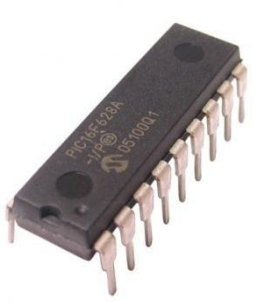
Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

BİLİŞİM TEKNOLOJİLERİ

ALANI

BİLGİSAYAR

TEKNİK SERVİS DALI



MİKRODENETLEYİCİLER

DERSİ

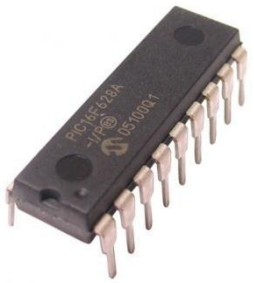
DERS NOTLARI

2013

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

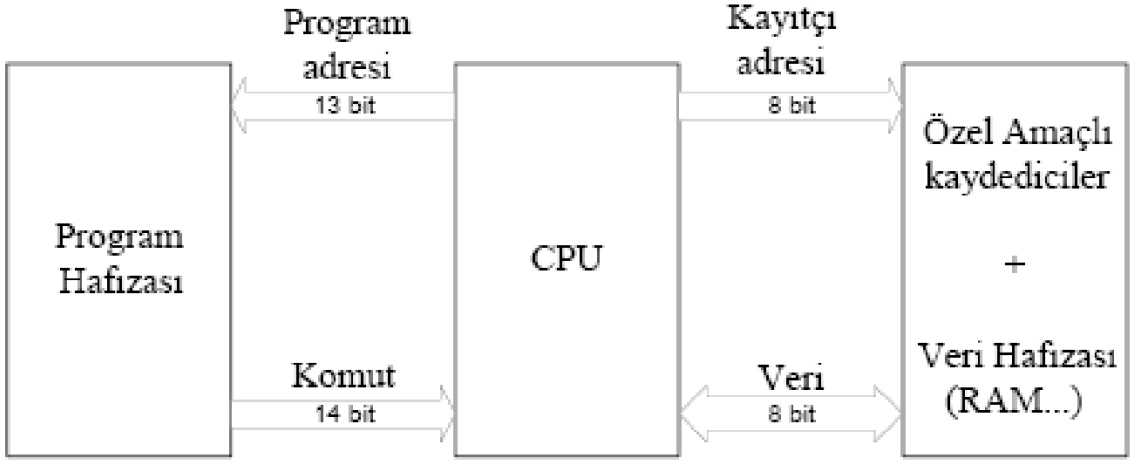
Günümüzde elektronik teknolojideki gelişmeler bilgisayar ve elektronik cihazların gelişmesine büyük katkı sağlamıştır. Bu gelişmelerin paralelinde mikrodenetleyiciler de gelişmiştir. Mikrodenetleyicilerin bu gelişmesi birçok endüstriyel amaçlar için kullanılmasına yol açmıştır. Hayatımızın hemen hemen her kesiminde yaygın olarak kullanılmaktadır. Bunlar çevremizdeki bir değişim ölçülmesi ya da bir cihazın otomatik kontrolü olarak sıkça karşımıza çıkmaktadır



1.1. PIC 16F877 Mikrodenetleyicisi

Mikrodenetleyicilerin kullanımı yaygınlaştıkça Atmel, Philips, Renasas, NEC, Microchip gibi firmalar mikrodenetleyici üretmeye başlamışlardır. Bu firmalardan Microchip, 1990 yılında itibaren 8-bit'lik mimari üzerine yaptığı özel donanım eklentileri ile günümüzde onlarca çeşit mikro denetleyici üretmektedir.8 bit mikrodenetleyiciler 8-bit veri yolu, 16-bitlik mikrodenetleyiciler ise 16-bitlik veri yolunu kullanılırlar [1].

PIC 16F87x serisi PIC 16CXX ailesinin özelliklerini taşır. PIC-16CXX de Harvardmimarisi kullanılmıştır. Veri yolu 8 bit genişliğindedir. Program belleğine program yolu yada adres yolu(program bus /addressbus) denilen 13 bit genişliğindeki diğer bir yolla erişilir. PIC 16C87X de komut kodları(opcode), 14 bittir. 14 bitlik program beleğinde her bir adresi, bir komut koduna karşılık gelir[1].Her komuta bir çevrim süresinde (saykıl, cycle) erişilir ve komut yazmacına yüklenir. Dallanma komutları dışındaki bütün komutlar, aynı çevrim süresinde çalıştırılırlar [1].



Şekil : PIC16F877 nin Harvard Mimarisi

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Komut seti: genel olarak picmcu için 35 komut bulunmaktadır.

2013

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Komut Yazılımı		Komut Tanımlaması	Çevrim Süresi	14-bitlik Opcode		STATUS Etkisi
				MSb	LSb	
BYTE Yönlendirmeli Komutlar						
ADDWF	f, d	W ile f 'yi topla	1	00 0111	d f f f f f f f	C,DC,Z
ANDWF	f, d	W ile f 'yi AND 'le	1	00 0101	d f f f f f f f	Z
CLRF	f	f 'yi sil	1	00 0001	l f f f f f f f	Z
CLRW	--	W 'yı sil	1	00 0001	0xxx xxxx	Z
COMF	f, d	f 'nin tersini al	1	00 1001	d f f f f f f f	Z
DECF	f, d	f 'yi bir azalt	1	00 0011	d f f f f f f f	Z
DECFSZ	f, d	f 'yi bir azalt, f – 0 ise bir komut atla	1 (2)	00 1011	d f f f f f f f	
INCF	f, d	f 'yı bir arttır	1	00 1010	d f f f f f f f	Z
INCFSZ	f, d	f 'yı bir arttır, f = 0 ise bir komut atla	1 (2)	00 1111	d f f f f f f f	
IORWF	f, d	W ile f 'yi XOR 'la	1	00 0100	d f f f f f f f	Z
MOVF	f, d	f 'yi taşı	1	00 1000	d f f f f f f f	Z
MOVWF	f	W 'yı f 'ye taşı (W → f)	1	00 0000	l f f f f f f f	
NOP		İşlem yapma	1	00 0000	0xx0 0000	
RLF	f, d	f 'yi birer bit sola döndür	1	00 1101	d f f f f f f f	C
RRF	f, d	f 'yı birer bit sağa döndür	1	00 1100	d f f f f f f f	C
SUBWF	f, d	f 'den W 'yı çıkart	1	00 0010	d f f f f f f f	C,DC,Z
SWAPF	f, d	f 'nin dörtlü bitlerinin yerini değiştir	1	00 1110	d f f f f f f f	
XORWF	f, d	W ile f 'yi XOR 'la	1	00 0110	d f f f f f f f	Z
BIT Yönlendirmeli Komutlar						
BCF	f, b	f 'nin b. bitini sil	1	01 00bb	b f f f f f f f	
BSF	f, b	f 'nin b. bitini bir yap	1	01 01bb	b f f f f f f f	
BTFSC	f, b	f 'nin b. biti "0" ise bir komut atla	1 (2)	01 10bb	b f f f f f f f	
BTFSS	f, b	f 'nin b. biti "1" ise bir komut atla	1 (2)	01 11bb	b f f f f f f f	
Literal ve Kontrol Komutları						
ADDLW	k	k 'yı W 'ya ekle	1	11 111x	kkkk kkkk	C,DC,Z
ANDLW	k	k 'yı W ile AND 'le	1	11 1001	kkkk kkkk	Z
CALL	k	k alt programını çağır	2	10 0kkk	kkkk kkkk	
CLRWDT		WDT yi sil	1	00 0000	0110 0100	$\overline{TO}$, $\overline{PD}$
GOTO	k	k adresine git	2	10 1kkk	kkkk kkkk	
IORLW	k	k ile W 'yı OR 'la	1	11 1000	kkkk kkkk	Z
MOVLW	k	k 'yı W 'ya taşı	1	11 00xx	kkkk kkkk	
RETFIE		Kesmeden geri dön	2	00 0000	0000 1001	
RETLW	k	k 'yı W 'ya yükle ve geri dön	2	11 01xx	kkkk kkkk	
RETURN		Alt programdan geri dön	2	00 0000	0000 1000	
SLEEP	--	Uyku moduna geç	1	00 0000	0110 0011	$\overline{TO}$, $\overline{PD}$
SUBLW	k	W 'yı k 'dan çıkart	1	11 110x	kkkk kkkk	C,DC,Z
XORLW	k	k ile W 'yı XOR 'la	1	11 1010	kkkk kkkk	Z

