Teknoloji Fakültesi Elektrik Elektronik Mühendisliği Bölümü

EE-302
Mikroişlemciler

Tuş Takımı ve GLCD Uygulamaları

12. Hafta

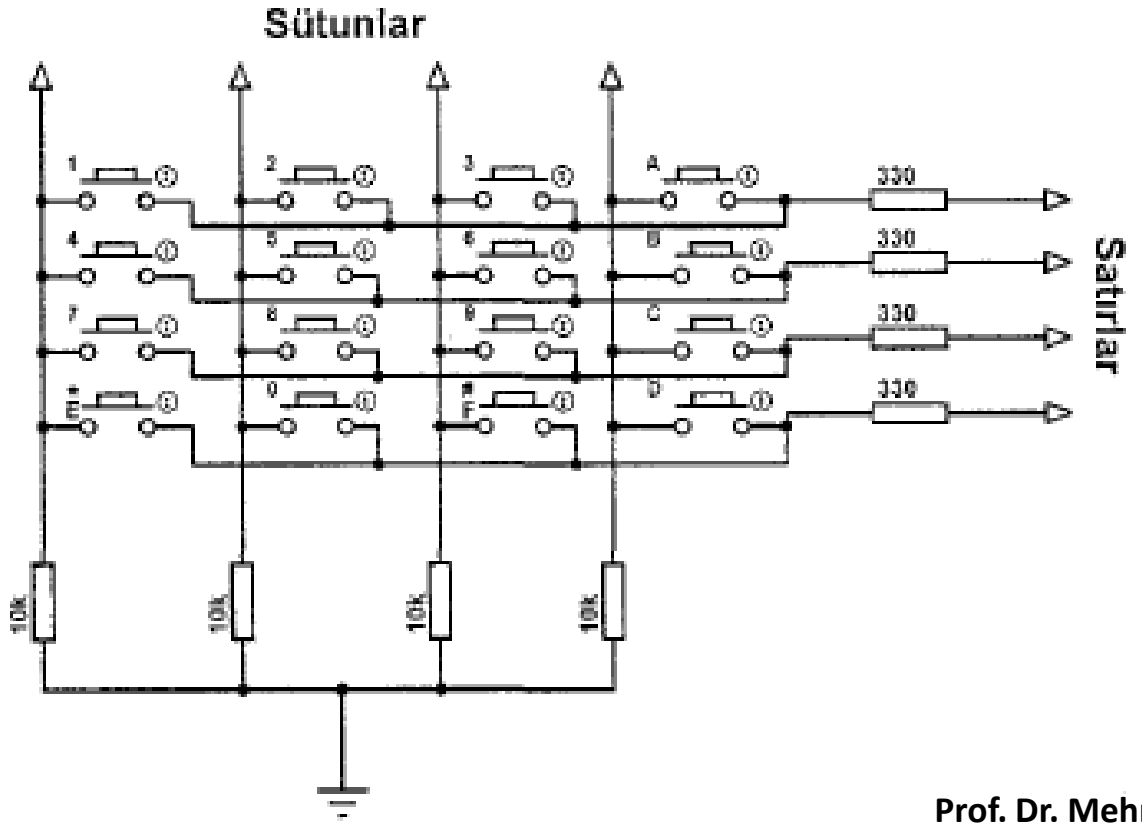
Prof. Dr. Mehmet DEMİRTAŞ

TUŞ TAKIMI (KEYPAD) HAKKINDA BİLGİ

- Kontrol sistemlerinde dış dünyadan insanlar tarafından veri girişleri genellikle tuş takımı (keypad-klavye) ile yapılır. Tuş takımı butonlarla gerçeklestirilebileceği gibi çeşitli hazır tuş takımları piyasada bulunmaktadır. Tuş takımı isimlendirmelerinde ilk sayı satır, ikinci sayı ise sütun sayısını belirtir. Örnegin 4x3'lük bir keypad, 4 satır ve 3 sütundur.

TUŞ TAKIMI (KEYPAD)

Tuş takımında hangi tuşa basıldığını bulmak için çeşitli yöntemler kullanılabilir. Bu yöntemlerden biri tarama yöntemidir. Şekil 2'de butonlarla yapılmış 4x4 tuş takımı görülmektedir. Butonların bir ucu satır kısmına, bir ucu da sütun kısmına bağlıdır. Denetleyici ile tarama yapılırken satırlar çıkış, sütunlar ise giriş olarak tanımlanır.



TUŞ TAKIMI (KEYPAD)

Sütunlarda hep lojik-0 (GND-şase) vardır. Hangi tuşa basıldığını anlamak için önce satırlardan biri lojik-1 diğerleri lojik-0 yapılır. Sonra sütunlar okunur, hangi giriş lojik-1 ise o satıra ait sütundaki tuşa basilmiş demektir. İstenen tuşa hangi değerin verileceği programcıya aittir. Şekil.2'deki bağlantıda 1. satir lojik-1 diğer satırlar lojik-0 iken, 1. satırda 3 nolu tuşa basıldığında 3.sütunda lojik- 1 bilgisi okunur. Böylece basılan tuş bulunabilir. Tabi ki 1. satırda değil de diğer satırlardaki tuşlara basılmışsa algılama yapılamaz. Bu nedenle 1. satırdan sonra sıra ile bir satir lojik- 1, diğer satırlar lojik-0 yapılarak tüm sütunlar okunur. Bahsedilen tarama işlemi bu şekilde yapılmaktadır. Tablo-1'de, Şekil-2'teki tuş takımına göre uyarlanmış tarama bilgileri verilmiştir.

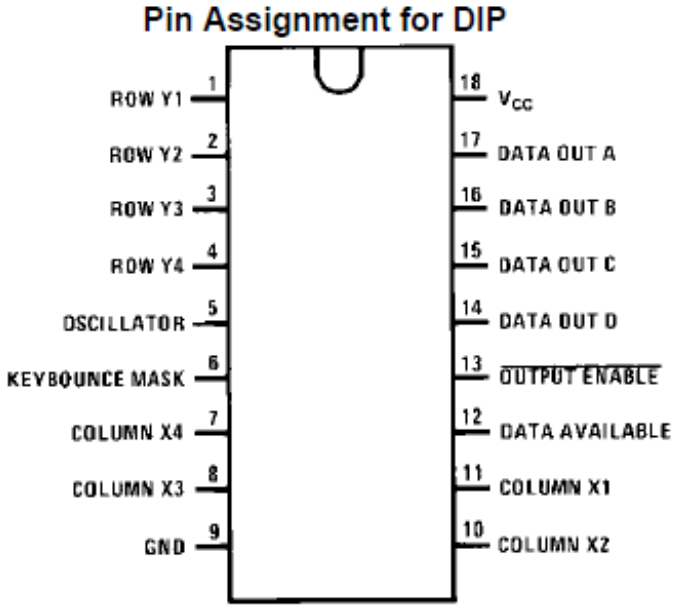
Satırlar				Sütunlar				Basılan Tuş
4	3	2	1	4	3	2	1	
0	0	0	1	0	0	0	1	1
0	0	0	1	0	0	1	0	2
0	0	0	1	0	1	0	0	3
0	0	0	1	1	0	0	0	A
0	0	1	0	0	0	0	1	4
0	0	1	0	0	0	1	0	5
0	0	1	0	0	1	0	0	6
0	0	1	0	1	0	0	0	B
0	1	0	0	0	0	0	1	7
0	1	0	0	0	0	1	0	8
0	1	0	0	0	1	0	0	9
0	1	0	0	1	0	0	0	C
1	0	0	0	0	0	0	1	*, E
1	0	0	0	0	0	1	0	0
1	0	0	0	0	1	0	0	#, F
1	0	0	0	1	0	0	0	D

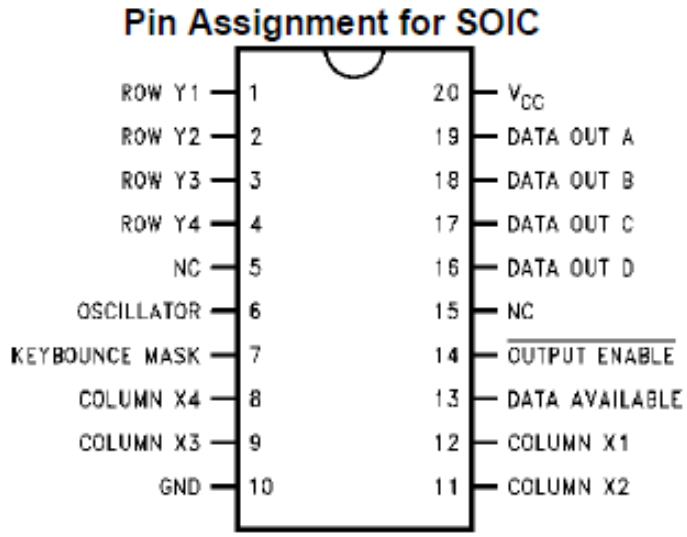
TUŞ TAKIMI (KEYPAD)

- Butona basıldığında ve bırakıldığında bir ark (parazit) meydana gelir. Buna tuş sıçraması da (key debounce) denilir. Şekil 4'te örnek bir tuş sıçraması görülmektedir. Bu sıçramayı önlemek için programda gerekli önlemler alınmalıdır. Tedbir olarak butona basıldıktan sonra 15-20 msn gecikme verilmesi gerekir veya butondan el çekilene kadar içinden çıkılmayacak bir döngü kurulmalıdır. Ayrıca tuş takımında aynı anda iki tuşa birden basılabilir. Bu gibi durumlarda hangi tuşun geçerli olacağı programla belirtilerek istenmeyen durumlar önlenmelidir.

