsuz teőekkürlerimizi sunarız. Ayrıca bu alıőmamızı bizleri bu günlere getiren deęerli annelerimize ithaf ediyoruz.

İÇİNDEKİLER

| | |
|---|----|
| İÇİNDEKİLER..... | i |
| 1. GİRİŞ..... | 1 |
| 1.1 Projenin Amacı..... | 2 |
| 2. KULLANILAN ELEMANLAR..... | 2 |
| 2.1. Step Motor..... | 2 |
| 2.1.1. Step Motorların Kullanım Alanları..... | 2 |
| 2.1.2. Step Motorların Avantajları..... | 3 |
| 2.1.3. Step Motorların Dezavantajları..... | 4 |
| 2.1.4. Step Motor Çeşitleri..... | 4 |
| 2.1.4.1. Sabit Mıknatıslı Step Motorlar..... | 5 |
| 2.1.4.2. Değişken Relüktanslı Step Motorlar..... | 6 |
| 2.1.5. Step Motor Çalışma Prensipleri..... | 7 |
| 2.1.6. Step Motorlarda Uçların Bulunması..... | 8 |
| 2.2. RÖLE..... | 10 |
| 2.3. TRANSİSTÖR..... | 10 |
| 2.4. PIC (Peripheral Interface Controller)..... | 12 |
| 2.4.1. Mikrodenetleyiciler..... | 12 |
| 2.4.1.1. Mikrodenetleyicilerin Kullanım Sebebi..... | 13 |
| 2.4.1.2. PIC Mikrodenetleyicisi Ve PIC16F628..... | 14 |
| 2.4.1.3. PIC 16F628 Donanım Özellikleri..... | 16 |

| | |
|---|----|
| 2.4.1.4. Bellek Düzeni..... | 19 |
| 2.4.1.5 PIC Assembly Ve Komut Kümesi..... | 21 |
| 2.5. ULN2003 Entegresi..... | 23 |
| 2.5.1 Unipolar Step Motor Sürücü Devresi..... | 25 |
| 2.6. Paralel Port..... | 25 |
| 2.6.1. Standart Paralel Port Yapısı..... | 25 |
| 2.6.2. Paralel Portun Sistemdeki Kullanımı..... | 27 |
| 2.6.3 Pascal'da Paralel Port Kullanımı..... | 28 |
| 3. MiniCiR..... | 30 |
| 3.1. MiniCiR Çalışma Prensipleri..... | 30 |
| 3.2. MiniCiR PIC Programı..... | 36 |
| 3.3. MiniCiR Bilgisayar Programı..... | 44 |
| Kaynaklar..... | 80 |

1. GİRİŞ

İlk defa 1936 yılında Avusturya'lı mühendis Paul Eisler tarafından kullanılmış olan baskı devre yöntemi, ilk zamanlarda Amerika'da radyo ve benzeri basit devrelerin sağlam bir biçimde yapılması amacıyla geliştirilmiş olup, günümüzde bilgisayar ve mobil telefonlar gibi kısıtlı alanlarda çok fazla elektronik elemanın kullanıldığı gelişmiş cihazlarda çok katlı baskı devre teknolojisine ulaşmıştır.

Elektronik devre şemaları, baskı devre şemalarına dönüştürülecek bakır plakaya aktarılır. Bu işleme baskı devre çıkarma tekniği denir. Devrelerin boyutlarının küçülmesini ve montajının kolaylaşmasını sağlar, elemanlar sabit ve düzgün yerleştirildiğinden dolayı kayıplar azalmakta ve devre daha kararlı çalışmaktadır, ayrıca baskı devre yönteminin kullanılması hem devrede yüksek frekanslarda çalışmayı hem de düşük güç tüketimi sağlar.

Sık kullanılan 3 çeşit baskı devre çıkarma çıkartma tekniği vardır.

- 1.) Baskı devre kalemiyle çizim tekniği
- 2.) Pozitif 20 tekniği
- 3.) İpek baskı tekniği

BASKI DEVRE KALEMİYLE ÇİZİM TEKNİĞİ

Baskı devre kalemi ile baskı devre yapılacağı zaman aşağıdaki malzemeler kullanılır.

- 1.) Bakır plaket
- 2.) Baskı devre kalemi
- 3.) Perhidrol
- 4.) Tuz ruhu
- 5.) Testere
- 6.) Yüksel devirli küçük matkap
- 7.) Temizlik malzemesi

1.1 PROJENİN AMACI

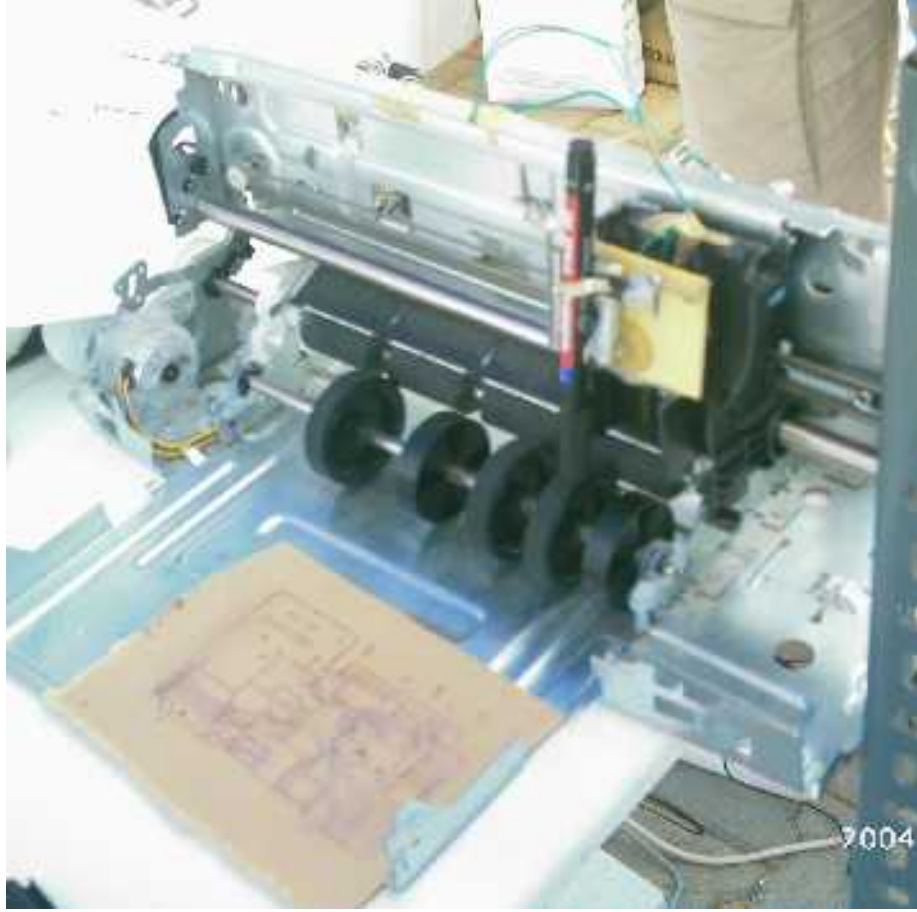
Küçük işletmelerde ve kişisel çalışmalarda baskı devre yapmak için prototip baskı devre üretim sistemlerine ihtiyaç duyulmaktadır. Özellikle kişisel kullanımlar için böyle bir cihaz bulunmamaktadır. Ayrıca piyasada baskı devrenin elde edilmesi aşaması zahmetli ve masraflı olmaktadır. Bu amaçla geniş bir kullanıcı kitlesine hitap edebilecek pratik, taşınabilir, seri ve ekonomik bir biçimde bilgisayar kontrollü baskı devre çizimi yapabilen bir çizici tasarlandı.

Bu çalışmada baskı devre şeklinin bilgisayar ortamında çizilip doğrudan plakete aktarılmasını mümkün kılacak bir sistem geliştirilmiştir.

2. KULLANILAN ELEMANLAR

Genel amaçlı, projeye özel bilgi içermediğinden çıkarılmıştır.