Sembol Tanımlamaları :

f → Register File Adress: kayıtçı adı veya adresi (0x00 ile 0x7F)

w → Akümülatör, çalışma kayıtçısı

b → Bit tanımlayıcısı; 8 bitlik kayıtçının 0~7 arasındaki bir biti veya etiket (EQU komutu ile adresi tanımlanmış olması gerekir)

d → Destination : Gönderilecek yer; komutun çalıştırılmasından sonra sonucun nereye yazılacağını belirler.

d = 0 → W kayıtçısına, d = 1 → dosya kayıtçısına

k → Sabit bir sayı (0x0C veya $0C_H$, 00001100_B, 10_D) veya adres etiketi

x → "0" yada "1" önemli değil

TO→ Zaman aşımı biti (Time-out bit)

PD→ Güç kesimi biti (Power-down)

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

000		GOTO BASLA	ORG 0X00 GOTO BASLA	BAŞLANGIÇ SATIRI	00 KONUMANA GOTO BASLA YAZ
001		GOTO KESME1		KESME SATIRLARI	01 KONUMANA GOTO KESME1 YAZ
002		GOTO KESME2			02 KONUMANA GOTO KESME2 YAZ
003		GOTO KESME3	ORG 0X01 GOTO KESME1		03 KONUMANA GOTO KESME3 YAZ
			ORG 0X02 GOTO KESME2	DEĞİŞKEN TANILAMA YERİ	03 NUMARALI ADRESİ STATUS OLARAK KULLANACAM
			ORG 0X03 GOTO KESME3		05 NUMARALI ADRESİ PORTA OLARAK KULLANACAM
			STATUS EQU H'03' PORTA EQU H'05'		
004		BSF STATUS,5	BSF STATUS,5	HAZIRLIK İŞLEMLERİ	BANK1 GEÇ
005		CLFR PORTA	CLRF TRISA;		PORTA BELLEK GÖZÜNÜ TEMİZLE(ÇIKIŞ YAP)
006		BCF STATUS,5	BCF STATUS,5		BANK0 GEÇ
				BAŞLANGIÇ İŞLEMLERİ	
007		CLRF PORTA	CLRF PORTA		PORTA ÇIKIŞLARINI SIFIR YAP
008	BASLA		BASLA	ANA PROGRAM BLOĞU	
009		…….	……		
010		……	……		
011		GOTO BASLA	GOTO BASLA		
012					
013	KESME1			KESME1 ALT PROGRAMI	
014					
015		RETFIE	KESME1		
016			……..		
…			RETFIE		
…					
017	KESME2			KESME2 ALT PROGRAMI	
018			KESME2		
019		RETFIE	……..		
			RETFIE		
020	KESME3				
021				KESME3 ALT PROGRAMI	
022		RETFIE	KESME3		
			……..		
023		END	RETFIE	PROGRAM SONU	
….			END		
……					
3FF					

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

16f84 mcu da genel ve özel amaçlı registerler bulunmaktadır. Bunlar porta, status ve kullanıcının kullanmak için tanımladığı kaydedicilerdir.

Bunlar bank0 yada bank1 de olabilir. Ama kullanıcının kullanımına açık kullanılan bellek gözleri her iki bankta da olsak erişebiliriz.

Kaydediciler üzerinde işlem yapmak için status kaydedicisi üzerinden bank değiştirmek gerekecektir.

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

```
BSF     STATUS,5    ;bank 1 e geçiş yapıldı
        CLRF        TRISB       ;b portu çıkış yapıldı
        MOVLW       H'FF'       ;
        MOVWF       TRISA       ;a portu giriş yapıldı
BCF     STATUS,5                ;tekrar bank sıfıra geçildi.
```

RP1:RP0	Bank
00	0
01	1
10	2
11	3

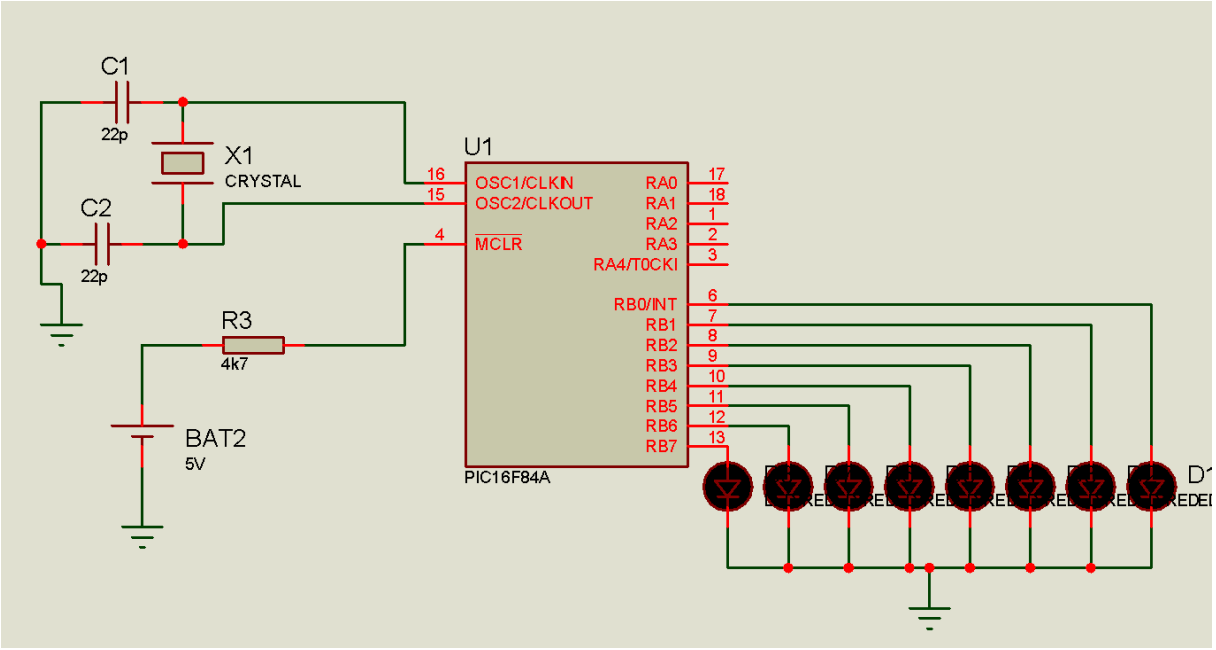
Kayıtçı Adresi	BANK0 Kayıtçı adı	Kayıtçı Adresi	BANK1 Kayıtçı adı
00_H	INDF	80_H	INDF
01_H	TMR0	81_H	OPTION
02_H	PCL	82_H	PCL
03_H	STATUS	83_H	STATUS
04_H	FSR	84_H	FSR
05_H	PORTA	85_H	TRISA
06_H	PORTB	86_H	TRISB
07_H	PORTC	87_H	TRISC
08_H	EEDATA	88_H	EECON1
09_H	EEADR	89_H	EECON2
$0A_H$	PCLATH	$8A_H$	PCLATH
$0B_H$	INTCON	$8B_H$	INTCON
$0C_H$ $7F_H$	(GPR)	$8C_H$ FF_H	(GPR)