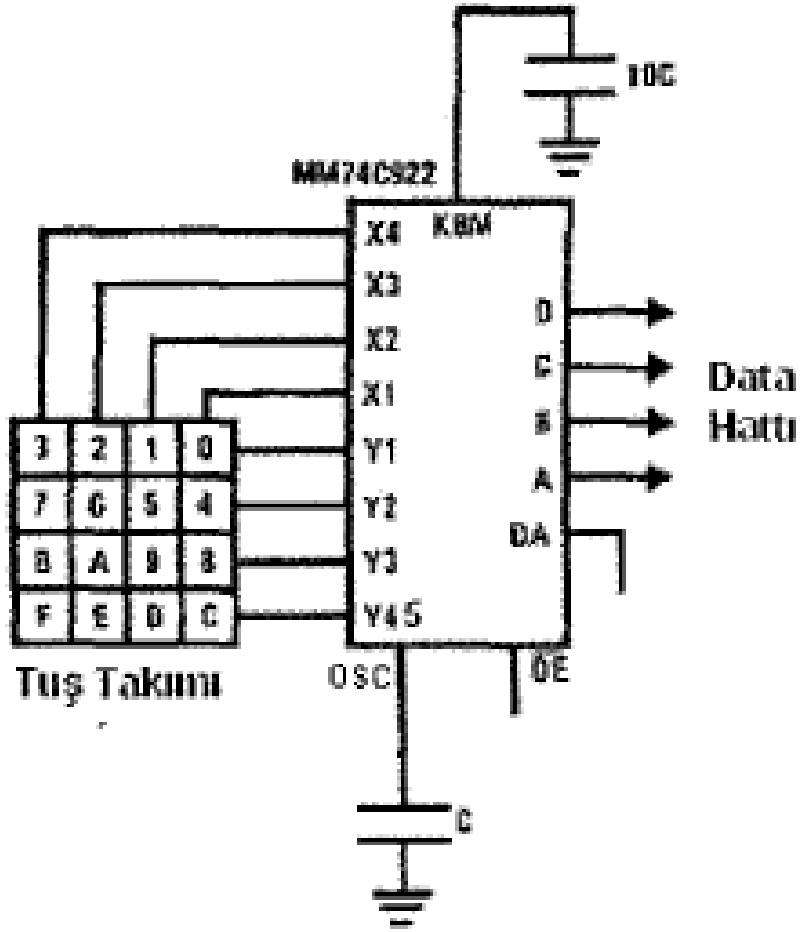
74C922 KEY ENKODER ENTEGRESI

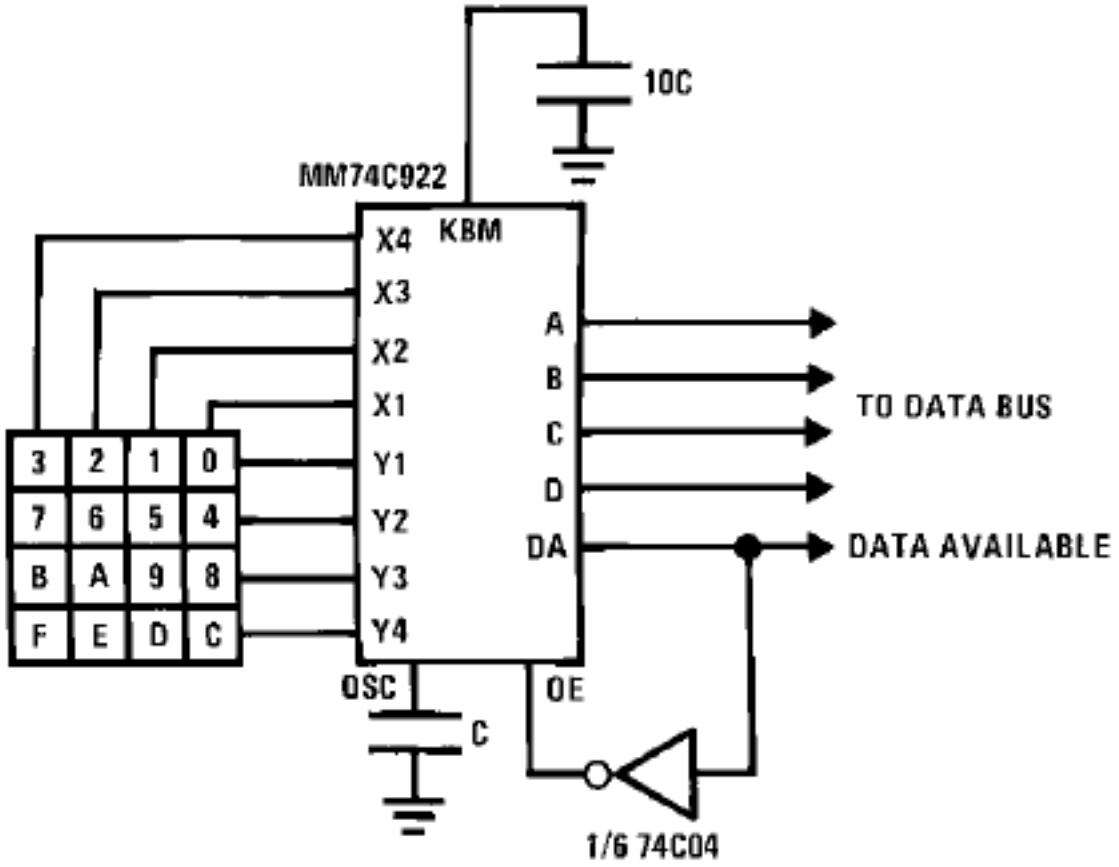
- Tuş tarama işlemi bir önceki uygulamada yapıldığı gibi kullanıldığında denetleyiciyi her zaman meşgul eder. Yani tuş tarama işleminin sürekli olarak yapılması gerekmektedir. Aynı anda başka işlemlerin de yapılacağı bir uygulamada bu yöntem sorunlara yol açabilir. Aynı zamanda önceki uygulamada tuş tarama işlemi için 8 adet pin kullanılmaktaydı, bu da port ihtiyacının fazla olduğu uygulamalarda sıkıntıya yol açabilir. Bu nedenle tuş takımı tarama işlemlerinde 74C922, 16 Key Encoder entegresi kullanılabilir. Bu entegre sayesinde tarama işlemi için denetleyici tümüyle meşgul edilmeyecek ve ayni zamanda fazla pin de kullanılm