3. MiniCiR

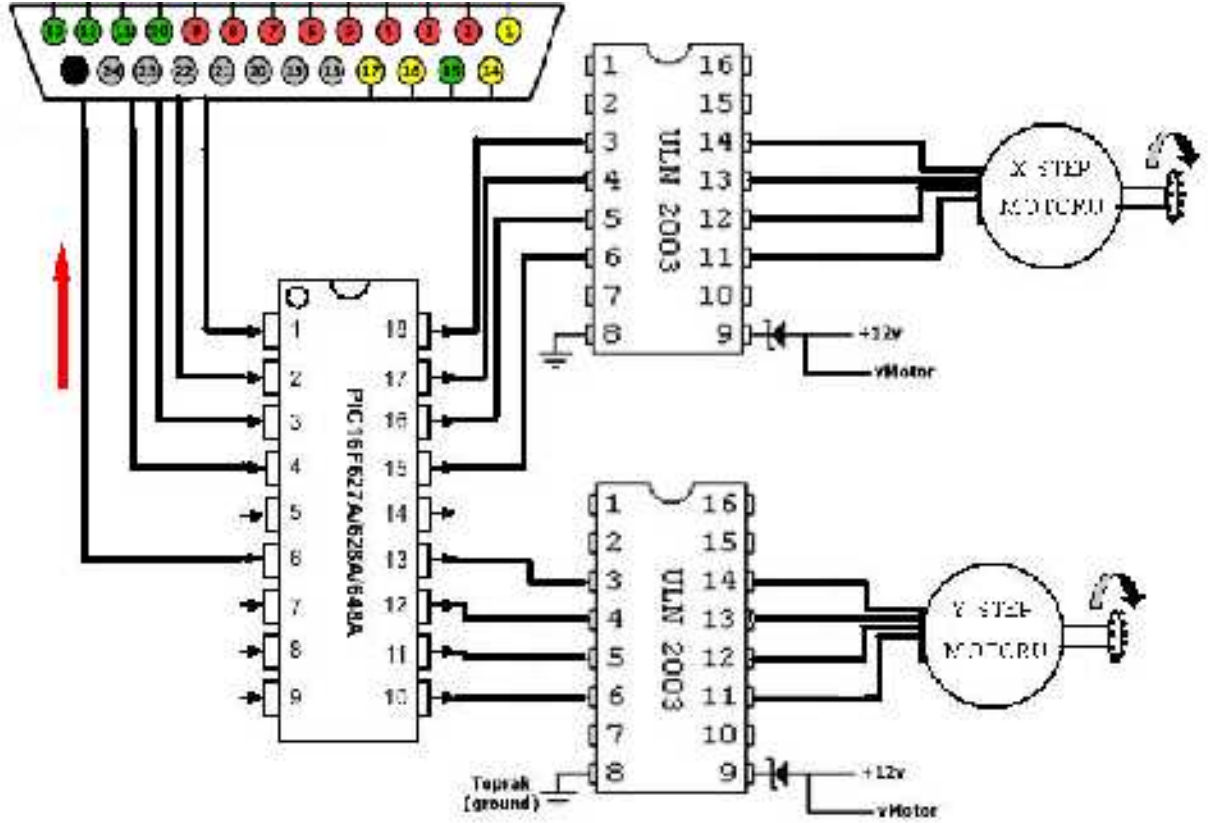


Fotoğraf - 1 - Sistemin Görüntüsü

3.1 MiniCiR ÇALIŞMA PRENSİPLERİ

Sistem için gereken tüm beslemeler; PIC için DC 3-5 volt, transistör için DC 6 volt ve ULN2003 için DC 12 volt uygulandıktan sonra yeterli büyüklükteki bir adet bakır levhannın makineye yerleştirilmesi gerekmektedir.

Elde etmek istediğimiz baskı devrenin şeklini makine için özel tasarlanmış olan programda çizip programdaki çiz butonuna bastıktan sonra bilgisayar, 1 tanesi Y-motorunu (ileri-geri) 1 tanesi X-motorunu (sağ-sol) sürmek için kullanılan iki adet ULN2003 entegresini kontrol amacıyla PIC'e veri göndermektedir.



Şekil -17- Sistemin Haberleşme Şeması

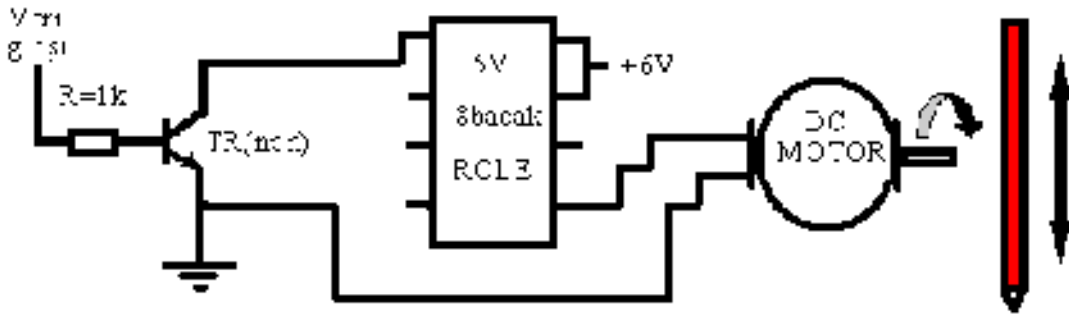
PIC'in bilgisayardan aldığı veriye göre step motorlardan biri, örneğin Y-motorunu çalıştırdığında ileri veya geri, X-motorunu çalıştırdığında sağa veya sola gitmesini sağlamaktadır. Duruma göre PIC aynı anda her iki motoru da çalıştırabilmektedir bu da sisteme hem düz (X veya Y yönünde) hem de çapraz (X ve Y yönünde) hareket kabiliyeti sağlamaktadır.

Ayrıca paralel porttan gelen diğer bir pin direk olarak bir adet transistör ve bir adet röle ile sürülen, kalemimizi(aşağı-yukarı) hareket için kullanılan DC motoru tetiklemektedir[şekil-18]. İlaveten biri toprak ve biri de PIC'in bilgisayarla haberleşmesini sağlayan bit olmak üzere bilgisayar ve PIC arasında toprak hariç dördü giriş bir çıkış olmak üzere toplam beş tane pin kullanılmaktadır[şekil-17].

Tabi PIC bilgisayardan aldığı veriyi direk göndermemekte veriyi kendi programı(PIC programı kısmında geniş olarak anlatılacak) sayesinde gerekli işlemleri yaptıktan sonra dörder

bitlik verileri ULN2003 entegrelerine; hangi motora, hangi yöne ve ne kadar süre gideceği bilgisini göndermektedir.

Eğer çizim yapılacaksa, bilgisayar direk olarak paralel porttan DC motora (kalemi kontrol eden motor) aşağı inmesi gerektiği verisini tek bit ile göndermekte ve bu veri bir adet BC237(NPN) transistor sayesinde 5 volta yükseltilip bir adet röleyi(5-volt, 8-bacak) tetiklemekte, DC motor çalışmakta ve kalem aşağı inmektedir[şekil-18].



Şekil - 18 - DC Motorun Çalışma ve Bağlantı Şeması

Çizim yapılmadığı esnada ise bilgisayar kalemi hareket için kullanılan transistöre gönderdiği bilgiyi kesmekte ve kalem, onu geri çeken bir yay yardımıyla hemen yukarı kalkmakta ve çizim işlemi kesilmektedir.

PIC ile bilgisayarın eş zamanlı olarak çalışması için bir adet ready-busy bacağı eklenmiştir. Bilindiği gibi bilgisayar PIC'ten daha hızlı olduğundan dolayı PIC bilgisayar hızına yetişememekte ve bazen veri kayırabilmektedir, fakat bu bit sayesinde bilgisayar PIC'in işini bitirmesini beklemekte, yani motorları her çalıştırıp durdurduktan sonra bilgisayara yeni bilgi için hazır olduğunu bildirmektedir, böylelikle veri kayıpları yaşanmamakta ve sistem daha kararlı çalışmaktadır.

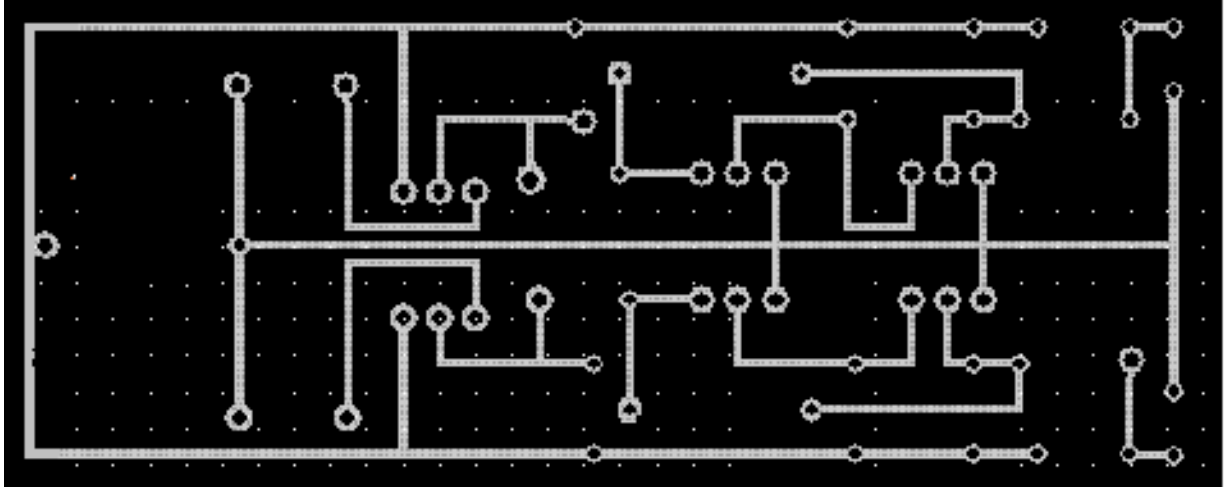
ULN2003 entegreleri ise PIC'ten gelen sürüş bilgisini, yani bit kaydırmasını 12 DC volta yükselterek step motorlarına vermektedir. Eğer ULN2003 kullanılmaz ise PIC'ten gelen 3-5 DC voltluk gerilim ve düşük akım step motorları süremeyecektir, sürse bile sistem için

gereken torku sağlayamayacaktır. Bu sebeple ULN2003 entegre devresi kullanma ihtiyacı doğmaktadır. ULN2003 entegre beslemesi bir diyot(1N4001) ile verilmektedir çünkü entegrenin yapısı gereği diyot kullanılmadığında motorların girişleri toprak olmakta ve motor çalışmamaktadır.