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

1-bütün b portuna bağlı bütün ledleri yakma programı

```
LIST    P=16F84;************************************************
STATUS  EQU     H'03'
PORTA   EQU     H'05'
PORTB   EQU     H'06'
TRISA           EQU     H'85'
TRISB           EQU     H'86'
                CLRF    PORTB
                CLRF    PORTA
;***********************************************
                BSF             STATUS,5        ;bank 1 e geçiş yapıldı
                CLRF            TRISB           ;b portu çıkış yapıldı
                MOVLW           H'FF'           ;
                MOVWF           TRISA           ;a portu giriş yapıldı
                BCF             STATUS,5
;************************************************
                MOVLW           H'FF'           ;
                MOVWF           PORTB           ; bütün b portuna bağlı ledler yanar
                END
```



Bit düzeyinde işlem yapmak için BSF veya BCF komutları ile bir bitlik alanı 1 ya da 0 yapmada kullanılır. Aşağıda bunla ilgili kullanılan komutlar verilmiştir.

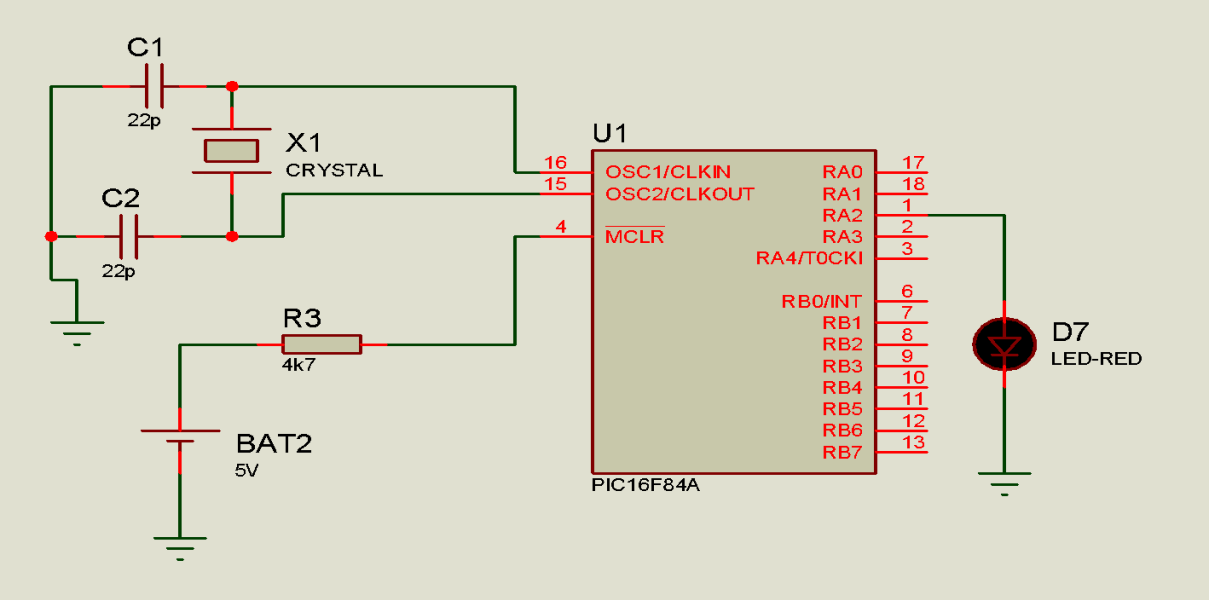
```
BSF             STATUS,5        ;bank 1 e geçiş yapıldı
BCF             STATUS,5        ;bank0 a geçilir
BSF             PORTA,2         ; a portunun 2 numaralı bacağına bağlı led yanar
```

Not
MOVLW B'000000**1**0' MOVWF PORTA
YADA
MOVLW H'02' kullanılabilir. MOVWF PORTA

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

2- a portuna bağlı bir led yakma

```
LIST      P=16F84
;*************************************************
STATUS EQU     H'03'
PORTA  EQU     H'05'
PORTB  EQU     H'06'
TRISA          EQU     H'85'
TRISB          EQU     H'86'
               CLRF    PORTB
               CLRF    PORTA
;*************************************************
               BSF             STATUS,5        ;bank 1 e geçiş yapıldı
               CLRF            TRISA           ;a portu çıkış yapıldı
               MOVLW           H'FF'           ;
               MOVWF           TRISB           ;b portu giriş yapıldı
               BCF             STATUS,5
;*************************************************
               BSF             PORTA,2         ; a portunun 2 numaralı bacağına bağlı led yanar
               END
```



Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

bit düzeyinde işlem yapma hakkında bilgi verilecek

3- b portuna bağlı led 2 ledi yakan program

```
LIST    P=16F84;************************************************
STATUS  EQU     H'03'
PORTA   EQU     H'05'
PORTB   EQU     H'06'
TRISA           EQU     H'85'
TRISB           EQU     H'86'
                CLRF    PORTB
                CLRF    PORTA
;*********************************************
                BSF             STATUS,5        ;bank 1 e geçiş yapıldı
                CLRF            TRISB           ;b portu çıkış yapıldı
                MOVLW           H'FF'           ;
                MOVWF           TRISA           ;a portu giriş yapıldı
                BCF             STATUS,5
;*************************************************
                BSF             PORTB,2         ;portb 2 numaralı bacağına bağlı led yakan progr.
                BSF             PORTB,1         ;portb 1 numaralı bacağına bağlı led yakan progr.
```

Bunun yerine yazılabilecek kod parçası
MOVLW B'00000110' MOVWF PORTB

```
                END
```

NOT
PROGRAMLARDA END KOMUTUNA PROGRAM VARDIĞINDA PROGRAM DURACAKTIR. GENEL OLARAK END KOMUTUNA GÖNDERİLMEZ… BUNUN İÇİN : SON GOTO SON END KOMUTU YAZILIR. YADA PROGRAM TEKRAR BAŞTAN ÇALIŞMASI İÇİN BASLA ……. Ana progra satırları …….. GOTO BASLA END BUNDAN SONRA YAZILACAK PROGRAMLARDA BU KOD BLOKLARI KULLANILACAKTIR!!