74C922 KEY ENKODER ENTEGRESI





74C922 KEY ENKODER ENTEGRESI

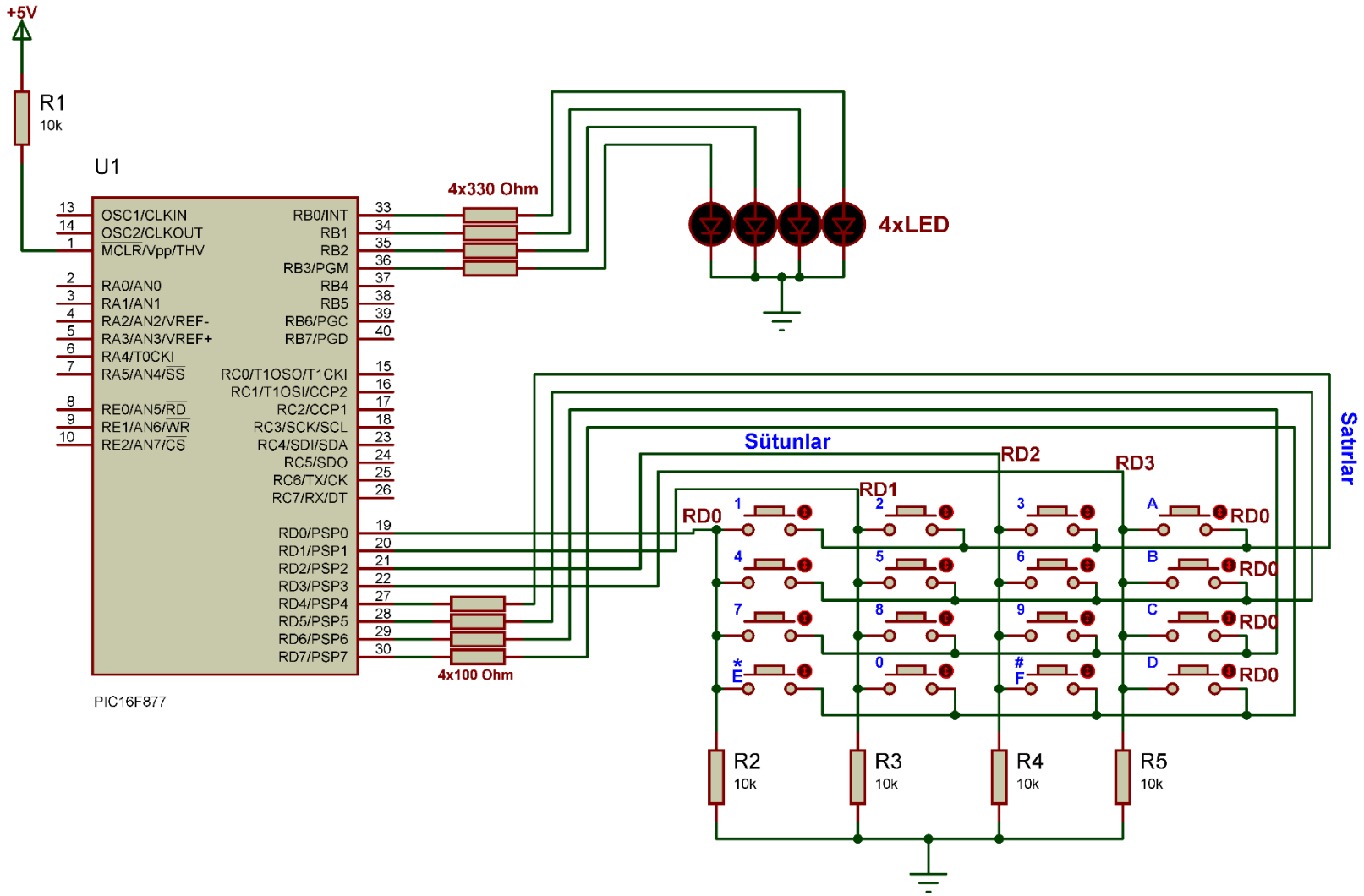
- 74C922 entegresi 3V ile 15V çalışma gerilim aralığında çalışabilir. Genel besleme gerilimleri 5V, 10V ve 15V'tur. Klavye satır hatları Y girişlerine (ROW), sütun hatları ise X girişlerine (COLUMN) bağlanır. OE entegre seçme ucudur ve Lojik-0'da aktiftir. Bu uca 0 verildiğinde entegre çıkışı aktif yapılmış demektir. "Data Available (DA) " ucu ise, klavyede herhangi bir tuşa basıldığında çıkışı lojik-1 olur.
- Böylece klavyeden bir tuşa basıldığını bize entegre haber verir. GE ucu aktif olmasa bile DA ucu herhangi bir tuşa basıldığında aktif olur, fakat GE ucu aktif olana kadar basılan tuşun değeri data çıkış uçlarında (A, B, C, D) görülmez. "Keybounce Mask KBM " ucu ise, tuşa basıldıktan sonra oluşan arkların önlenmesi için ne kadar bir süre beklenmesi gerekeceğini belirleyen bir kontrol ucudur. Bu uca bağlanacak kondansatörün değeri bekleme süresini belirler. Tarama osilatör ucuna (Oscillator, pin 5) bir osilatör bağlanabileceği gibi sadece kondansatörde bağlanabilir. Kondansatör bağlanması durumunda Şekil-6'da görüldüğü gibi KBM ucuna bağlanan kondansatör değeri, OSC ucuna bağlanan kondansatör değerinin 10 katı olmalıdır.

74C922 KEY ENKODER ENTEGRESI

- 74C922 entegresi toplam 16 tuş için 4 bit'lik değerler üretir. Şekil-6. ve Tablo-2. birlikte incelendiğinde hangi satır ve sütundaki (X sütun, Y satir) tuşa basıldığında entegre tarafından üretilen değer (kesişme olan tuş) görülebilir. Program vasıtasıyla üretilen bu değerlere istenen karakter karşılığı verilebilir.

Tuş	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Pozisyonu	Y1,X1	Y1,X2	Y1,X3	Y1,X4	Y2,X1	Y2,X2	Y2,X3	Y2,X4	Y3,X1	Y3,X2	Y3,X3	Y3,X4	Y4,X1	Y4,X2	Y4,X3	Y4,X4
DATA ÇIKIŞI	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Örnek Programlar (Tuş Takımı Uyg.-1)



Örnek Programlar (Tuş Takımı Uyg.-1)

```
#include <16f877.h>
#fuses XT,NOWDT
#use delay (clock=4000000)
#use fast_io(b)
#use fast_io(d)

#byte   portb=0x06   // B portu "tus" ismine eşitleniyor.

#define sut1   pin_d0 // sut1 ifadesi pin_d0 ifadesine eşitleniyor
#define sut2   pin_d1 // sut2 ifadesi pin_d1 ifadesine eşitleniyor
#define sut3   pin_d2 // sut3 ifadesi pin_d2 ifadesine eşitleniyor
#define sut4   pin_d3 // sut3 ifadesi pin_d2 ifadesine eşitleniyor

#define sat1   pin_d4 // sat1 ifadesi pin_d4 ifadesine eşitleniyor
#define sat2   pin_d5 // sat2 ifadesi pin_d5 ifadesine eşitleniyor
#define sat3   pin_d6 // sat3 ifadesi pin_d6 ifadesine eşitleniyor
#define sat4   pin_d7 // sat4 ifadesi pin_d7 ifadesine eşitleniyor

char tus=0; // karakter tipinde değişken tanımlanıyor

```

Örnek Programlar (Tuş Takımı Uyg.-1)

```
21  //*********** Keypad Tarama Fonksiyonu *********
22  char keypad_oku() // Fonksiyon ismi
23  {  output_d(0x00); // D portu çıkışı sıfırlanıyor
24
25     output_high(sat1); // 1. satır lojik-1 yapılıyor
26     if (input(sut1))   // 1. sütun okunuyor
27        { delay_ms(20); tus=1; }
28     if (input(sut2))   // 2. sütun okunuyor
29        { delay_ms(20); tus=2; }
30     if (input(sut3))   // 3. sütun okunuyor
31        { delay_ms(20); tus=3; }
32     if (input(sut4))   // 4. sütun okunuyor
33        { delay_ms(20); tus=0xA; }
34     output_low(sat1); // 1. satır lojik-0 yapılıyor
35
36     output_high(sat2); // 2. satır lojik-1 yapılıyor
37     if (input(sut1))   // 1. sütun okunuyor
38        { delay_ms(20); tus=4; }
39     if (input(sut2))   // 2. sütun okunuyor
40        { delay_ms(20); tus=5; }
41     if (input(sut3))   // 3. sütun okunuyor
42        { delay_ms(20); tus=6; }
43     if (input(sut4))   // 4. sütun okunuyor
```