ULN2003'le aynı işi yapabilecek ayrı ayrı beslenmiş transistör grupları da kullanılabilirdi fakat entegre devre hem daha az kaynak gerektirecek hem de senkronize çalışacaktır. Bu sebeple bu entegre devre daha uygun olmaktadır.

Kullanılan step motorlar 7.5° adım boyundadırlar bu da bir tam dönüş için 48 adım hassasiyeti vermektedir, bunun yanında dişli sistemlerinden Y-motoruna bağlı dişlilerden 1/12, X-motoruna bağlı dişlilerden 1/10 kat dönme kazancı sağlamaktadır. Bu da her bir motorda yaklaşık olarak 1-2 mm çizim hassasiyeti sağlamaktadır. Bu değer her ne kadar küçük boyutlardaki; örneğin mobil telefonlar için yeterli bir hassasiyet olmasa da kafi sayılabilecek bir miktardadır.

Basit bir robot devresi için hazırlanıp MiniCiR programında çizilmiş olan bir devre aşağıda görülmektedir

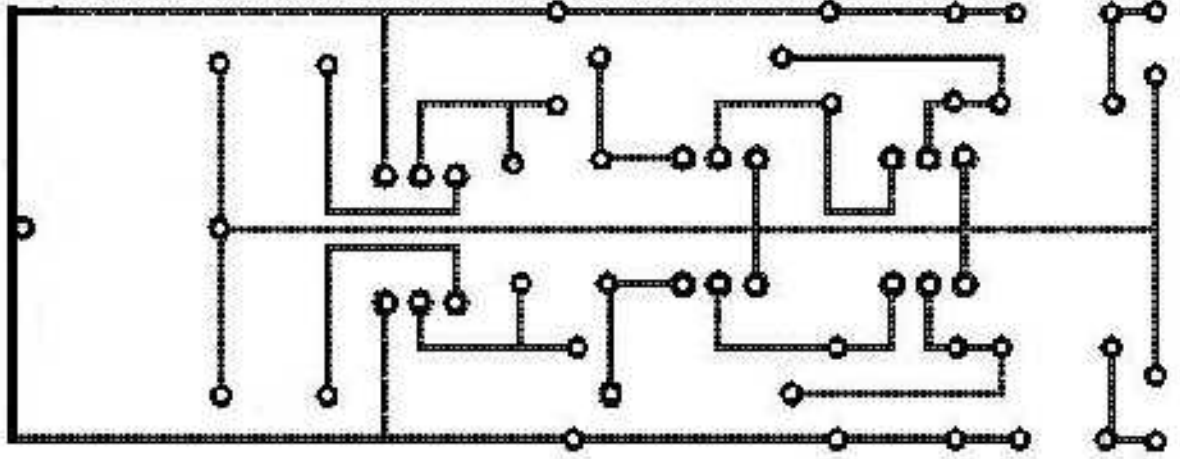


Şekil - 19 - MiniCiR'de Çizilmiş Baskı Devre Örneği

Çizim hassasiyeti kalem inceliğine de bağlıdır. İnce kalem için daha hassas, biraz daha kalın kalem için daha az hassas olmaktadır. Eğer istediğimiz devre çok ince yollara ihtiyaç duymuyorsa, bağlantı yollarımızı kalın çizmekte fayda vardır çünkü bilindiği gibi iletkenlerde direnç, kalınlık arttıkça düşmekte, böylelikle güç kaybı ve beraberinde devrede ısınma problemi de azalmaktadır.

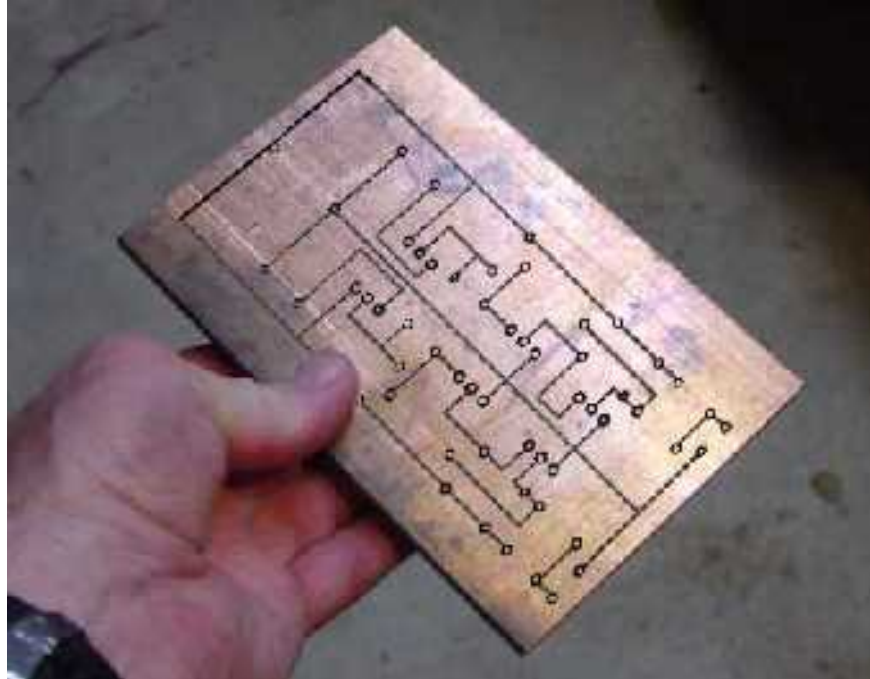
İstedığımız baskı devre bakır levhaya çizildikten sonra kalemimiz harekete başladığı noktaya en kısa yoldan geri dönmektedir. Bu da sistemin anlık dahi olsa veri kaybı yapmadığı yani geçtiği noktaları belleğinde tuttuğunu göstermektedir.

Makinenin, sesle çalışan küçük bir robot için hazırlanmış olan baskı devresini, bakır levhaya aktardığı basit bir örneği gösterilmektedir.



Şekil - 20 – MiniCiR'in Bakır Plakete Çizdiği Baskı Devre Örneği

Şekilde görüldüğü gibi makine, elemanların yerleştirileceği bağlantı uçlarına küçük halkalar koymaktadır. Bu halkalar lehim işlemleri esnasında lehimin bağlantı noktalarına tutunmasını kolaylaştırmakta ve ayrıca lehimin dışarı taşmasını da engellemektedir.



Fotoğraf - 2 - Plakete Çizilmiş Devre Örneđi

Plakete aktarılan Őekil, gerekli kimyasal iŐlemlerden sonra kullanıma hazır hale gelecektir.

Elektronik elemanlarımız lehimlendikten sonra devremiz, test ve kullanım iŐin hazır olmuŐ olacaktır.

3.2. MiniCiR PIC Programı

```
; tanımlamalar
status      EQU      h'03'
porta       EQU      H'05'
portb       EQU      H'06'
trisa       EQU      H'85'
trisb       EQU      H'86'
syc1        equ      h'20'
syc2        equ      h'21'

; portların giriş çıkış ayarları
; porta giriş , portb çıkış yapılıyor

          BSF        status,5
          movlw      b'00001111'
          movwf      trisa
          CLRF       trisb
          BCF        status,5

; analog karşılaştırma özelliği
; devre dışı bırakılıyor

          movlw      h'7'
          movwf      h'1f'

; portlar ileri kullanım için temizleniyor

          CLRF       portb
          CLRF       porta

basla

; Bilgisayardan gelen veri kontrol ediliyor

; 1. adım , porta pin 0 kontrol ediliyor
; 1 ise sol prosedürü çağrılıyor

          movlw      b'00000001'
          subwf      porta,0
          btfsc      status,2
          call       sol

; 2. adım , porta pin 1 kontrol ediliyor
; 1 ise sağ prosedürü çağrılıyor

          movlw      b'00000010'
          subwf      porta,0
          btfsc      status,2
          call       sag

; 3. adım , porta pin 2 kontrol ediliyor
```

; 1 ise yukari prosedürü çağrılıyor

```
movlw    b'00000100'  
subwf    porta,0  
btfsc    status,2  
call     yukari
```

; 4. adım , porta pin 3 kontrol ediliyor

; 1 ise asagi prosedürü çağrılıyor

```
movlw    b'00001000'  
subwf    porta,0  
btfsc    status,2  
call     asagi
```