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

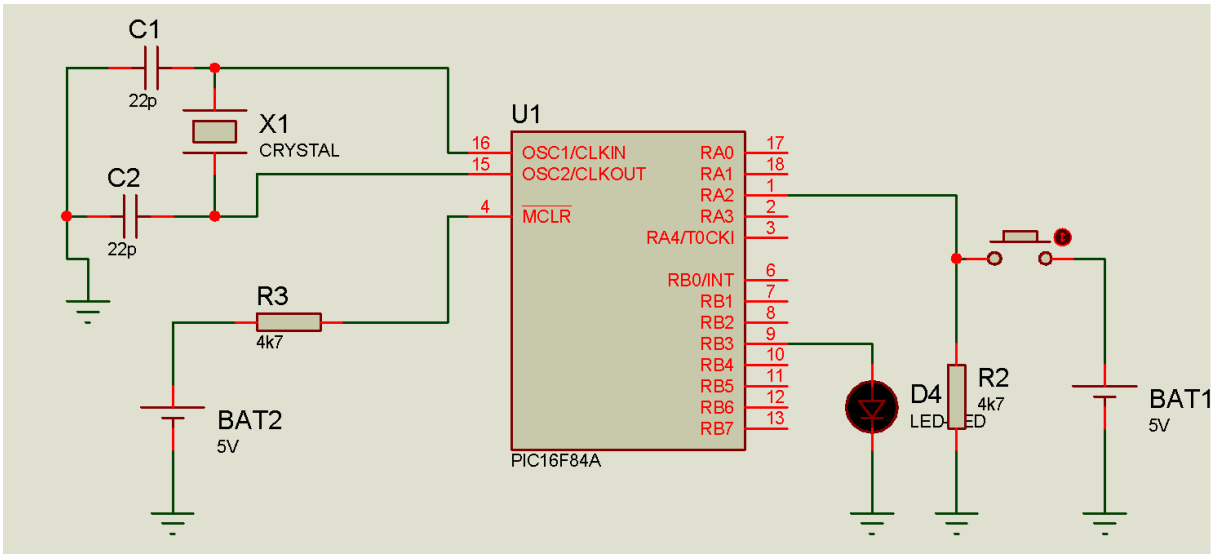
Bit düzeyinde işlemler sadece BSF ve BCF komutları ile olmaz. Bir bitlik bilginin 1 ya da 0 olup olmadığını test etmek için kullanılan komutlar bulunmaktadır. Bir bitlik bilginin 1 mi diye test eden komut BTFSS ve 0 mı diye kontrol eden komut BTFSC dir. Bu komutların test ettiği bilgi doğruysa bir satıt atlar ve devam eder. Örneğin aşağıda görülmektedir.

```
BT1     BTFSS   PORTA,2     ; portanın 2 numaralı bacağına bağlı butona basılmışsa potb,3 'e 5 volt ver
GOTO            BT1
BSF             PORTB,3     ;portb 3 numaralı bacağına bağlı led yak
```

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

4- a portunun 2 numaralı bacağına bağlı bir buton yardımı ile b portunun 3 numaralı bacağına bağlı bir led yakma programı

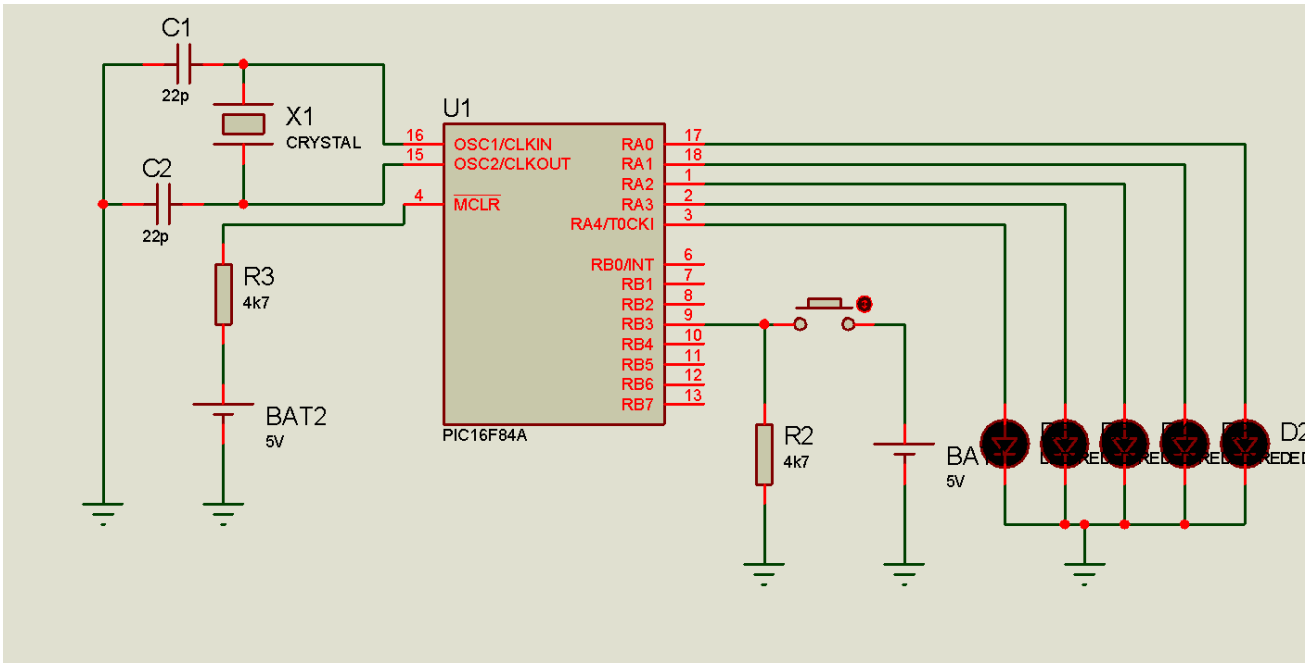
```
LIST    P=16F84;*************************************************
STATUS  EQU     H'03'
PORTA   EQU     H'05'
PORTB   EQU     H'06'
TRISA           EQU     H'85'
TRISB           EQU     H'86'
                CLRF    PORTB
                CLRF    PORTA
;***********************************************
                BSF             STATUS,5        ;bank 1 e geçiş yapıldı
                CLRF            TRISB           ;b portu çıkış yapıldı
                MOVLW           H'FF'           ;
                MOVWF           TRISA           ;a portu giriş yapıldı
                BCF             STATUS,5
;*************************************************
BASLA
BT1             BTFSS           PORTA,2         ; portanın 2 numaralı bacağına bağlı butona basılmışsa potb,3
'e 5 volt ver
                GOTO            BT1
BSF             PORTB,3         ;portb 3 numaralı bacağına bağlı led yak
                GOTO BASLA
;*************************************************
                END
```



Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

5- b portunun 3 numaralı bacağına bağlı bir butonla porta ya bağlı bütün ledleri yakan program