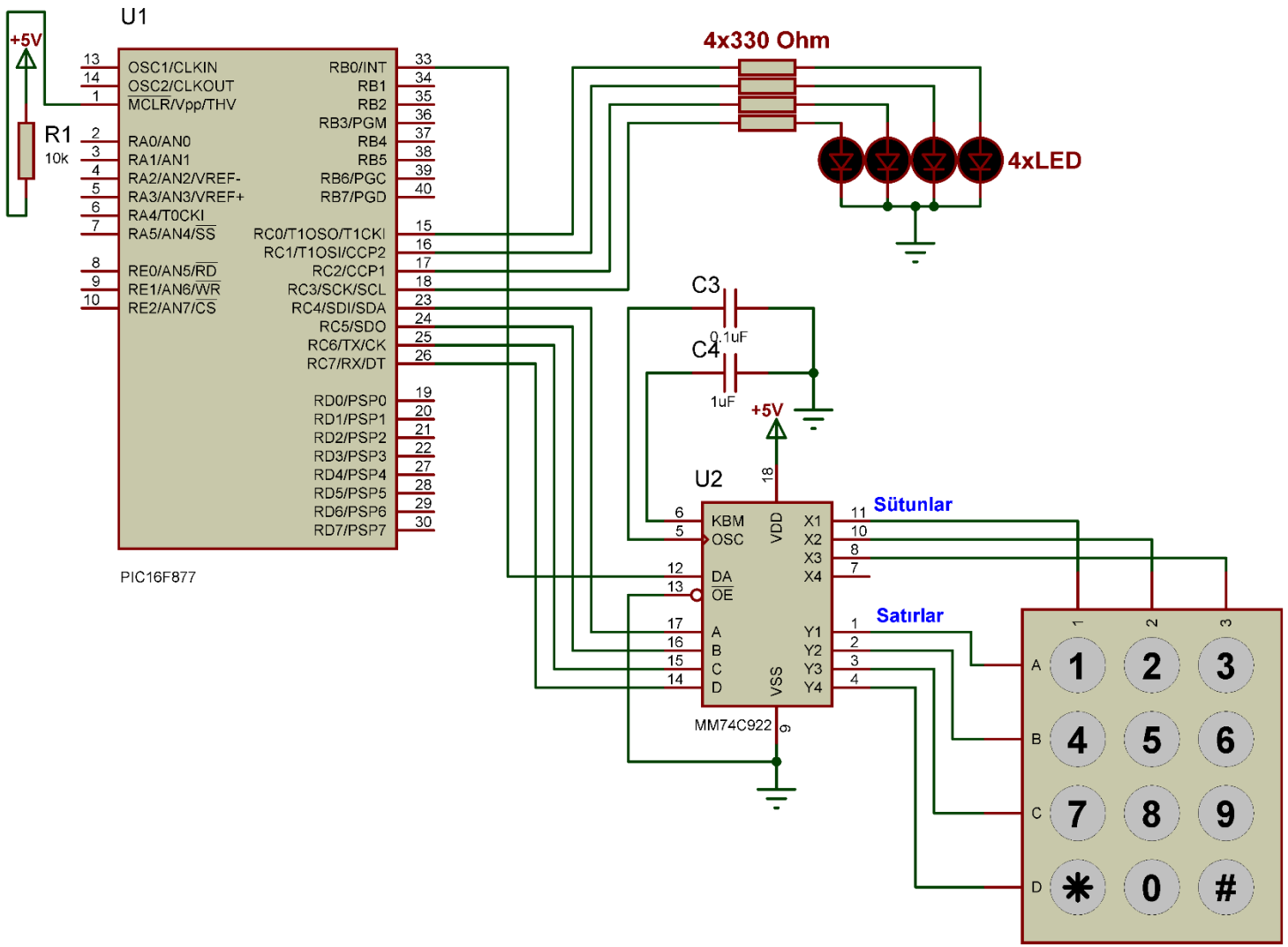
Örnek Programlar (Tuş Takımı Uyg.-1)

```
44      { delay_ms(20); tus=0xB; }
45    output_low(sat2); // 2. satır lojik-0 yapılıyor
46
47    output_high(sat3); // 3. satır lojik-1 yapılıyor
48    if (input(sut1))   // 1. sütun okunuyor
49       { delay_ms(20); tus=7; }
50    if (input(sut2))   // 2. sütun okunuyor
51       { delay_ms(20); tus=8; }
52    if (input(sut3))   // 3. sütun okunuyor
53       { delay_ms(20); tus=9; }
54    if (input(sut4))   // 4. sütun okunuyor
55       {delay_ms(20); tus=0x0C; }
56    output_low(sat3); // 3. satır lojik-0 yapılıyor
57
58    output_high(sat4); // 3. satır lojik-1 yapılıyor
59    if (input(sut1))   // 1. sütun okunuyor
60       { delay_ms(20); tus=0xE; }
61    if (input(sut2))   // 2. sütun okunuyor
62       { delay_ms(20); tus=0; }
63    if (input(sut3))   // 3. sütun okunuyor
64       { delay_ms(20); tus=0xF; }
65    if (input(sut4))   // 4. sütun okunuyor
```

Örnek Programlar (Tuş Takımı Uyg.-1)

```
63      if (input(sut3))    // 3. sütun okunuyor
64         { delay_ms(20); tus=0xF; }
65      if (input(sut4))    // 4. sütun okunuyor
66         {delay_ms(20); tus=0xD; }
67      output_low(sat4); // 3. satır lojik-0 yapılıyor
68
69      return tus; // Fonksiyon "tus" değeri ile geri döner
70   }
71
72   void main ( )
73   {  set_tris_b(0x00);    // B portu komple çıkış
74      set_tris_d(0x0F);    // Yüksek değerlikli 4 bit çıkış, düşük değer
75
76      output_b(0x00); // İlk anda B portu çıkışı sıfırlanıyor
77
78      while(1)  // Sonsuz döngü
79      {
80          portb=keypad_oku();  // Basılan tuş değerini B portuna aktar
81      }
82   }
```

Örnek Programlar (Tuş Takımı Uyg.-2)



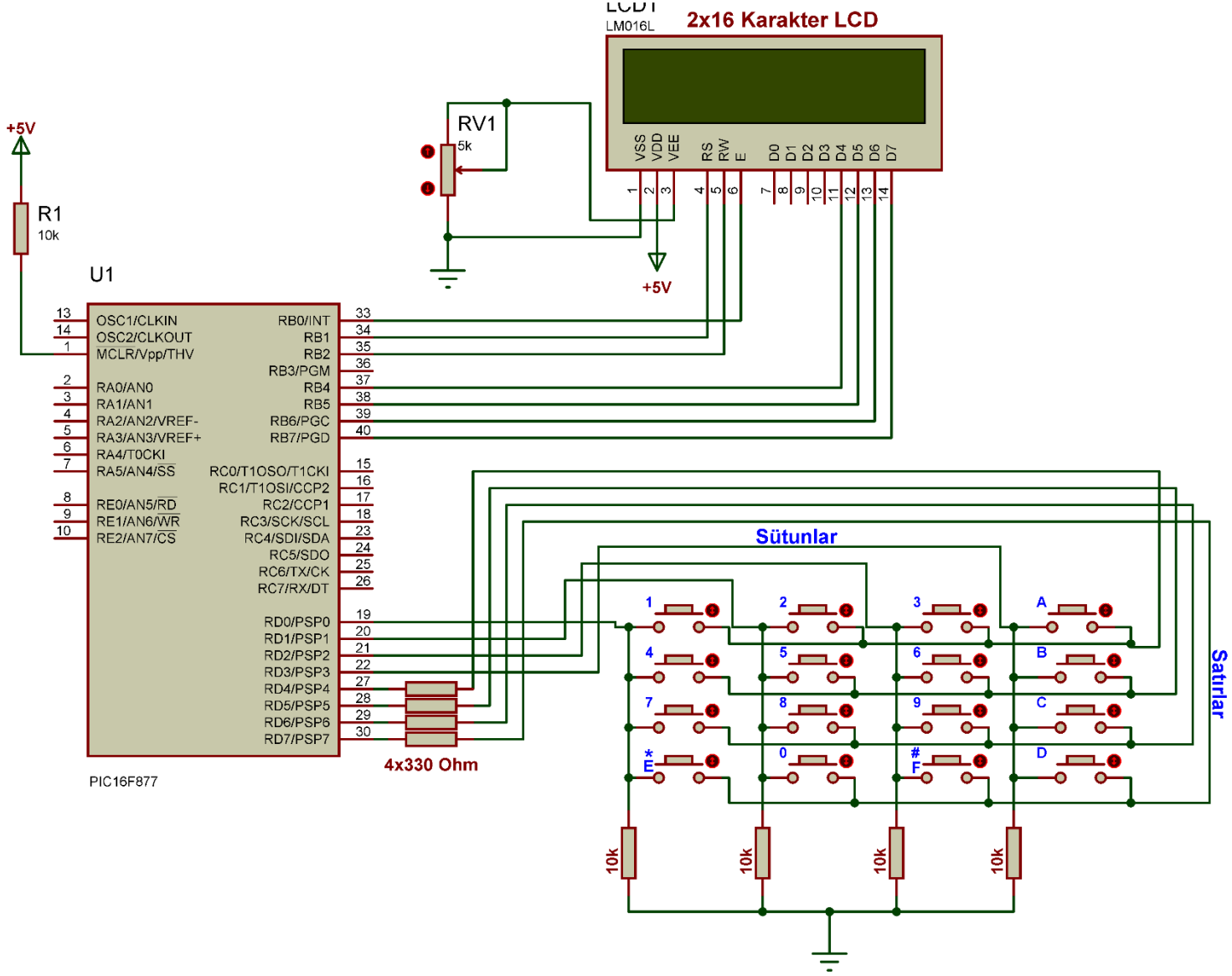
Örnek Programlar (Tuş Takımı Uyg.-2)

```
#include <16f877.h>
#fuses XT,NOWDT
#use delay (clock=4000000)
#use fast_io(b)
#use fast_io(c)
#byte   portc=0x07   // C portu "portc" ismine eşitleniyor.

char const keys[] ={1,2,3,0,   // Gelen bilgilere göre gösterilmesi
                    4,5,6,0,    // istenen karakterler
                    7,8,9,0,
                    0xF,0,0xF,0 };
//************ Dış Kesme Fonksiyonu Tanımlanıyor ********
#int_ext            // Dış kesme fonksiyonu
void dis_kesme ()
{
   portc=keys[portc>>4];  // C portuna C portunun okunan yüksek değe
}

void main ( )
{
   setup_psp(PSP_DISABLED);        // PSP birimi devre dışı
   setup_timer_1(T1_DISABLED);     // T1 zamanlayıcısı devre dışı
```