; 5. adım , porta pin 0 ve 2 kontrol ediliyor

; her ikisi 1 ise solyukari prosedürü çağrılıyor

```
movlw    b'00000101'  
subwf    porta,0  
btfsc    status,2  
call     solyukari
```

; 6. adım , porta pin 0 ve 3 kontrol ediliyor

; her ikisi 1 ise solasagi prosedürü çağrılıyor

```
movlw    b'00001001'  
subwf    porta,0  
btfsc    status,2  
call     solasagi
```

; 7. adım , porta pin 1 ve 2 kontrol ediliyor

; her ikisi 1 ise sagyukari prosedürü çağrılıyor

```
movlw    b'00000110'  
subwf    porta,0  
btfsc    status,2  
call     sagyukari
```

; 8. adım , porta pin 1 ve 3 kontrol ediliyor

; her ikisi 1 ise sagasagi prosedürü çağrılıyor

```
movlw    b'00001010'  
subwf    porta,0  
btfsc    status,2  
call     sagasagi
```

```
goto     basla
```

; prosedür sol : X motoru ile kalemi sola kaydırır

```

sol
; bilgisayara meşgul olduğunu bildir
    bsf        porta,4
    movlw     b'00010000'
    MOVWF    portb
    call     dur
    movlw     b'00110000'
    MOVWF    portb
    call     dur
    movlw     b'00100000'
    MOVWF    portb
    call     dur
    movlw     b'01100000'
    MOVWF    portb
    call     dur
    movlw     b'01000000'
    MOVWF    portb
    call     dur
    movlw     b'11000000'
    MOVWF    portb
    call     dur
    movlw     b'10000000'
    MOVWF    portb
    call     dur
    movlw     b'10010000'
    MOVWF    portb
    call     dur
; bilgisayara hazır olduğunu bildir
    bcf        porta,4
    return

; prosedür sag : X motoru ile kalemi sağa kaydırır

sag
; bilgisayara meşgul olduğunu bildir
    bsf        porta,4
    movlw     b'10010000'
    MOVWF    portb
    call     dur
    movlw     b'10000000'
    MOVWF    portb
    call     dur
    movlw     b'11000000'
    MOVWF    portb
    call     dur
    movlw     b'01000000'
    MOVWF    portb
    call     dur
    movlw     b'01100000'
    MOVWF    portb
    call     dur

```

```
        movlw      b'00100000'  
        MOVWF     portb  
        call      dur  
        movlw      b'00110000'  
        MOVWF     portb  
        call      dur  
        movlw      b'00010000'  
        MOVWF     portb  
        call      dur  
; bilgisayara hazır olduğunu bildir  
        bcf       porta,4  
        return
```

; prosedür yukari : Y motoru ile plakayı ileri kaydırır

yukari

```
; bilgisayara meşgul olduğunu bildir  
        bsf       porta,4  
        movlw      b'00001001'  
        MOVWF     portb  
        call      dur  
        movlw      b'00001000'  
        MOVWF     portb  
        call      dur  
        movlw      b'00001100'  
        MOVWF     portb  
        call      dur  
        movlw      b'00000100'  
        MOVWF     portb  
        call      dur  
        movlw      b'00000110'  
        MOVWF     portb  
        call      dur  
        movlw      b'00000010'  
        MOVWF     portb  
        call      dur  
        movlw      b'00000011'  
        MOVWF     portb  
        call      dur  
        movlw      b'00000001'  
        MOVWF     portb  
        call      dur  
; bilgisayara hazır olduğunu bildir  
        bcf       porta,4  
        return
```

; prosedür asagi : Y motoru ile plakayı geri kaydırır

asagi

```
; bilgisayara meşgul olduğunu bildir  
        bsf       porta,4
```

```

movlw      b'00000001'
MOVWF     portb
call      dur
movlw      b'00000011'
MOVWF     portb
call      dur
movlw      b'00000010'
MOVWF     portb
call      dur
movlw      b'00000110'
MOVWF     portb
call      dur
movlw      b'00000100'
MOVWF     portb
call      dur
movlw      b'00001100'
MOVWF     portb
call      dur
movlw      b'00001000'
MOVWF     portb
call      dur
movlw      b'00001001'
MOVWF     portb
call      dur
; bilgisayara hazır olduğunu bildir
bcf        porta,4
return

```

*; prosedür sagyukari : Y motoru ile plakayı ileri
; kaydırır aynı zamanda X motoru ile kalemi sağa kaydırır*

```

sagyukari
; bilgisayara meşgul olduğunu bildir
bsf        porta,4
movlw      b'10011001'
MOVWF     portb
call      dur
movlw      b'10001000'
MOVWF     portb
call      dur
movlw      b'11001100'
MOVWF     portb
call      dur
movlw      b'01000100'
MOVWF     portb
call      dur
movlw      b'01100110'
MOVWF     portb
call      dur
movlw      b'00100010'
MOVWF     portb

```

```
        call    dur
        movlw   b'00110011'
        MOVWF   portb
        call    dur
        movlw   b'00010001'
        MOVWF   portb
        call    dur
; bilgisayara hazır olduğunu bildir
        bcf     porta,4
        return
```

```
; prosedür solasagi : Y motoru ile plakayı geri kaydırır
; aynı zamanda X motoru ile kalemi sola kaydırır
```

```
solasagi
; bilgisayara meşgul olduğunu bildir
        bsf     porta,4
        movlw   b'00010001'
        MOVWF   portb
        call    dur
        movlw   b'00110011'
        MOVWF   portb
        call    dur
        movlw   b'00100010'
        MOVWF   portb
        call    dur
        movlw   b'01100110'
        MOVWF   portb
        call    dur
        movlw   b'01000100'
        MOVWF   portb
        call    dur
        movlw   b'11001100'
        MOVWF   portb
        call    dur
        movlw   b'10001000'
        MOVWF   portb
        call    dur
        movlw   b'10011001'
        MOVWF   portb
        call    dur
; bilgisayara hazır olduğunu bildir
        bcf     porta,4
        return
```

```
; prosedür sagasagi : Y motoru ile plakayı geri kaydırır
; aynı zamanda X motoru ile kalemi sağa kaydırır
```

```
sagasagi
; bilgisayara meşgul olduğunu bildir
        bsf     porta,4
```

```

movlw      b'10010001'
MOVWF     portb
call      dur
movlw      b'10000011'
MOVWF     portb
call      dur
movlw      b'11000010'
MOVWF     portb
call      dur
movlw      b'01000110'
MOVWF     portb
call      dur
movlw      b'01100100'
MOVWF     portb
call      dur
movlw      b'00101100'
MOVWF     portb
call      dur
movlw      b'00111000'
MOVWF     portb
call      dur
movlw      b'00011001'
MOVWF     portb
call      dur
; bilgisayara hazır olduğunu bildir
bcf        porta,4
return

```

*; prosedür solyukari : Y motoru ile plakayı ileri
; kaydırır aynı zamanda X motoru ile kalemi sola kaydırır*

```

solyukari
; bilgisayara meşgul olduğunu bildir
bsf        porta,4
movlw      b'00011001'
MOVWF     portb
call      dur
movlw      b'00111000'
MOVWF     portb
call      dur
movlw      b'00101100'
MOVWF     portb
call      dur
movlw      b'01100100'
MOVWF     portb
call      dur
movlw      b'01000110'
MOVWF     portb
call      dur
movlw      b'11000010'
MOVWF     portb

```

```
        call    dur
        movlw   b'10000011'
        MOVWF  portb
        call    dur
        movlw   b'10010001'
        MOVWF  portb
        call    dur
; bilgisayara hazır olduğunu bildir
        bcf     porta,4
        return
```

; PIC'i bir süreliğine durdurur

```
dur
        movlw   h'8'
        movwf   syc1
tekrar1
        movlw   h'ff'
        movwf   syc2
tekrar2
        decfsz  syc2,1
        goto    tekrar2
        decfsz  syc1,1
        goto    tekrar1
        return
```