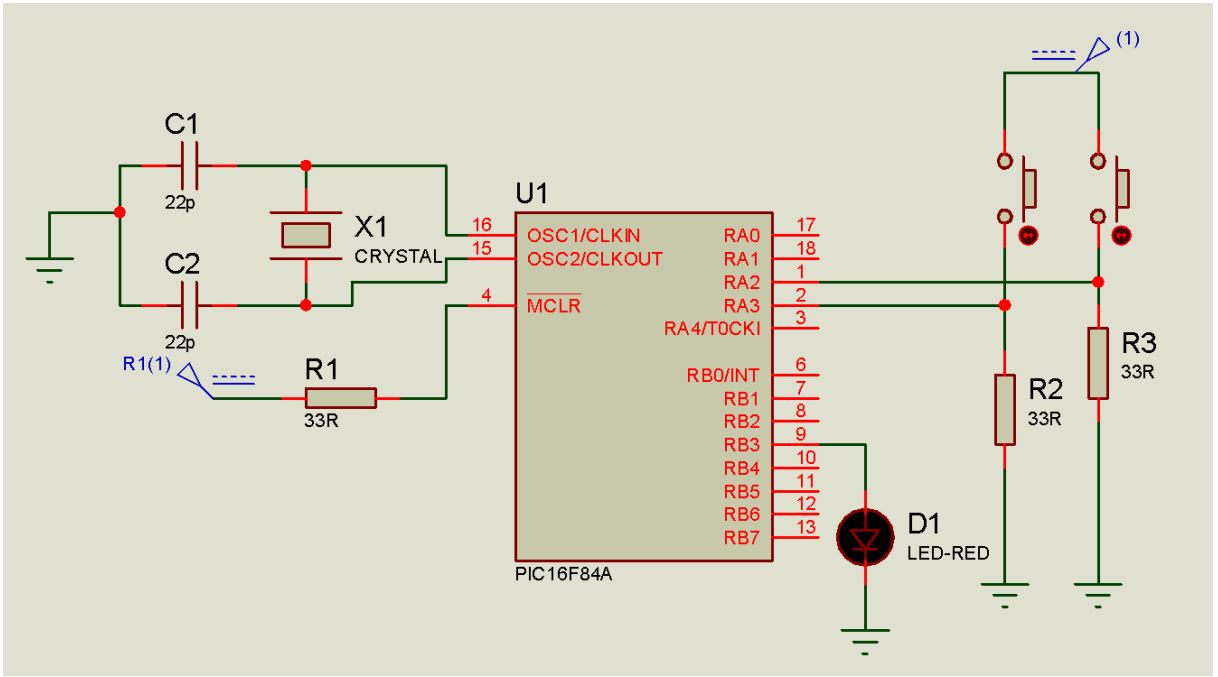
```
LIST      P=16F84
;*************************************************
STATUS EQU     H'03'
PORTA  EQU     H'05'
PORTB  EQU     H'06'
TRISA          EQU     H'85'
TRISB          EQU     H'86'
               CLRF    PORTB
               CLRF    PORTA
;*************************************************
               BSF             STATUS,5        ;bank 1 e geçiş yapıldı
               CLRF            TRISA           ;a portu çıkış yapıldı
               MOVLW           H'FF'           ;
               MOVWF           TRISB           ;b portu giriş yapıldı
               BCF             STATUS,5
;*************************************************
BASLA
BT1            BTFSS           PORTB,3         ; portBnin 3 numaralı bacağına bağlı butona basılmışsa porta ya bağlı bürün ledleri yak
               GOTO            BT1
MOVLW          B'11111111'
MOVWF          PORTA  ;portA ya bağlı ledleri yak
               GOTO BASLA
;*************************************************
               END
```



Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

6- a portunun 2 numaralı bacağına bağlı bir buton yardımı ile b portunun 3 numaralı bacağına bağlı bir led yakan ve a portunun 3 numaralı bacağına bağlı bir butonla da söndüren program

```
LIST    P=16F84;*************************************************
STATUS  EQU     H'03'
PORTA   EQU     H'05'
PORTB   EQU     H'06'
TRISA           EQU     H'85'
TRISB           EQU     H'86'
                CLRF    PORTB
                CLRF    PORTA
;***********************************************
                BSF             STATUS,5        ;bank 1 e geçiş yapıldı
                CLRF            TRISB           ;b portu çıkış yapıldı
                MOVLW           H'FF'           ;
                MOVWF           TRISA           ;a portu giriş yapıldı
                BCF             STATUS,5
;*************************************************
BASLA
BT1             BTFSS           PORTA,2         ; portanın 2 numaralı bacağına bağlı butona basılmışsa potb,3
'e 5 volt ver
                GOTO            BT2
BSF             PORTB,3         ;portb 3 numaralı bacağına bağlı led yak
BT2             BTFSS           PORTA,3         ; portanın 3 numaralı bacağına bağlı butona basılmışsa potb,3
'e0 volt ver

                GOTO            BT1
                BCF             PORTB,3
                GOTO BASLA
;*************************************************
                END
```



Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

7- b portunun 3 numaralı bacağına bağlı bir butonla porta ya bağlı bütün ledleri yakan ve b portunun 5 numaralı bacağına bağlı bir butonla a portuna bağlı bütün ledleri söndüren program

```
LIST      P=16F84
;*************************************************
STATUS  EQU     H'03'
PORTA   EQU     H'05'
PORTB   EQU     H'06'
TRISA           EQU     H'85'
TRISB           EQU     H'86'
                CLRF    PORTB
                CLRF    PORTA
;*************************************************
                BSF             STATUS,5        ;bank 1 e geçiş yapıldı
                CLRF            TRISA           ;a portu çıkış yapıldı
                MOVLW           H'FF'           ;
                MOVWF           TRISB           ;b portu giriş yapıldı
```

NOT
BSF TRISB,3 BSF TRISB,5

```
                BCF             STATUS,5
;*************************************************
BASLA
BT1             BTFSS           PORTB,3         ; portanın 2 numaralı bacağına bağlı butona basılmışsa potb,3
'e 5 volt ver
                GOTO            BT2
MOVLW           B'11111111'
MOVWF           PORTA   ;portA ya bağlı ledleri yak
BT2             BTFSS           PORTB,5         ; portanın 3 numaralı bacağına bağlı butona basılmışsa potb,3
'e0 volt ver

                GOTO            BT1

MOVLW           B'00000000'
MOVWF           PORTA   ;portA ya bağlı ledleri SÖNDÜR
```

NOT: YUKARIDAKİ İKİ SATIR İÇİN İKİ ALTERNATİF VAR BUNLAR AŞAĞIDAKİ GİBİDİR
CLRF PORTA ; YALNIZ BU PROGRAMA UGUN OLĞU İÇİN KULLANILDI
YA DA
COMF PORTA ; YALNIZ BU PROGRAMA UGUN OLĞU İÇİN KULLANILDI

```
                GOTO BASLA
;*************************************************
                END
```

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Zamanlama ve bekletme işlemi

Zamanlama denildiğinde aslında belirli bir süre MCU işlem yatırmama ya da işlem yaparken zamanın önemli olduğu yerlerde belirli bir süre bekletme amaçlı kullanılırlar. Harvard mimariyle üretilen mcular genel olarak her saat darbesinde bir komut işletmek amaçlı tasarlanmış mimarilerdir. Örneğin 4 mhz denildiğinde 1 saniyede 4 milyon komut icra edilecek anlamına gelir. İşte zamanlama kavramı burada devreye girmektedir.