Örnek Programlar (Tuş Takımı Uyg.-2)

```
19  void main ( )
20  {
21     setup_psp(PSP_DISABLED);         // PSP birimi devre dışı
22     setup_timer_1(T1_DISABLED);      // T1 zamanlayıcısı devre dışı
23     setup_timer_2(T2_DISABLED,0,1); // T2 zamanlayıcısı devre dışı
24     setup_adc_ports(NO_ANALOGS);     // ANALOG giriş yok
25     setup_adc(ADC_OFF);              // ADC birimi devre dışı
26
27     set_tris_b(0x01);    // B portu komple çıkış
28     set_tris_c(0xF0);    // C portunun Yüksek değerlikli 4 bit çıkış,
29
30     ext_int_edge(L_TO_H);  // INT_EXT dış kesmesinin yükselen kenarda
31
32     enable_interrupts(INT_EXT);  // Dış kesme aktif
33     enable_interrupts(GLOBAL);   // Aktif edilen kesmelere izin ver
34
35     portc=0;  // Başlangıçta C portu çıkışı sıfırlanıyor
36
37     while(1);  // Sonsuz döngü
38  }
```

Örnek Programlar (Tuş Takımı Uyg.-3)



Örnek Programlar (Tuş Takımı Uyg.-3)

```
1  #include <16f877.h>
2  #fuses XT,NOWDT
3  #use delay (clock=4000000)
4  #use fast_io(b)
5  #use fast_io(d)
6  #define use_portb_lcd TRUE
7  #include <lcd.c>
8  #define sut1   pin_d0 // sut1 ifadesi pin_d0 ifadesine eşitleniyor
9  #define sut2   pin_d1 // sut2 ifadesi pin_d1 ifadesine eşitleniyor
10 #define sut3   pin_d2 // sut3 ifadesi pin_d2 ifadesine eşitleniyor
11 #define sut4   pin_d3 // sut3 ifadesi pin_d2 ifadesine eşitleniyor
12
13 #define sat1   pin_d4 // sat1 ifadesi pin_d4 ifadesine eşitleniyor
14 #define sat2   pin_d5 // sat2 ifadesi pin_d5 ifadesine eşitleniyor
15 #define sat3   pin_d6 // sat3 ifadesi pin_d6 ifadesine eşitleniyor
16 #define sat4   pin_d7 // sat4 ifadesi pin_d7 ifadesine eşitleniyor
17
18  char tus=0; // karakter tipinde değişken tanımlanıyor
19
```

Örnek Programlar (Tuş Takımı Uyg.-3)

```
22  //*********** Keypad Tarama Fonksiyonu *********
23  char keypad_oku() // Fonksiyon ismi
24  {  output_d(0x00); // D portu çıkışı sıfırlanıyor
25
26     output_high(sat1); // 1. satır lojik-1 yapılıyor
27     if (input(sut1))   // 1. sütun okunuyor
28        { delay_ms(20); tus=1; }
29     if (input(sut2))   // 2. sütun okunuyor
30        { delay_ms(20); tus=2; }
31     if (input(sut3))   // 3. sütun okunuyor
32        { delay_ms(20); tus=3; }
33     if (input(sut4))   // 4. sütun okunuyor
34        { delay_ms(20); tus=0xA; }
35     output_low(sat1); // 1. satır lojik-0 yapılıyor
36
37     output_high(sat2); // 2. satır lojik-1 yapılıyor
38     if (input(sut1))   // 1. sütun okunuyor
39        { delay_ms(20); tus=4; }
40     if (input(sut2))   // 2. sütun okunuyor
41        { delay_ms(20); tus=5; }
42     if (input(sut3))   // 3. sütun okunuyor
43        { delay_ms(20); tus=6; }
44     if (input(sut4))   // 4. sütun okunuyor
```

Örnek Programlar (Tuş Takımı Uyg.-3)

```
44      if (input(sut4))   // 4. sütun okunuyor
45         { delay_ms(20); tus=0xB; }
46      output_low(sat2); // 2. satır lojik-0 yapılıyor
47
48      output_high(sat3); // 3. satır lojik-1 yapılıyor
49      if (input(sut1))   // 1. sütun okunuyor
50         { delay_ms(20); tus=7; }
51      if (input(sut2))   // 2. sütun okunuyor
52         { delay_ms(20); tus=8; }
53      if (input(sut3))   // 3. sütun okunuyor
54         { delay_ms(20); tus=9; }
55      if (input(sut4))   // 4. sütun okunuyor
56         {delay_ms(20); tus=0x0C; }
57      output_low(sat3); // 3. satır lojik-0 yapılıyor
58
59      output_high(sat4); // 3. satır lojik-1 yapılıyor
60      if (input(sut1))   // 1. sütun okunuyor
61         { delay_ms(20); tus=0xE; }
62      if (input(sut2))   // 2. sütun okunuyor
63         { delay_ms(20); tus=0; }
64      if (input(sut3))   // 3. sütun okunuyor
65         { delay_ms(20); tus=0xF; }
66      if (input(sut4))   // 4. sütun okunuyor
```

Örnek Programlar (Tuş Takımı Uyg.-3)

```
65        { delay_ms(20); tus=0xF; }
66     if (input(sut4))    // 4. sütun okunuyor
67        {delay_ms(20); tus=0xD; }
68     output_low(sat4); // 3. satır lojik-0 yapılıyor
69
70     return tus; // Fonksiyon "tus" değeri ile geri döner
71  }
72
73  void main ( )
74  {  set_tris_b(0x00);    // B portu komple çıkış
75     set_tris_d(0x0F);    // Yüksek değerlikli 4 bit çıkış, düşük değerlikli 4 bit giriş
76
77     lcd_init();    // LCD hazırlanıyor
78     printf(lcd_putc,"\fBasilan Tus="); // LCD'ye string yazdırılıyor
79
80     while(1) // Sonsuz döngü
81     { lcd_gotoxy(13,1); // İmleç 3.sütun, 1.satıra konumlandırılıyor
82        if (keypad_oku()>9)    // Eğer basılan tuş değeri 9'dan büyük ise
83           printf(lcd_putc,"%d",keypad_oku());  // Tuş değeri LCD'ye yazdırılıyor
84        else                   // Eğer basılan tuş değeri 9'dan büyük değilse
85           printf(lcd_putc,"%d ",keypad_oku()); // Tuş değeri LCD'ye yazdırılıyor
86     }
87  }
```

GRAFİK LCD HAKKINDA BİLGİ

- Grafik LCD'ler karakter tabanlı LCD'lerin tüm özelliklerini yerine getirebilirler. Bunun yaninda grafik LCD'lerde piksel düzeyinde kontrol imkanı oldugundan, grafik LCD'ler istenen çizimlerde gösterilebilir. Grafik LCD'lerde karakter tabanlı LCD'lerde olduğu gibi kontrol entegreleri içerirler. Grafik LCD'lerin kontrol edilmesi işlemi daha zordur.

GRAFİK LCD HAKKINDA BİLGİ

Grafik LCD'lerin farklı kontrol entegreleri mevcuttur. Piyasada K0108 entegreleri ile uyumlu birçok grafik LCD rahatlıkla bulunabilir. Tablo-3 de K0108 entegreli HDM64GS12 grafik LCD'nin pin isimleri ve açıklamaları verilmiştir. Verilen pin numaraları çeşitli grafik LCD modellerine göre değişiklik gösterebilir. Bu nedenle doğru pin bağlantısı için mutlaka kullandığınıza grafik LCD'nin teknik dökümanına bakmanız gereklidir.