END

3.3 MiniCiR Bilgisayar Programı

```
{kullanilacak kutuphaneler}
uses graph,crt,dos;

{kendi özel tiplerimizi tanımlıyoruz}
type
{mouse cursor şekli ve orijini}
mshape=array [1..9,1..9] of byte;

{alan sınırları için area tipi tanımlıyoruz}
area=record
  {başlangıç [x1,y1] bitiş [x2,y2] }
  x1,y1,x2,y2:integer;
end;

{fare ile tiklanacak butonlar için tip}
button = record
  {durumu , yeniden çizilmesi gerekliliği}
  state,draw:boolean;
  {sınırları}
  x1,y1,x2,y2:integer;
end;

{dosya için tip tanımı}
{çizgiler için }
landc=record
  lines:array[1..1000] of record
    act:boolean;
    xs,ys,xe,ye:integer;
  end;
```

```

{daireler icin }
    circs:array[1..1000] of record
        act:boolean;
        x,y:integer;
    end;
end;

{sabitler}
const
    {mouse imlecinin seklini belirtiyoruz circle}
    mcirc:mshape =
    (
    (0 ,0 ,0 ,15,15,15,0 ,0 ,0 ),
    (0 ,15,0 ,0 ,15,0 ,0 ,15,0 ),
    (0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ),
    (15,0 ,0 ,0 ,0 ,0 ,0 ,0 ,15),
    (15,15,0 ,0 ,0 ,0 ,0 ,15,15),
    (15,0 ,0 ,0 ,0 ,0 ,0 ,0 ,15),
    (0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ),
    (0 ,15,0 ,0 ,15,0 ,0 ,15,0 ),
    (0 ,0 ,0 ,15,15,15,0 ,0 ,0 ));

    {mouse imlecinin seklini belirtiyoruz cross}
    mcross:mshape =
    (
    (0 ,0 ,0 ,15,15,15,0 ,0 ,0 ),
    (0 ,0 ,0 ,0 ,15,0 ,0 ,0 ,0 ),
    (0 ,0 ,0 ,0 ,15,0 ,0 ,0 ,0 ),
    (15,0 ,0 ,0 ,0 ,0 ,0 ,0 ,15),
    (15,15,15,0 ,0 ,0 ,15,15,15),
    (15,0 ,0 ,0 ,0 ,0 ,0 ,0 ,15),
    (0 ,0 ,0 ,0 ,15,0 ,0 ,0 ,0 ),
    (0 ,0 ,0 ,0 ,15,0 ,0 ,0 ,0 ),
    (0 ,0 ,0 ,15,15,15,0 ,0 ,0 ));

```

```
{mouse imlecinin seklini belirtiyoruz arrow}
```

```
marrow:mshape =
```

```
(  
  (15,15,15,15,15,15,15,0 ,0 ),  
  (15,15,15,15,15,0 ,0 ,0 ,0 ),  
  (15,15,15,15,15,0 ,0 ,0 ,0 ),  
  (15,15,15,15,15,15,0 ,0 ,0 ),  
  (15,15,15,15,15,15,15,0 ,0 ),  
  (15,0 ,0 ,15,15,15,15,15,0 ),  
  (15,0 ,0 ,0 ,15,15,15,15,15),  
  (0 ,0 ,0 ,0 ,0 ,15,15,15,0 ),  
  (0 ,0 ,0 ,0 ,0 ,0 ,15,0 ,0 ));
```

```
{tam ekran sinirlari}
```

```
m1_fullscreen:area=(x1:0;y1:0;x2:639;y2:479);
```

```
{cizimalani sinirlari}
```

```
m1_drawscreen:area=(x1:0;y1:0;x2:639;y2:409);
```

```
{menualani sinirlari}
```

```
m1_menuscreen:area=(x1:0;y1:420;x2:639;y2:479);
```

```
linemode=1;
```

```
circmode=2;
```

```
lineeditmode=3;
```

```
circeditmode=4;
```

```
sns=5;
```

```
{degiskenleri tanimliyoruz}
```

```
var
```

```
{grafik modu degiskenleri}
```

```
grDriver: Integer;
```

```
grMode: Integer;
```

```
{genel sayac degiskenleri}
```

```
i,j: Integer;
```

```
{genel konum degiskenleri}
  x1,x2,y1,y2:integer;
  mouseback:array [1..9,1..9] of byte;
  mouseshape:mshape;

{mouse konum degiskenleri x y xeski yeski}
mousex,mousey,mousexold,mouseyold:integer;

{mouse hareket alani limit degiskeni }
  mouselimits:area;

{buttonlar}
lineb:button;

{mouse kullanimi icin register tipi degisken}
r: registers;

{cizgi modu degiskenleri}
pxs,pxe:integer;

{cizim modu degiskenleri}
pencolor:byte;
portval:byte;
act_x,act_y:integer;

{global degiskenler}
activemode:integer;
actx,acty:integer;
cirsize:integer;
cx,cy:integer;
ch:char;
```

```

    {dosya degiskenleri}
    lac:file of landc;
    landcs:landc;
    fn:string;

    {modeshow procedure ileride tanimlanacak}
    {daha oncesinde kullanabilmek icin burada belirtiyoruz}
procedure modeshow; forward;

    {koordinat gosterme alanini hazirliyoruz}
procedure showcoordinate(x,y,bg,fg:word);
var
    mx,my:string;
begin
    {dolgu stili belirliyoruz : 1(tamdolu) renk: bg }
    SetFillStyle(1, bg);    { New style }
    {cerceve ciziyoruz}
    rectangle(x,y,x+83,y+10);
    {cerceve icini dolduruyoruz}
    bar(x+1,y+1,x+82,y+9);
    {renk belirleniyor}
    setcolor(fg);
    {X: yaziliyor}
    outtextxy(x+3,y+2,'X:');
    {Y: yaziliyor}
    outtextxy(x+46,y+2,'Y:');
end;

```

```

{koordinatin devamli gosterimi icin prosedur}
procedure reshowcoordinate(x,y,bg,fg:integer);
{konum icin string tipi degiskenler
outtext komutu string gerektirmektedir.}
var
    mx,my:string;
begin
    {dolgu stili belirliyoruz : 1(tamdolu) renk: bg }
    SetFillStyle(1, bg);
    {eski koordinatlari siliyoruz}
    bar(x+17,y+2,x+45,y+8);
    bar(x+60,y+2,x+82,y+8);
    {renk belirleniyor}
    setcolor(fg);
    {yeni koordinatlari string tipine ceviriyoruz}
    str(mousex,mx);
    str(mousey,my);
    {yeni koordinatlar yaziliyor}
    outtextxy(x+17,y+2,mx);
    outtextxy(x+60,y+2,my);
end;

```

```

{cizimalani noktaları çizme proseduru}
procedure drawgrids;
begin
    {ekran 10 ar pixel uzaklıkta
    x ekseninde 63 y ekseninde 40 noktaya bolunuyor}
    for i:=0 to 410 div sns-2 do
        for j:=0 to 640 div sns-2 do putpixel(5+j*sns,5+i*sns,8);
end;

```

```

{ekran taslagi cizim proseduru}
procedure drawtemplate;
begin
  setcolor(15);
  {cizimalani belirleniyor}
  rectangle(0,0,639,410);
  {menualani belirleniyor}
  rectangle(0,420,639,479);
  {koordinat gostermealani hazirlaniyor}
  showcoordinate(550,425,1,15);
  {cizimalani noktaları ciziliyor}
  drawgrids;

  {Dugme yazilari}
  setcolor(9);
  SetTextStyle(0,1,0);
  outtextxy(215,430,'yeni');
  outtextxy(265,430,'degis');
  outtextxy(315,430,'dosya');
  SetTextStyle(0,0,0);
  outtextxy(327,458,'kaydet');
  outtextxy(340,433,'ac');
  outtextxy(400,433,'yeni');
  outtextxy(400,458,'ciz');
  outtextxy(590,456,'cik');

  {dugme kutucuklari}
  setcolor(8);
  rectangle(580,450,630,470); {quit}
  rectangle(220,427,240,447); {line}
  rectangle(220,452,240,472); {circ}
  rectangle(270,427,290,447); {lineedit}
  rectangle(270,452,290,472); {circedit}
  rectangle(320,427,380,447); {openfile}

```

```

rectangle(320,452,380,472); {savefile}
rectangle(385,427,445,447); {newfile}
rectangle(385,452,445,472); {drawall}

{dugme ic sekilleri}
setcolor(1);
  circle(230,462,2);
  line(225,437,235,437);
setcolor(4);
  circle(280,462,2);
  line(275,437,285,437);
setcolor(15);

{Tanitim yazisi of MiniCiR}
SetTextStyle(5,0,4);
outtextxy(50,415,'MiniCiR');
SetTextStyle(2,0,4);
outtextxy(25,460,'By Muhammed Cevher Ramazan');
SetTextStyle(0,0,1);
end;

{ekran tazeleme fonksiyonu}
procedure restore;
var i:integer;
begin
  setfillstyle(0,0);
  bar(0,0,639,479);
  drawtemplate;
  setcolor(15);
  for i:=1 to 1000 do
  begin
    if landcs.circs[i].act then
circle(landcs.circs[i].x,landcs.circs[i].y,circsize);
  end;
end;

```

```
setcolor(15);
for i:=1 to 1000 do
begin
  if landcs.lines[i].act then
    line(landcs.lines[i].xs,landcs.lines[i].ys,landcs.lines[i]
.xe,landcs.lines[i].ye);
  end;
end;
```

{cizgi ve daireleri temizleme}

```
procedure freelandcs;
var i:integer;
begin
  for i:=1 to 1000 do
  begin
    landcs.circs[i].act:=false;
    landcs.circs[i].x:=0;
    landcs.circs[i].y:=0;
    landcs.lines[i].act:=false;
    landcs.lines[i].xs:=0;
    landcs.lines[i].ys:=0;
    landcs.lines[i].xe:=0;
    landcs.lines[i].ye:=0;
  end;
end;
```