Eğer 1 saniye bekletmek istersek 4 milyon komut işletmeliyiz ya da 4 milyon defa boş bir şekilde tekrarlatmalıyız. Bu işlemleri yaptırmak için döngüler kullanılmaktadır. Örneğin aşağıdaki program parçası 1 milyon defa saat darbesine ihtiyaç duyan bir döngü yapısıdır. Aşağıdaki yapının kaç saat palsine ihtiyaç duyduğunu bulmak için kabaca kat sayılar çarpılır daha sonra 3 katı alınır. Yani 100*30*100*3=900000 yaklaşık 0,9 saniyedir. Aşağıda iç içe bilok yapıları görülmektedir.

```
ZAMAN1S         MOVLW D'100'
                MOVWF SAYAC1
TIMER1  MOVLW D'30'
                MOVWF SAYAC2
TIMER2
                MOVLW D'100'
                MOVWF SAYAC3
TIMER3
        DECFSZ  SAYAC3,F
                GOTO    TIMER3
                DECFSZ  SAYAC2,F
                GOTO    TIMER2
                DECFSZ  SAYAC1,F
                GOTO    TIMER1
                RETURN
```

Değişkenlerin adresleri ve özel amaçlı registerler

Değişkenler kullanıcının kullanımı için genel amaçlı bellek ayrılmıştır. Bu bellek alanlarına erişim için adresi ile ulaşabileceğimiz gibi değişken tanımlayarak kullanabiliriz. Bu bellek gözleri H'0C' ile başlar. Örneğin üç adet değişken tanımlanması görülmektedir. Değişken tanımlarken EQU komutu ile bir bellek gözüne ataması yapılır. Değişken ismiyle aslında atanan bellek gözüne erişim yapılmış oluruz.

```
SAYAC1  EQU     H'0C'
SAYAC2  EQU     H'0D'
SAYAC3  EQU     H'0E'
```

Not:Bu değişkenlere erişim yapılırken bank değiştirmeye gerek yoktur. Hangi bankta olursak olalım erişim yapılabilir.

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

8- a portunun 2 numaralı bacağına bağlı bir buton yardımı ile b portunun 3 numaralı bacağına bağlı bir led yakan ve a portunun 3 numaralı bacağına bağlı bir butonla da söndüren VE İKİ SANİYE ARALIKLARLA BEKLEME YAPAN program

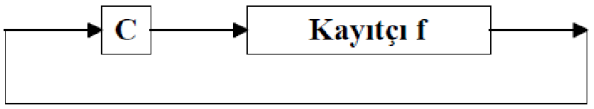
```
LIST    P=16F84;*************************************************
STATUS  EQU     H'03'
PORTA   EQU     H'05'
PORTB   EQU     H'06'
TRISA           EQU     H'85'
TRISB           EQU     H'86'
;*****************************
SAYAC1  EQU     H'0C'
SAYAC2  EQU     H'0D'
SAYAC3  EQU     H'0E'
;*************************
                CLRF    PORTB
                CLRF    PORTA
;*********************************************
                BSF             STATUS,5        ;bank 1 e geçiş yapıldı
                CLRF            TRISB           ;b portu çıkış yapıldı
                MOVLW           H'FF'           ;
                MOVWF           TRISA           ;a portu giriş yapıldı
                BCF             STATUS,5
;************************************************
BASLA
BT1
BTFSS           PORTA,2         ; portanın 2 numaralı bacağına bağlı butona basılmışsa potb,3 'e 5 volt ver
                GOTO            BT2
BSF             PORTB,3         ;portb 3 numaralı bacağına bağlı led yak
CALL            ZAMAN1S
CALL            ZAMAN1S
BT2             BTFSS           PORTA,3         ; portanın 3 numaralı bacağına bağlı butona basılmışsa potb,3
'e 0 volt ver
                GOTO            BT1
                BCF             PORTB,3
                CALL            ZAMAN1S
                CALL            ZAMAN1S
                GOTO BASLA
;*************************************************
ZAMAN1S         MOVLW           D'5'
                MOVWF           SAYAC1
TIMER1 MOVLW            D'100'
                MOVWF           SAYAC2
TIMER2
                MOVLW           D'100'
                MOVWF           SAYAC3
TIMER3
        DECFSZ SAYAC3,F
                GOTO    TIMER3
                DECFSZ SAYAC2,F
                GOTO    TIMER2
                DECFSZ SAYAC1,F
                GOTO    TIMER1
                RETURN
END
```

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Rrf ve Rlf komutları ;

Bir register içeriği sağa ya da sola kaydırılmak istendiğinde RRF ya da RLF komutları kullanılır. Yalnız baştaki yada sondaki kayma sonucunda elde bayrağına atılır. Elde bayrağı bilindiği üzere STATUS kaydedicisinde bulunan bir bitlik alandır.

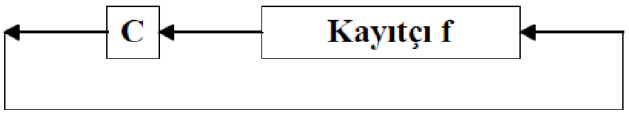
RRF:



Örnek:

```
RRF   SAG,1      ; komuttan önce SAG=H'02', ve C=0 ise, komut çalışınca
                 ; SAG=b '0000 0001'=01h ve C=0 olur.
RRF   SAG,1      ; komut bir kez daha çalışınca, SAG=b '0000 0000' ve C=1  olur.
```

RLF:



Örnek:

```
RLF   SOL,1      ;komuttan önce SOL=H'01', ve C=1 ise, komut çalışınca
                  SOL=b'0000 0011' = 03h ve C=0 olur.

RLF   SOL,1      ;komut bir kez daha çalışınca, SOL=b'0000 0110' ve C=0 olur.
```

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

9- iki buton yardımı ile portb ye bağlı ledleri sağa sola kaydırma programı

```
LIST P=16F84
STATUS EQU      03H
PORTA EQU       05H
PORTB EQU       06H
TRISA EQU       85H
TRISB EQU       86H
SAYAC1  EQU     H'0C'
SAYAC2  EQU     H'0D'
SAYAC3  EQU     H'0E'
SAYI    EQU     H'0F'

CLRF PORTA ;PORTA temizlenir
CLRF PORTB ;PORTB temizlenir

CLRF    SAYI
BSF             SAYI,0
BSF             STATUS, 5 ;BANK1'e geçilir
MOVLW   H'FF'
MOVWF   TRISA ;PORTA tüm uçlar giriş olacaktır
CLRF            TRISB ;PORTB tüm uçlar çıkış olacaktır
BCF             STATUS, 5 ;BANK0'a geçilir

BASLA
BT1     BTFSS PORTA,0 ;PORTA 0. bitini test et
        GOTO    BT2

        CALL    ZAMAN
        RLF     SAYI,1
        MOVF    SAYI,0
        MOVWF   PORTB
        GOTO    BASLA

BT2     BTFSS PORTA,1 ;PORTA 0. bitini test et
        GOTO    BT1

        CALL    ZAMAN
        RRF     SAYI,1
        MOVF    SAYI,0
        MOVWF   PORTB

        GOTO    BASLA

GOTO BASLA ;Başa dön
;*****************************************************
ZAMAN   MOVLW   D'5'
                MOVWF   SAYAC1
TIMER1  MOVLW   D'100'
                MOVWF   SAYAC2
TIMER2
                MOVLW   D'100'
                MOVWF   SAYAC3

TIMER3
                DECFSZ  SAYAC3,F
                GOTO    TIMER3
                DECFSZ  SAYAC2,F
                GOTO    TIMER2
                DECFSZ  SAYAC1,F
                GOTO    TIMER1
                RETURN

END ;Program sonu.
```