Pin No	Pin İsmi	Açıklama
1	VSS (GND)	Şase
2	VDD (Vcc)	+5V
3	Vo	Parlaklık Ayarı
4	D/I	Data veya Komut Giriş Seçme Ucu
5	R/W	Data Okuma/Yazma
6	E	Yetki verme ucu
7	DB0	Veri Uçlarıdır. Bu uçlardan hem veri hem de komut kodu gönderilir.
8	DB1	
9	DB2	
10	DB3	
11	DB4	
12	DB5	
13	DB6	
14	DB7	
15	CS1	IC1 seçme ucu
16	CS2	IC2 seçme ucu
17	RES (RST)	Reset ucu
18	VEE (-Vout)	Negatif Voltaj Çıkışı
19	K	Arka ışık GND ucu
20	A	Arka ışık +5V ucu

CCS C Grafik LCD Kütüphane Dosyaları

- CCS C programı içinde grafik LCD ile ilgili 4 adet kütüphane dosyası bulunmaktadır. Bu dosyalar c:\Program Files\PICC\Drivers\ klasörü içinde bulunmaktadır. Grafik LCD ile ilgili CCS C dosyaları şunlardır;

• GLCO.c • HDM64GS12.c • GRAPHICS.c • KS0108.c

- GLCD.c dosyası, Hantronix HDM64GS12 grafik LCD ile ilgili kontrol fonksiyonlarını içerir. 128x64 Grafik LCD'ler için uyumludur.
- HDM64GS12.c dosyası, KS0108 LCD kontrolcüsüne sahip Hantronix HOM64GS12 grafik LCD ile ilgili kontrol fonksiyonları içerir. 128x64 Grafik LCD'ler için uyumludur.
- GRAPHICS.c dosyası, grafik LCD'lerde çizgi, dikdörtgen, daire, bar çubuk ve karakter gösterimleri ile ilgili fonksiyonları içerir.

CCS C Grafik LCD Kütüphane Dosyaları

- KS0108.c dosyası, KS0108 LCD kontrolcüsü ile ilgili kontrol fonksiyonlarını içerir. Bahsedilen dosyaların içinde geçen fonksiyonların bir çoğu aynı isimdedir ve aynı işlemleri yaparlar. İstenirse CCS C için yazılmış başka grafik LCD fonksiyon dosyaları CCS C forum sitesinden ve diğer internet kaynaklarından da bulunabilir. Aşağıda GLCD.c, HDM64GS12.c ve GRAPHICS.c dosyalarının içerdiği fonksiyonlar ve kullanımları verilmiştir. Dosyalar içinde bulunan aynı isimli fonksiyonlar aynı görevi yerine getirirler.
- GLCD.c ve HDM64GS12.c dosyalarında grafik LCD bağlantısı aşağıda verilmiştir.
- Grafik LCD bağlantı pin'leri şunlardır. D port'una bağlı olan data hatları hariç, diğer uçlar bu dosya içinde değişiklik yapılarak farklı pin'ler kullanılabilir.

RB0	CS1
RB1	CS2
RB2	DI
RB4	R/W
RB5	E
RC0	RST

RD0	DB0
RD1	DB1
RD2	DB2
RD3	DB3
RD4	DB4
RD5	DB5
RD6	DB6
RD7	RD7

GLCD.c Dosyası Fonksiyonları

- **glcd_init (mode)** = Diğer tüm grafik LCD komutları kullanılmadan önce tanımlanması gerekir. Fonksiyondaki "mode" kısmına "ON" veya "OFF" sabitleri yazılabilir. "ON" LCD'yi açar, OFF kapatır.
- **glcd_pixel (x,y,color)** = Belirtilen x ve y koordinatlarındaki pikseli aktif veya pasif etmeye yarar. "x" ve "y" değişkenleri koordinatları belirtir. "color" değişkeni ise ON veya OFF değerlerini alarak belirtilen yerdeki pikselin siyah veya beyaz olmasını belirler. ON siyah, OFF beyaz görevini görür.
- **glcd_line (x1, y1, x2, y2, color)** = Verilen iki nokta arasında çizgi çizer. Fonksiyondaki x1, y1 değişkenleri ilk noktanın konumunu x2,y2 değişkenleri ikinci noktanın konumunu belirtir. "color" değişkeni "ON" veya "OFF" değerlerini alabilir. "ON" değişkeni ile çizgi siyah, "OFF" değişkeni ile çizgi beyaz olur.

GLCD.c Dosyası Fonksiyonları

- **glcd_rect (x1, y1, x2, y2, fill, color)** = Belirlenen 2 köşe noktası arasında dikdörtgen çizme işlemini yapar. Fonksiyondaki x1, y1 değişkenleri ilk köşenin konumunu x2,y2 değişkenleri ikinci köşenin konumunu belirtir. " fill " değişkeni "YES" ve "NO" sabitlerini alabilir. "YES" sabiti ile çizilen dikdörtgenin içi boyanır, "NO" sabiti ile çizilen dikdörtgenin içi boyanmaz. "color" değişkeni "ON" veya "OFF " değerlerini alabilir. "ON" değişkeni ile dikdörtgen siyah, "OFF" değişkeni ile dikdörtgen beyaz olur.
- **glcd_bar (x1, y1, x2, y2, width, color)** = Belirlenen iki nokta arasında çubuk (bar) çizer. Fonksiyondaki x1, y1 değişkenleri ilk noktanın konumunu x2,y2 değişkenleri ikinci noktanın konumunu belirtir. Fonksiyondaki "width " değişkeni çizdirilen çubuğun piksel olarak genişliğini belirtir. " color " değişkeni "ON " veya "OFF " değerlerini alabilir. "ON" değişkeni ile çubuk siyah, "OFF" değişkeni ile beyaz olur.

GLCD.c Dosyası Fonksiyonları

- **glcd_circle (x, y, radius, fill, color)** = Belirlenen merkez noktası etrafında belirlenen yarıçapta daire çizer. Fonksiyondaki x ve y değişkenleri daire merkezini, ''radius'' değişkeni dairenin yarıçapını belirtir. " fill " değişkeni "YES " ve "NO " sabitlerini alabilir. "YES" sabiti ile çizilen dairenin içi boyanır, "NO" sabiti ile çizilen dairenin içi boyanmaz. "color " değişkeni "ON " veya "OFF" değerlerini alabilir. "ON " değişkeni ile daire siyah, "OFF" değişkeni ile daire beyaz olur.
- **glcd_text57(x, y, textptr, size, color)** = Bu fonksiyon ile x ve y koordinatları ile belirlenen noktadan başlanarak, "textptr " string dizisindeki karakterleri, "size " değişkeni ile belirtilen piksel büyüklüğünde display'de yazmaya yarar. Ekrana yazdırılan karakterler 5 piksel genişliğinde, 7 piksel yüksekliğindedir. "color " değişkeni "ON " veya "OFF " değerlerini alabilir. "ON" değişkeni ile yazı siyah, "OFF " değişkeni ile yazı beyaz olur. Bu fonksiyon ile ekrana yazdırılan yazılar ekran sınırını aşarsa otomatik olarak bir alt satıra geçilir.
- **glcd_fillScreen(color)** = Tüm LCD ekranını karartır veya karartılmış ekranı normal haline geri döndürür. "color " değişkeni "ON " veya "OFF" değerlerini alabilir. "ON" değeri ile ekran tümüyle karartılır. "OFF" değeri ile karartılmış ekran eski haline getirilir.