{dosya kaydetme}

```
procedure savefile(fnn:string);
begin
  if fnn<>' ' then begin
    assign(lac,fnn+'.mcr');
```

*{Hata yakalama bolgesi , istenilen dosya
bulunamazsa hatayi bulup programi kesmeyi*

```

    engeller.}
{$I-}
    rewrite(lac);
{$I+}
if IOResult = 0 then
begin
    write(lac,landcs);
    close(lac);
end else
begin

end;
end;
end;

```

```

{dosya cagirma}
procedure loadfile(fnn:string);
begin
    if fnn<>' ' then begin
        assign(lac,fnn+'.mcr');
        {Hata yakalama bolgesi , istenilen dosya
        bulunamazsa hatayi bulup programi kesmeyi
        engeller.}
    {$I-}
        reset(lac);
    {$I+}
if IOResult = 0 then
begin
    read(lac,landcs);
    close(lac);
    restore;
    modeshow;
end;

```

```

end;
end;

{mouse cizim oncesi arkaplani okuyoruz}
procedure getmouseback;
{yerel sayac degiskenleri}
var
  i,j:integer;
begin
  { 9x9 pixel arkaplani okuyoruz}
  { merkezi [mousex,mousey] }
  for i:=1 to 9 do
    for j:=1 to 9 do
      mouseback[j,i]:=getpixel(mousex-5+j,mousey-5+i);
    end;
  end;

  {mouse eski arkaplani ciziyoruz}
procedure drawmouseback;
{yerel sayac degiskenleri}
var
  i,j:integer;
begin
  {9x9 bellekteki arkaplani ciziyoruz}
  {merkezi [mousex,mousey]}
  for i:=1 to 9 do
    for j:=1 to 9 do
      putpixel(mousexold-5+j,mouseyold-5+i,mouseback[j,i]);
    end;
  end;

  {mouse cizim proseduru}
procedure paintmouse;

```

```

{yerel sayac degiskenleri}
var i,j:integer;
begin
{9x9 alana sekli ciziyoruz}
{merkez [mousex,mousey]}
for i:=1 to 9 do
for j:=1 to 9 do
if mouseshape[j,i]<>0 then putpixel(mousex-5+j,mousey-
5+i,mouseshape[j,i]);
end;

```

{mouse kontrol ve gerektiğinde çizim proseduru}

procedure drawmouse;

```

begin
{mouse konumu 33. kesmeden okunuyor}
r.ax:=03;
intr($33,r);
{mouse konumu mousex,mousey ye aktariliyor}
mousex:=r.cx;
mousey:=r.dx;
{mouse limitler disinda ise limit icine aliniyor}
if mousex<mouselimits.x1 then mousex:=mouselimits.x1;
if mousey<mouselimits.y1 then mousey:=mouselimits.y1;
if mousex>mouselimits.x2 then mousex:=mouselimits.x2;
if mousey>mouselimits.y2 then mousey:=mouselimits.y2;
{mouse çizim alanında ise noktalar (grid)
uzerine konumlandırıyoruz}
if mousey<410 then
begin
mousex:=mousex - mousex mod sns +5;
mousey:=mousey - mousey mod sns +5;
if activemode=linemode then mouseshape:=mccross;
if activemode=circmode then mouseshape:=mccirc;
if activemode=lineditmode then mouseshape:=mccross;

```

```

    if activemode=circreditmode then mouseshape:=mcirc;
end else    mouseshape:=marrow;

{mouse konumunun degismesini kontrol ediyoruz
   sadece yeri degismis ise yeniden ciziyoruz }
if(mousex<>mousexold) or (mousey<>mouseyold) then
begin
    {eski mouse arkaplanini yerine ciziyoruz}
    drawmouseback;

    {cizim alaninda ise koordinati yazdiriyoruz}
    if(mousex>0) and (mousex<620) and (mousey>0) and (mousey<410) then
    reshowcoordinate(550,425,1,15);

    {mouse arkaplanini aliyoruz}
    getmouseback;

    {mouse imlecini ciziyoruz, cool }
    paintmouse;

    {eski konum degiskenlerini yeniliyoruz}
    mousexold:=mousex;
    mouseyold:=mousey;
end;
end;

```

```

{en iyi konum karar mekanizmasi
yatay dikey 45° acili cizim isteklerimizi
otomatik olarak algilar}

```

```

procedure bestplace;

```

```

{yerel degiskenler

```

```

fx/fy x/y konumundaki yerdegistirme

```

```

fxs/fys x/y konumundaki degisimin isareti (sign)

```

```

fmin - x ve y yerdegistirmelerinden minimum olan icin.}

```

```

var

```

```

    fx,fy,fmin,fxs,fys:integer;

```

```

begin

```

```

    x2:=x2-x2 mod sns;

```

```
y2:=y2-y2 mod sns;
```

```
{x koordinatındaki yerdegistirme miktarı  
mutlak deger olarak hesapliyoruz}
```

```
fx:=abs(x2-x1);
```

```
{y koordinatındaki yerdegistirme miktarı  
mutlak deger olarak hesapliyoruz}
```

```
fy:=abs(y2-y1);
```

```
{fx in isareti bulunuyor-fxs}
```

```
if fx<>0 then fxs:=(x2-x1) div fx;
```

```
{fy nin isareti bulunuyor-fys}
```

```
if fy<>0 then fys:=(y2-y1) div fy;
```

```
{fx ve fy den kucuk olan bulunuyor}
```

```
if fx<fy then fmin:=fx else fmin:=fy;
```

```
{x koordinatındaki degisim cok ise}
```

```
if (fx>fy) then
```

```
begin
```

```
{x degisimi ynin 2 katından fazla ise}
```

```
if fx>fy*2 then
```

```
  y2:=y1
```

```
  else
```

```
{x degisimi ynin 2 katından fazla degil ise}
```

```
  x2:=x1+fxs*fmin;
```

```
end else
```

```
{y koordinatındaki degisim cok ise}
```

```
begin
```

```
{y degisimi xin 2 katından fazla ise}
```

```
if fy>fx*2 then
```

```
  x2:=x1
```

```
  else
```

```
{y degisimi xin 2 katından fazla degil ise}
```

```
  y2:=y1+fys*fmin;
```

```
end;
```

```

    {noktalar (grid) uzerine konumlandir }
end;

{daha once kullanilmamis siradaki bos çizgiyi buluyoruz}
function freeline:integer;
var i:integer;
begin
    for i:=1 to 100 do begin if not landcs.lines[i].act then
break; end;
        freeline:=i;
end;

{fare belirtilen konum icinde mi?}
function ismouseover(x1,y1,x2,y2:integer):boolean;
begin
    if (mousex>x1) and (mousex<x2) and (mousey>y1) and (mousey<y2) then
        ismouseover:=true else ismouseover:=false;
end;

{dosya acma buttonu}
procedure buttonopenfile;
begin
    if ismouseover(320,427,380,447) then
begin
    {Cikis buttonu}
    drawmouseback;
    setcolor(15);
    rectangle(320,427,380,447);
        getmouseback;
        paintmouse;
repeat
    drawmouse;
    if r.bx=1 then

```

```

begin
rectangle(470,445,570,465);
gotoxy(60,29);
readln(fn);
loadfile(fn);
setfillstyle(0,0);
bar(470,445,570,465);
end;

if r.bx=1 then
until not ismouseover(320,427,380,447);
drawmouseback;
setcolor(8);
rectangle(320,427,380,447);
setcolor(15);
    getmouseback;
    paintmouse;
end;
end;

```

```

{dosya kaydet buttonu}
procedure buttonsavfile;
begin
if ismouseover(320,452,380,472) then
begin
    {Cikis buttonu}
drawmouseback;
setcolor(15);
rectangle(320,452,380,472);
    getmouseback;
    paintmouse;
repeat
drawmouse;

```

```

if r.bx=1 then
begin
rectangle(470,445,570,465);
gotoxy(60,29);
readln(fn);
savefile(fn);
setfillstyle(0,0);
bar(470,445,570,465);
end;
until not ismouseover(320,452,380,472);
drawmouseback;
setcolor(8);
rectangle(320,452,380,472);
setcolor(15);
    getmouseback;
    paintmouse;
end;
end;

{cizdirme bolumu GO ve ek fonksiyonlar}

{kafa belirtilen konumda mi?}
function isonplace(cx,cy,dx,dy:integer):boolean;
begin
    if (cx=dx)and(cy=dy) then isonplace:=true
        else isonplace:=false;
end;

{kafa x konumu belirtilen konumda mi?}
function isonx(cx,cy,dx,dy:integer):integer;
begin
    if cx=dx then isonx:=0;
    if cx>dx then isonx:=-1;