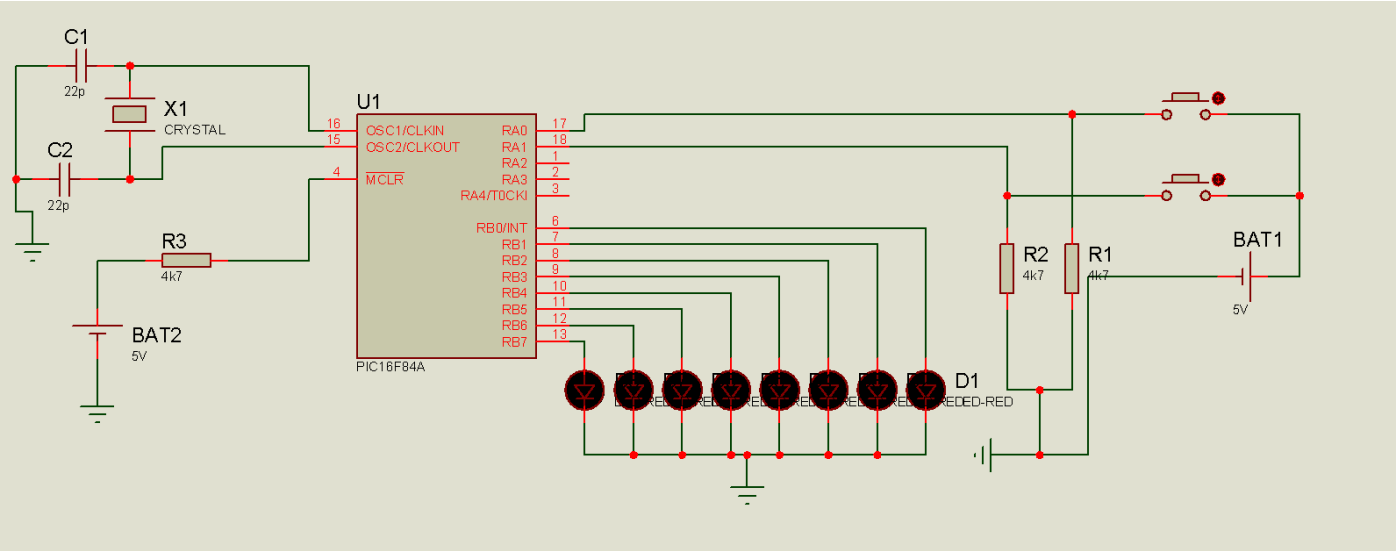
Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

10- artırma ve azaltma yapan bir program

```
        LIST P=16F84
STATUS EQU     03H
PORTA EQU      05H
PORTB EQU      06H
TRISA EQU      85H
TRISB EQU      86H
SAYAC1 EQU     H'0C'
SAYAC2 EQU     H'0D'
SAYAC3 EQU     H'0E'
SAYI    EQU    H'0F'


CLRF PORTA ;PORTA temizlenir
CLRF PORTB ;PORTB temizlenir
CLRF    SAYI
BSF STATUS, 5 ;BANK1'e geçilir
MOVLW          H'FF'
MOVWF TRISA ;PORTA tüm uçlar giriş olacaktır
CLRF TRISB ;PORTB tüm uçlar çıkış olacaktır
BCF STATUS, 5 ;BANK0'a geçilir

BASLA

BT1     BTFSS PORTA,0 ;PORTA 0. bitini test et
        GOTO   BT2

        CALL    ZAMAN
        INCF    SAYI,1
        MOVF   SAYI,0
        MOVWF          PORTB
        GOTO   BASLA

BT2     BTFSS PORTA,1 ;PORTA 0. bitini test et
        GOTO   BT1

        CALL    ZAMAN
        DECF    SAYI,1
        MOVF   SAYI,0
        MOVWF          PORTB

        GOTO   BASLA

GOTO BASLA ;Başa dön


;*****************************************************
ZAMAN MOVLW          D'5'
                MOVWF          SAYAC1
TIMER1 MOVLW         D'100'
                MOVWF          SAYAC2
TIMER2
                MOVLW          D'100'
                MOVWF          SAYAC3
```

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

```
TIMER3
        DECFSZ SAYAC3,F
        GOTO   TIMER3
        DECFSZ SAYAC2,F
        GOTO   TIMER2
        DECFSZ SAYAC1,F
        GOTO   TIMER1
        RETURN
END ;Program sonu.
```



Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

7 SegmentliDisplay Nedir

Aşağıdaki şekilde görüldüğü gibi led kullanılarak yapılan rakam, harf gösterici devre elemanlarına display denir. Yaygın olan yedi parçalı led göstergeler anodu şase (ortak) ve katodu şase olmak üzere iki tipte üretilir.

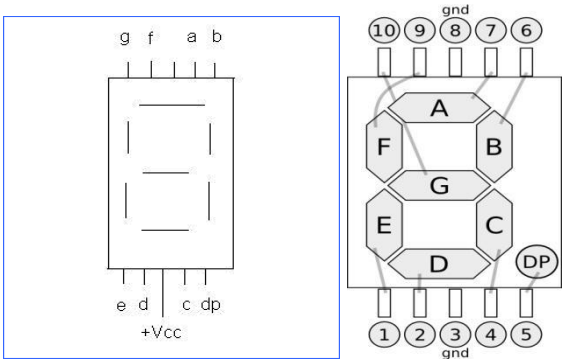
Ortak Anotlu (commonanode) display'ler

Bu tip display'lerin içinde bulunan tüm ledlerinanodları gövde içinde birbiriyle birleştirilmiştir. Eleman alıştırılırken artı (+) besleme ortak anoda uygulanır. Diğer uçlara uygulanan eksi (-) beslemelere göre display'de çeşitli rakamlar oluşur.