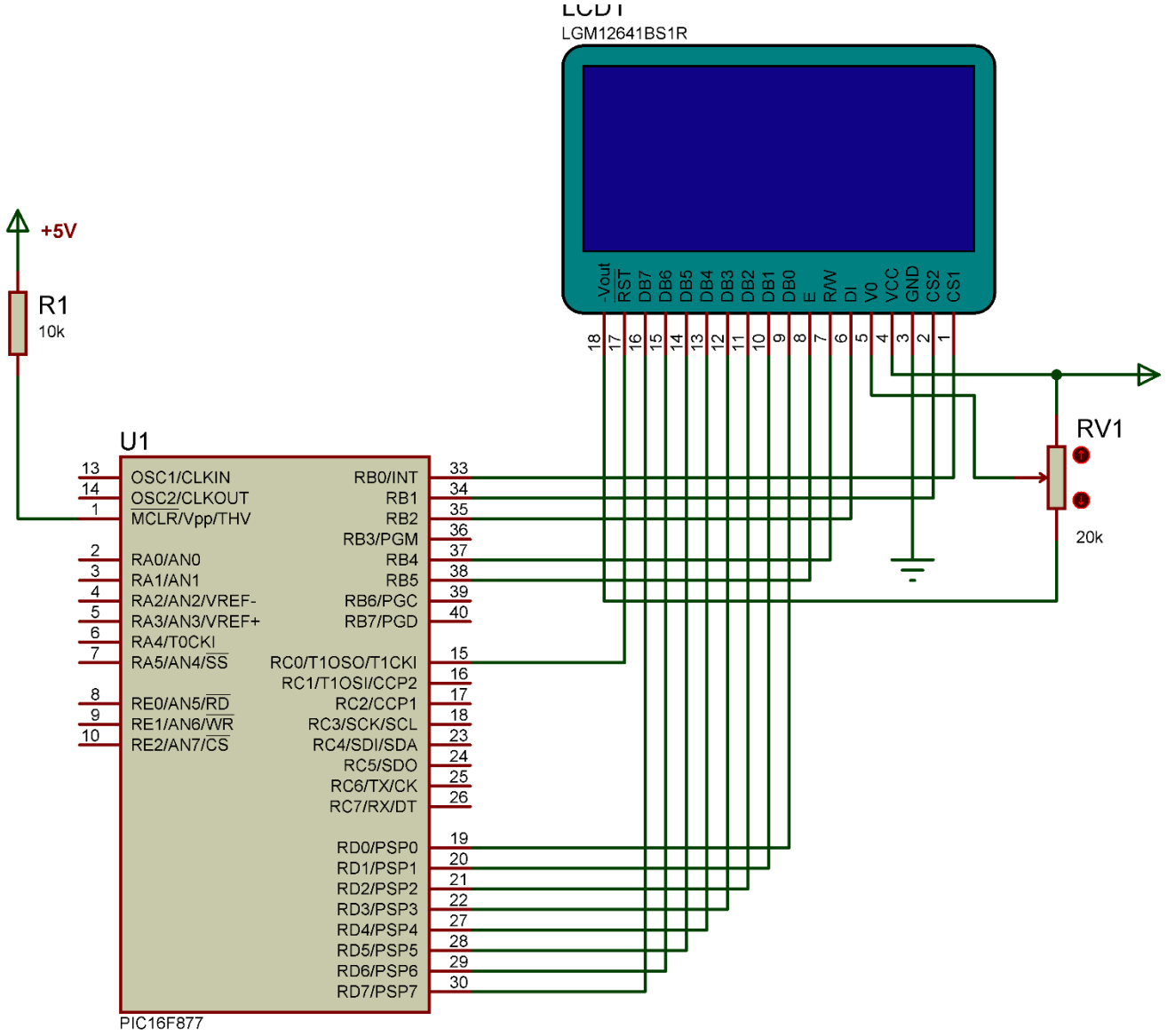
HDM64GS12.e Dosyası Fonksiyonları

- **#define FAST_GLCD** = Eğer kullanılan PIC denetleyici RAM hafızası fazla olan denetleyiciler de kullanılabilir (18F serisi için uygundur). Bu komutun program başında HDM64GS12.c dosyası tanıtılmadan önce kullanılması gerekir. Bu komut ile hızlı bir LCD display'inin tazelenmesi işlemi için RAM hafızası yoğunlukla kullanılır. glcd_update ( ) = RAM bellekte saklanmış display bilgilerini LCD'ye yazar. Bu komut sadece #define fast_GLCD komutu kullanılmış ise kullanılır. glcd_init(mode), glcd_pixel(x,y,color), glcd_fillScreen(color) fonksiyonları da bu dosya içindedir. Bu fonksiyonların görevleri GLCD.c dosyası tanıtılırken belirtilmişti.

GRAPHICS.c Dosyası Fonksiyonları

- GRAPHICS,c dosyası şu fonksiyonları içerir. glcd_line(x1, y1, x2, y2, color), glcd_rect(x1, y1, x2, y2, fill, color), glcd_bar(x1, y1, x2, y2, width, color), glcd_circle(x, y, radius, fill, color), glcd_text57(x, y, textptr, size, color). Bu fonksiyonlar daha önce GLCD.c dosyası tanıtılırken belirtilmişti. GRAPHICS.c dosyası geometrik şekillerin çizilmesi ile ilgili ayrı bir dosyadır. Bu dosya kullanılan bir grafik LCD kontrol dosyası ile birlikte kullanılabilir, Böylece geometrik şekiller için ayrıca fonksiyon yazmaya gerek kalmaz.

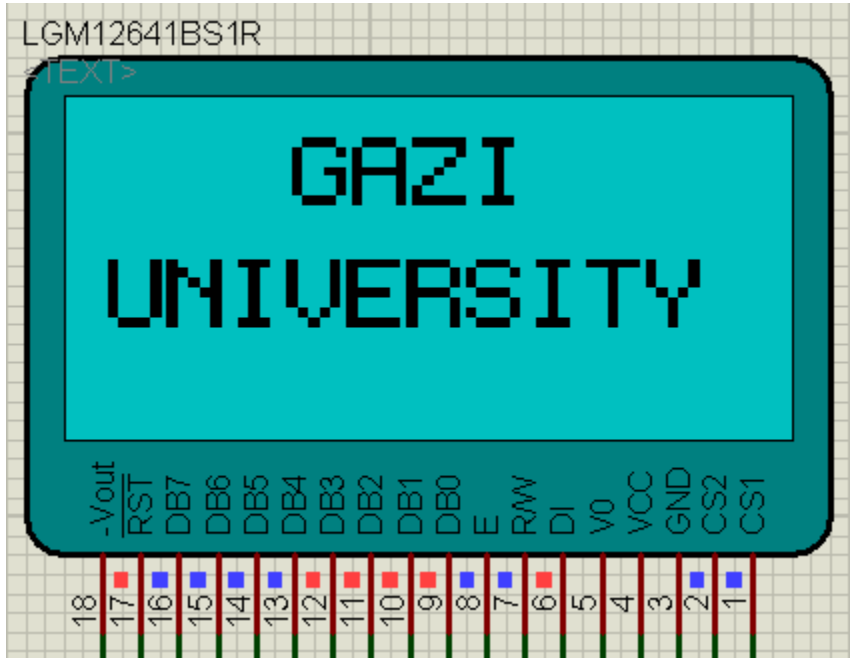
Örnek Programlar (GLCD Uygulaması)



Örnek Programlar (GLCD Uygulaması)

```
1  #include <16f877.h>
2  #fuses HS,NOWDT
3  #use delay (clock=20000000)
4  #include <HDM64GS12.c> // HDM64GS12.c dosyası programa ekleniyor
5  #include <graphics.c>  // graphics.c dosyası programa ekleniyor
6
7  char yazi1[]="GAZI";         // Karakter dizisi tanımlanıyor
8  char yazi2[]="UNIVERSITY";   // Karakter dizisi tanımlanıyor
9  int i,x1,y1,x2,y2;
10
11 void main ( )
12 {
13    glcd_init(ON); // Grafik LCD hazırlanıyor ve ekran siliniyor
14
15    while(1)
16    {  glcd_init(ON); // Ekran siliniyor
17       glcd_text57(39, 5, yazi1, 2, ON);  // GLCD'de yazı yazdırılıyor
18       glcd_text57(5, 30, yazi2, 2, ON);  // GLCD'de yazı yazdırılıyor
19       delay_ms(2000);
20
```

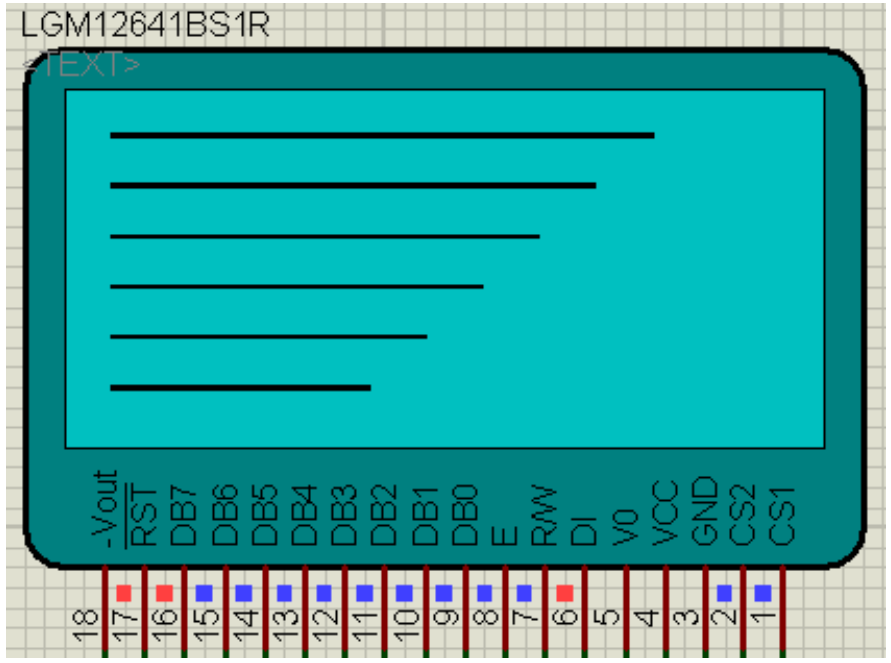
Örnek Programlar (GLCD Uygulaması)