```

```

    if cx<dx then isonx:=1;
end;

{kafa y konumu belirtilen konumda mi?}
function isony(cx,cy,dx,dy:integer):integer;
begin
    if cy=dy then isony:=0;
    if cy>dy then isony:=-1;
    if cy<dy then isony:=1;
end;

{bit islemleri icin fonksiyon}
function bit(n:byte):byte;
var
    i2,bitt:integer;
begin
    bitt:=1;
    for i2:=1 to n do bitt:=bitt*2;
    bit:=bitt;
end;

{biti 1 yap}
function biton(n:byte):byte;
begin
    portval:=portval or bit(n);
    port[$378]:=portval;
end;

{biti 0 yap}
function bitoff(n:byte):byte;
begin
    portval:=portval and (not bit(n));
    port[$378]:=portval;

```

```
end;
```

```
{kafayi belirtilen noktaya konumlandir}
```

```
function go(dx,dy:integer):boolean;
```

```
var
```

```
    color,wait:byte;
```

```
begin
```

```
repeat
```

```
    if isonx(cx,cy,dx,dy)=1 then biton(0);
```

```
    if isonx(cx,cy,dx,dy)=-1 then biton(1);
```

```
    if isony(cx,cy,dx,dy)=1 then biton(3);
```

```
    if isony(cx,cy,dx,dy)=-1 then biton(2);
```

```
    cx:=cx+isonx(cx,cy,dx,dy);
```

```
    cy:=cy+isony(cx,cy,dx,dy);
```

```
    color:=getpixel(cx,cy);
```

```
    putpixel(cx,cy,4);
```

```
    delay(10);
```

```
    bitoff(0); bitoff(1); bitoff(2); bitoff(3);
```

```
    repeat
```

```
        wait:=port[$379];
```

```
    until wait>128;
```

```
    putpixel(cx,cy,color);
```

```
    if pencolor<>0 then putpixel(cx,cy,pencolor);
```

```
    delay(100);
```

```
until isonplace(cx,cy,dx,dy);
```

```
end;
```

```
{kalemi kaldir}
```

```
procedure pen_up;
```

```
begin
```

```
    pencolor:=0;
```

```
    bitoff(6);
```

```
end;
```

```
{kalemi indir}
```

```
procedure pen_down;
```

```
begin
```

```
  pencolor:=4;
```

```
  biton(6);
```

```
end;
```

```
{ekrandaki herseyi cizmeye basla}
```

```
procedure draw_all;
```

```
begin
```

```
  pen_up;
```

```
  for j:=1 to 1000 do if landcs.lines[j].act=true then
```

```
  begin
```

```
    go(landcs.lines[j].xs,landcs.lines[j].ys);
```

```
    pen_down;
```

```
    delay(30000);
```

```
    go(landcs.lines[j].xe,landcs.lines[j].ye);
```

```
    pen_up;
```

```
    delay(1000);
```

```
    if keypressed then ch:=readkey;
```

```
    if ch=#27 then break;
```

```
      {ESC tusuna basarak cizimi kesme}
```

```
  end;
```

```
  for j:=1 to 1000 do if landcs.circs[j].act=true then
```

```
  begin
```

```
    go(landcs.circs[j].x-2,landcs.circs[j].y);
```

```
    pen_down;
```

```
    delay(30000);
```

```
    go(landcs.circs[j].x-1,landcs.circs[j].y+1);
```

```
    go(landcs.circs[j].x,landcs.circs[j].y+2);
```

```
    go(landcs.circs[j].x+1,landcs.circs[j].y+1);
```

```

go(landcs.circs[j].x+2,landcs.circs[j].y);
go(landcs.circs[j].x+1,landcs.circs[j].y-1);
go(landcs.circs[j].x,landcs.circs[j].y-2);
go(landcs.circs[j].x-1,landcs.circs[j].y-1);
go(landcs.circs[j].x-2,landcs.circs[j].y);
pen_up;
delay(1000);
if keypressed then ch:=readkey;
if ch=#27 then break;
  {ESC tusuna basarak cizimi kesme }
end;

go(0,0);
restore;
modeshow;
end;

{ciz buttonu}
procedure buttondraw;
begin
if ismouseover(385,452,445,472) then
begin
  {Cikis buttonu}
drawmouseback;
setcolor(15);
rectangle(385,452,445,472);
  getmouseback;
  paintmouse;
repeat
drawmouse;
if r.bx=1 then
begin
draw_all;
end;
end;

```

```
until not ismouseover(385,452,445,472);
  drawmouseback;
  setcolor(8);
  rectangle(385,452,445,472);
  getmouseback;
  paintmouse;
end; end;
```

{aktif ve deaktif modlari goster}

procedure modeshow;

begin

```
  drawmouseback;
  setcolor(14);
  if activemode=circmode      then  rectangle(220,452,240,472);
  if activemode=circreditmode then  rectangle(270,452,290,472);
  if activemode=linemode      then  rectangle(220,427,240,447);
  if activemode=lineeditmode  then  rectangle(270,427,290,447);
  setcolor(8);
  if not(activemode=circmode)  then
rectangle(220,452,240,472);
  if not(activemode=circreditmode) then
rectangle(270,452,290,472);
  if not(activemode=linemode)  then
rectangle(220,427,240,447);
  if not(activemode=lineeditmode) then
rectangle(270,427,290,447);
  getmouseback;
  paintmouse;
end;
```

{daire buttonu}

procedure buttoncirc;

begin

```
if ismouseover(220,452,240,472) then
```

```

begin
  drawmouseback;
  setcolor(15);
  rectangle(220,452,240,472);
  getmouseback;
  paintmouse;
repeat
  drawmouse;
  if r.bx=1 then
  begin
    activemode:=circmode;
    modeshow;
  end;
until not ismouseover(220,452,240,472);
  modeshow;
end;
end;

```

{daire duzelt buttonu}

```

procedure buttoncircedit;
begin
  if ismouseover(270,452,290,472) then
  begin
    drawmouseback;
    setcolor(15);
    rectangle(270,452,290,472);
    getmouseback;
    paintmouse;
  repeat
    drawmouse;
    if r.bx=1 then
    begin
      activemode:=circeditmode;

```

```
    modeshow;
end;
until not ismouseover(270,452,290,472);
    modeshow;
end;
end;
```

```
{cizgi buttonu}
procedure buttonline;
begin
if ismouseover(220,427,240,447) then
begin
    drawmouseback;
    setcolor(15);
    rectangle(220,427,240,447);
    getmouseback;
    paintmouse;
repeat
    drawmouse;
    if r.bx=1 then
    begin
        activemode:=linemode;
        modeshow;
    end;
until not ismouseover(220,427,240,447);
    modeshow;
end;
end;
```

```
{cizgi duzelt buttonu}
procedure buttonlineedit;
begin
```

```

if ismouseover(270,427,290,447) then
begin
  drawmouseback;
  setcolor(15);
  rectangle(270,427,290,447);
  getmouseback;
  paintmouse;
repeat
  drawmouse;
  if r.bx=1 then
  begin
    activemode:=lineeditmode;
    modeshow;
  end;
until not ismouseover(270,427,290,447);
  modeshow;
end;
end;

```

{yeni dosya buttonu}

```

procedure buttonnewfile;
begin
if ismouseover(385,427,445,447) then
begin
  drawmouseback;
  setcolor(15);
  rectangle(385,427,445,447);
  getmouseback;
  paintmouse;
repeat
  drawmouse;
  if r.bx=1 then
  begin

```

```

    freelandcs;
    restore;
    modeshow;
    end;
until not ismouseover(385,427,445,447);
    drawmouseback;
    setcolor(8);
    rectangle(385,427,445,447);
    setcolor(15);
        getmouseback;
        paintmouse;
end;
end;

{cikis buttonu}
procedure buttonquit;
begin
if ismouseover(580,450,630,470) then
begin
    drawmouseback;
    setcolor(15);
    rectangle(580,450,630,470);
        getmouseback;
        paintmouse;
repeat
    drawmouse;
    if r.bx=1 then begin closegraph; halt;end;
until not ismouseover(580,450,630,470);
    drawmouseback;
    setcolor(8);
    rectangle(580,450,630,470);
    setcolor(15);
        getmouseback;

```

```

    paintmouse;
end;
end;

{daire yok mu?}
function circnonexist(x,y:integer):boolean;
var i:integer;
begin
    circnonexist:=true;
    for i:=1 to 1000 do if
        (landcs.circs[i].x=x)and(landcs.circs[i].y=y) then
            circnonexist:=false;
        end;
    end;

    {daire numarasi }
function circnum(x,y:integer):integer;
var i:integer;
begin
    circnum:=0;
    for i:=1 to 1000 do if
        (landcs.circs[i].x=x)and(landcs.circs[i].y=y)and(landcs.circs[
        i].act=true) then break;
        if i<>1000 then circnum:=i;
        end;
        {kullanilmamis ilk daire numarasi}
function freecirc:integer;
var i:integer;
begin
    for i:=1 to 1000 do if landcs.circs[i].act=false then break;
        freecirc:=i;
    end;

    {daire ciz}
procedure drawcirc;

```

```

var fcirc:integer;
begin
if
(r.bx=1) and (ismouseover(0,0,639,409)) and (circnonexist(mousex,mousey)) then
begin
drawmouseback;
setcolor(15);
fcirc:=freecirc;
circle(mousex,mousey,circsize);
landcs.circs[fcirc].act:=true;
landcs.circs[fcirc].x:=mousex;
landcs.circs[fcirc].y:=mousey;
getmouseback;
paintmouse;
end;
end;

```

{d aire duzelt}

```

procedure editcirc;
var acirc:integer;
begin
if
(r.bx=1) and (ismouseover(0,0,639,409)) and (circnum(mousex,mousey) <> 0) then
begin
drawmouseback;
setcolor(4);
circle(mousex,mousey,circsize);
acirc:=circnum(mousex,mousey);
repeat
r.bx:=0;

```

```

    r.ax:=3;intr($33,r);
until (r.bx=3) or (r.bx=2);
if r.bx=2 then begin
    setcolor(15);
    circle(landcs.circs[acirc].x,landcs.circs[acirc].y,circsize
);
end;
if r.bx=3 then begin
    landcs.circs[acirc].act:=false;
    setcolor(0);
    circle(landcs.circs[acirc].x,landcs.circs[acirc].y,circsize
);
    landcs.circs[acirc].x:=0;
    landcs.circs[acirc].y:=0;
end;
getmouseback;
paintmouse;
end;
end;

```

{cizgiyi hafizaya ciz}

```

function drawlinemem(x1,y1,x2,y2:integer):boolean;
var fline:integer;
begin
    fline:=freeline;
    landcs.lines[fline].act:=true;
    landcs.lines[fline].xs:=x1;
    landcs.lines[fline].ys:=y1;
    landcs.lines[fline].xe:=x2;
    landcs.lines[fline].ye:=y2;
end;

```

{cizgiyi hafizadan sil}

```

procedure dellinemem(fline:integer);
begin
  landcs.lines[fline].act:=false;
  landcs.lines[fline].xs:=0;
  landcs.lines[fline].ys:=0;
  landcs.lines[fline].xe:=0;
  landcs.lines[fline].ye:=0;
end;

{cizgi ciz}
procedure drawline;
begin

if (r.bx=1) and (ismouseover(0,0,639,409)) then
begin
  if r.bx=1 then
  begin
    x1:=mousex;
    y1:=mousey;
    x1:=x1-x1 mod sns;
    y1:=y1-y1 mod sns;
    mouselimits:=ml_drawscreen;
    r.bx:=0;
    repeat
      drawmouse;
      x2:=mousex;
      y2:=mousey;
      bestplace;
      pxe:=getpixel(x2,y2);
      pxs:=getpixel(x1,y1);
      putpixel(x2,y2,9);
      putpixel(x1,y1,9);
      delay(500);
      putpixel(x2,y2,pxe);

```

```

    putpixel(x1,y1,pxs);
until (r.bx=0) or (r.bx=3);

if (r.bx=0) and ((x1<>x2) or (y1<>y2)) then
begin
    drawlinemem(x1,y1,x2,y2);
    drawmouseback;
    line(x1,y1,x2,y2);
    getmouseback;
    paintmouse;
end;
    mousetimits:=ml_fullscreen;
end;
end;
end;

```

{cizgi tipi}

```

function linetype(ln:integer):byte;
begin
    if landcs.lines[ln].xs=landcs.lines[ln].xe then linetype:=1;
    if landcs.lines[ln].ys=landcs.lines[ln].ye then linetype:=2;
    if
(landcs.lines[ln].xs<>landcs.lines[ln].xe) and (landcs.lines[ln]
.ys<>landcs.lines[ln].ye)
        then linetype:=3;
end;

```

{arasinda mi?}

```

function isbetween(cntv,c1,c2:integer):boolean;

```

```

begin
  isbetween:=false;
  if c1<c2 then if (cntv>c1)and(cntv<c2)then isbetween:=true;
  if c1>c2 then if (cntv<c1)and(cntv>c2)then isbetween:=true;
end;

{maksimum hangisi}
function max(a,b:integer):integer;
begin
  max:=a;
  if a<b then max:=b;
end;

{minimum hangisi}
function min(a,b:integer):integer;
begin
  min:=a;
  if a>b then min:=b;
end;

function onlinenum:integer;
var
  i:integer;
begin
  onlinenum:=0;
  for i:=1 to 1000 do
    begin
      if linetype(i)=1 then
        if
(mousex=landcs.lines[i].xs)and(isbetween(mousey,landcs.lines[i
].ys,landcs.lines[i].ye)) then
          begin onlinenum:=i; break; end;
      if linetype(i)=2 then

```

```

    if
(mousey=landcs.lines[i].ys) and (isbetween(mousex,landcs.lines[i
].xs,landcs.lines[i].xe)) then
    begin onlinenum:=i; break; end;
    if linetype(i)=3 then
    if
(isbetween(mousey,landcs.lines[i].ys,landcs.lines[i].ye))
    and
(isbetween(mousex,landcs.lines[i].xs,landcs.lines[i].xe))
    and ((mousex-min(landcs.lines[i].xs,landcs.lines[i].xe)=
    mousey-min(landcs.lines[i].ys,landcs.lines[i].ye))
    or(mousex-min(landcs.lines[i].xs,landcs.lines[i].xe)=
    max(landcs.lines[i].ys,landcs.lines[i].ye)-mousey ))
then
    begin onlinenum:=i;break; end;
end;
end;

```

{cizgi duzelt}

procedure editline;

var

aline:integer;

begin

aline:=onlinenum;

if (r.bx=1) and (ismouseover(0,0,639,409)) and (aline<>0) then

begin

drawmouseback;

setcolor(4);

line(landcs.lines[aline].xs,landcs.lines[aline].ys,landcs.lin
es[aline].xe,landcs.lines[aline].ye);

repeat

r.ax:=3;

intr(\$33,r);

until (r.bx=2) or (r.bx=3);

```

if (r.bx=3) then
begin
  dellinemem(aline);
  restore;
  getmouseback;
  paintmouse;
end;
if r.bx=2 then
begin
  setcolor(15);
line(landcs.lines[aline].xs,landcs.lines[aline].ys,landcs.lines[aline].xe,landcs.lines[aline].ye);
  getmouseback;
  paintmouse;
end;
end;
end;
{ana program blogu}
begin
  {Grafik moduna gecis yapiliyor}
  grDriver := Detect;
  InitGraph(grDriver, grMode, ' ');
  {mouse limitleri tam ekrana ayarlaniyor}
  mouselimits:=ml_fullscreen;
  {ekran taslagi hazirlaniyor}
  drawtemplate;
  modeshow;
  circline:=2;
  r.ax:=00;
  intr($33,r);
  {ilk mouse artalani okunuyor}
  getmouseback;
  {mouse sekli cross ayarlaniyor}
  mouseshape:=mccross;

```

```

    activemode:=linemode;
    modeshow;
    act_x:=0; act_y:=0;
repeat
    {mouse ciziliyor}
    drawmouse;

    {buttonlar}
    buttonquit;
    buttonline;
    buttoncirc;
    buttonlineedit;
    buttoncircedit;
    buttonopenfile;
    buttonsavfile;
    buttonnewfile;
    buttondraw;

    {gorevler}
    if activemode=linemode then drawline;
    if activemode=circmode then drawcirc;
    if activemode=lineeditmode then editline;
    if activemode=circeditmode then editcirc;

    {tusla kontrol}
    if keypressed then ch:=readkey;
    if ch='4' then begin biton(0); delay(100); end;
    if ch='6' then begin biton(1); delay(100); end;
    if ch='8' then begin biton(3); delay(100); end;
    if ch='2' then begin biton(2); delay(100); end;
    if ch='5' then begin biton(6); end;
    if ch='0' then begin bitoff(6); end;
    ch:=#0;

```

*{Motorların çalışması gerekmediği sürece bitler 0 kalacak
böylece motorlar çalışmayacak}*

bitoff(0);bitoff(1);bitoff(2);bitoff(3);

{bitirme!,sonsuz döngü}

{program içinde çık butonu ile bitirilecek}

until 1=0;

{grafik modundan çıkış}

closegraph;

{ekrani temizle}

clrscr;

{programi bitir}

end.

KAYNAKLAR

- 1.) Elektronik Devre Elemanları Test Yöntemleri ve Devre Uygulamaları
Harun BAYRAM
- 2.) Pascal - Turbo Pascal
Mithat UYSAL
- 3.) Turbo Pascal
Bahattin BAYBURAN
- 4.) Turbo Pascal
Ömer AKGÖBEK
- 5.) PIC Programlama
Orhan ALTINBAŞAK
- 6.) Datasheet : ULN2003
- 7.) Datasheet : PIC16F62X
- 8.) <http://www.wikipedia.org>
- 9.) <http://www.microchip.com>
- 10.) <http://www.ams2000.com> – Stepper Motor Basics