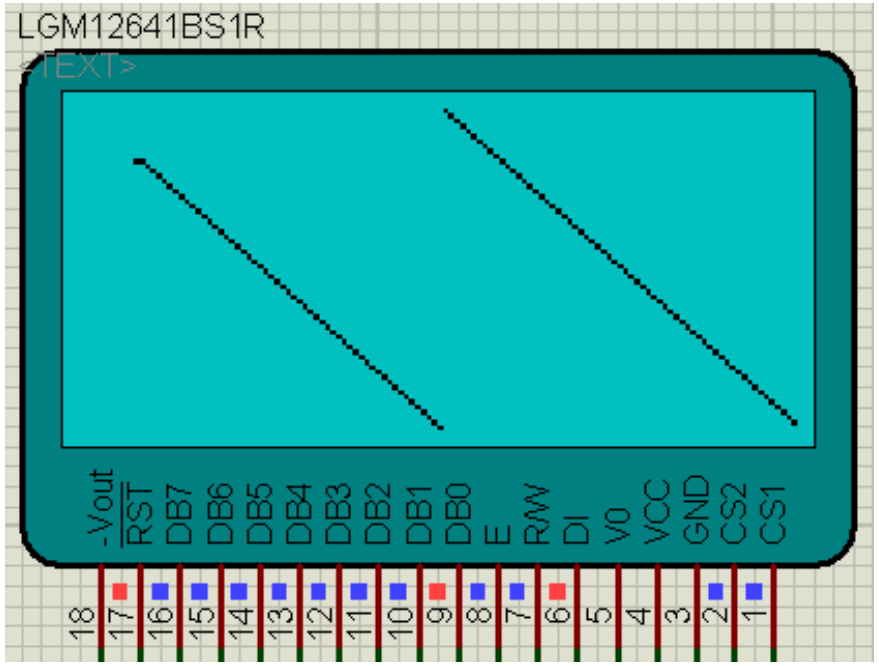


Örnek Programlar (GLCD Uygulaması)

```
21      glcd_init(ON); // Ekran siliniyor
22      x1=5;
23      y1=5;
24      x2=100;
25      y2=5;
26      for(i=0;i<6;i++)
27      {
28         glcd_line(x1, y1, x2, y2, ON); // GLCD'de Çizgi çizdiriliyor
29         y2=y1+=10; // y2=y1=y2+10; anlamında
30         x2-=10;    // x2=x2-10; anlamında
31         delay_ms(1000);
32      }
33
34      glcd_init(ON); // Ekran siliniyor
35      x1=y1=10;
36      for (i=0;i<120;i++)
37      {
38         glcd_pixel(x1,y1,ON); // Ekranda istenen pikseller aktif yap
39         y1=x1++;   // y1=x1+1; anlamında
40         delay_ms(50);
41      }
42
```

Örnek Programlar (GLCD Uygulaması)

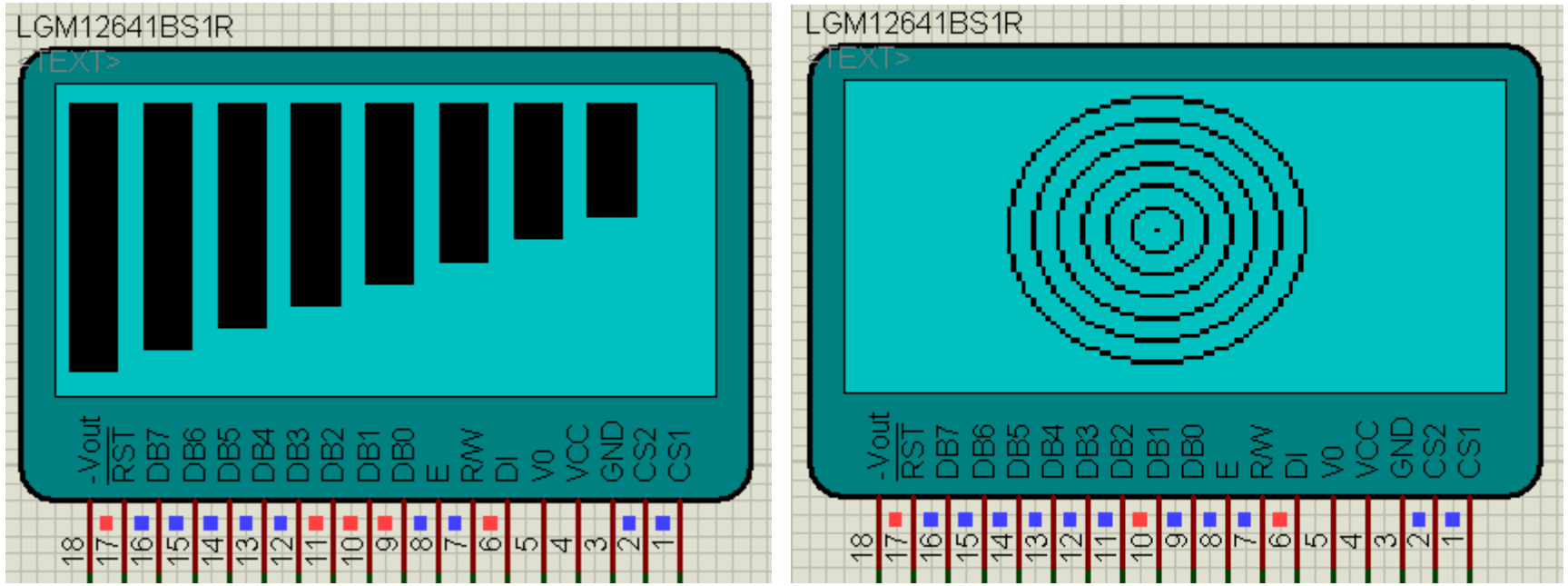




Örnek Programlar (GLCD Uygulaması)

```
43      glcd_init(ON); // Ekran siliniyor
44      y2=60;
45      x1=5;
46      for (i=0;i<8;i++)
47      {
48         glcd_bar(x1, 0, x1, y2, 10, ON); // GLCD'de çubuk çizdiriliyor
49         delay_ms(1000);
50         x1+=15; // x1=x1+15; anlamında
51         y2-=5;  // y2=y2-5; anlamında
52      }
53
54      glcd_init(ON); // Ekran siliniyor
55      for(i=0;i<=30;i=i+5)
56      {
57         glcd_circle(60, 30, i, NO, ON); // GLCD'de Daire çizdiriliyor
58         delay_ms(1000);
59      }
```

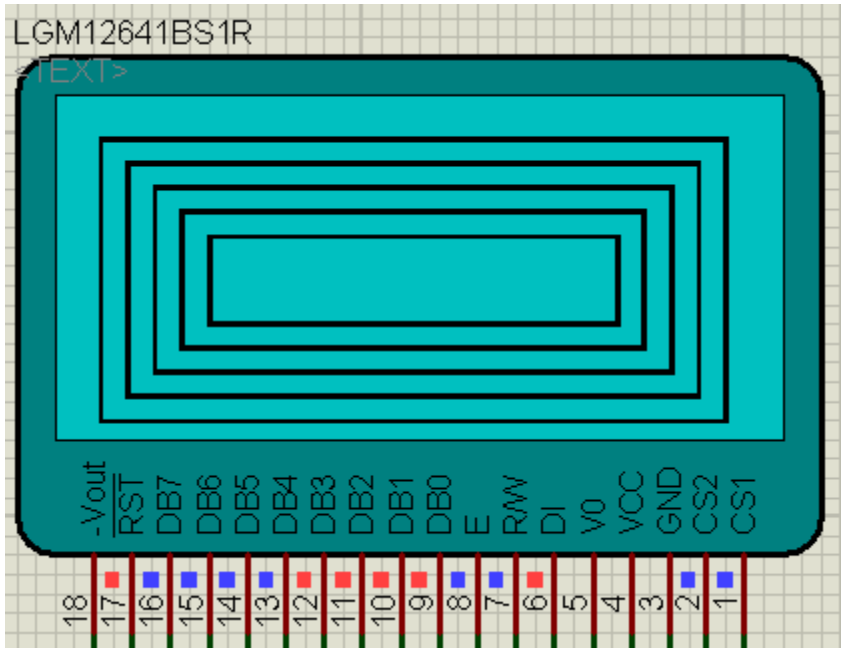
Örnek Programlar (GLCD Uygulaması)



Örnek Programlar (GLCD Uygulaması)

```
60
61          glcd_init(ON); // Ekran siliniyor
62          x1=5;
63          y1=5;
64          x2=120;
65          y2=63;
66          for(i=0;i<6;i++)
67          {
68              glcd_rect(x1, y1, x2, y2, NO, ON); // GLCD'de dikdötrgen çizdiriliyor
69              y1=x1+=5; // y1=x1+5; anlamında
70              x2-=5;    // x2=x2-5; anlamında
71              y2-=5;    // y2=y2-5; anlamında
72              delay_ms(500);
73          }
74      }
75  }
```

Örnek Programlar (GLCD Uygulaması)



Kaynaklar

- CCS C Programlama Kitabı, Serdar Çiçek, Altaş Yayıncılık
- Mikroelektronika C programlama e-kitabı «https://www.mikroe.com/ebooks/pic-microcontrollers-programming-in-c»