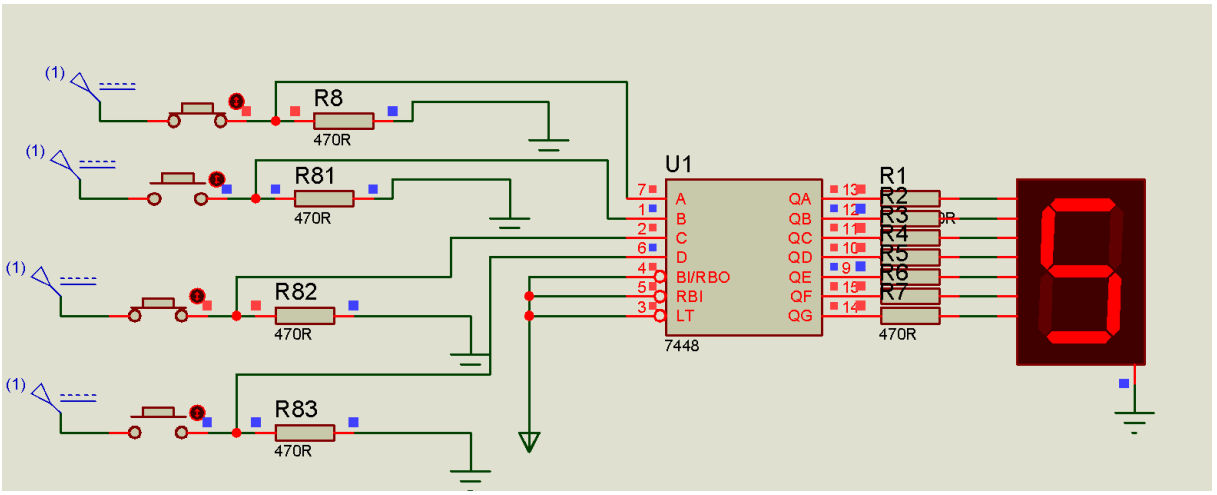
Ortak Katotlu (commoncathode) display'ler

Anodu şasenin tam tersi özelliktedir. Yani gövde içindeki ledlerin tümünün katot uçları birbirine bağlıdır.

Ortak Anotlu bir display'de ortak uca (+) besleme uygulanır. Display'de onlu 0 sayısını görebilmek için, a, b, c, d, e, f ledlerine kod çözücü entegre tarafından 0 V (yani şase) gönderilir. Display'dedesimal (onlu) 1 sayısı görülmek istendiğinde ise b ve c ledlerine 0 V uygulanır.7 Segmentli bir Display bacak bağlantı yapısı aşağıdaki gibidir.



Bu displayleri doğrudan bağlayabildiğimiz gibi bdc formatta düzenlenmiş bilgileri gösterme özelliğine de sahiptir. Fakat doğrudan kullanılamaz. Display bağlantı yapmadan önce 4511, 7447 ve 7448 entegreleri piyasada bulunmaktadır. Bağantı şekli aşağıda verilmiştir.



Yukarıdaki gibi 7448 bağlantısı yapmaksızın çevrim tabloları kullanılarak istediğimiz desende girilen bilgiyi dönüştürmeye yarayan yani entegre kullanmadan yapabileceğimiz program yapısıdır. Hem devremiz daha kolay hem de maliyeti düşürmesi açısından çevrim tabloları büyük önem arz etmektedir.

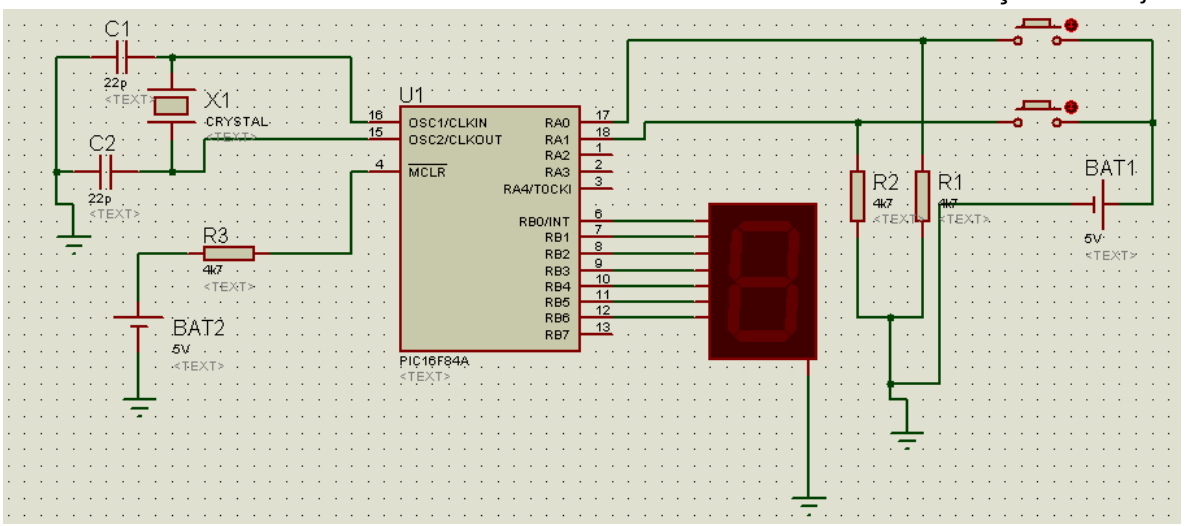
Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

11- 7 segmentli bir display e artırma ezaltma yaparak sayıları gösterme yapan program

```
        LIST P=16F84
STATUS EQU     03H
PORTA EQU      05H
PORTB EQU      06H
TRISA EQU      85H
TRISB EQU      86H
PCL            EQU     02H
SAYAC1 EQU     H'0C'
SAYAC2 EQU     H'0D'
SAYAC3 EQU     H'0E'
SAYI   EQU     H'0F'


CLRF PORTA ;PORTA temizlenir
CLRF PORTB ;PORTB temizlenir
CLRF   SAYI
BSF STATUS, 5 ;BANK1'e geçilir
MOVLW          H'FF'
MOVWF TRISA ;PORTA tüm uçlar giriş olacaktır
CLRF TRISB ;PORTB tüm uçlar çıkış olacaktır
BCF STATUS, 5 ;BANK0'a geçilir

BASLA

BT1    BTFSS PORTA,0 ;PORTA 0. bitini test et
       GOTO   BT2
       CALL   ZAMAN
       INCF   SAYI,1
       MOVF   SAYI,0
       CALL   TABLO
       MOVWF          PORTB
       GOTO   BASLA
BT2    BTFSS PORTA,1 ;PORTA 0. bitini test et
       GOTO   BT1
       CALL   ZAMAN
       DECF   SAYI,1
       MOVF   SAYI,0
       CALL   TABLO
       MOVWF          PORTB

       GOTO   BASLA

GOTO BASLA ;Başa dön
```

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen




```
;*****************************************************
TABLO
        ADDWF PCL,1 ; PCL ?? W( h '05')
        RETLW h'3F' ; 0
        RETLW h'06' ; 1
        RETLW h'5B' ; 2
        RETLW h'4F' ; 3
        RETLW h'66' ; 4
        RETLW h'6D' ; 5
        RETLW h'7D' ; 6
        RETLW h'07' ; 7
        RETLW h'7F' ; 8
        RETLW h'6F' ; 9
        RETLW h'77' ; A
        RETLW h'7C' ; B
        RETLW h'39' ; C
        RETLW h'5E' ; D
        RETLW h'79' ; E
        RETLW h'71' ; F

;*****************************************************
ZAMAN MOVLW     D'5'
        MOVWF   SAYAC1
TIMER1 MOVLW    D'200'
        MOVWF   SAYAC2
TIMER2
        MOVLW   D'100'
        MOVWF   SAYAC3

TIMER3
        DECFSZ SAYAC3,F
        GOTO   TIMER3
        DECFSZ SAYAC2,F
        GOTO   TIMER2
        DECFSZ SAYAC1,F
        GOTO   TIMER1
        RETURN


END ;Program sonu.
```

2013

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

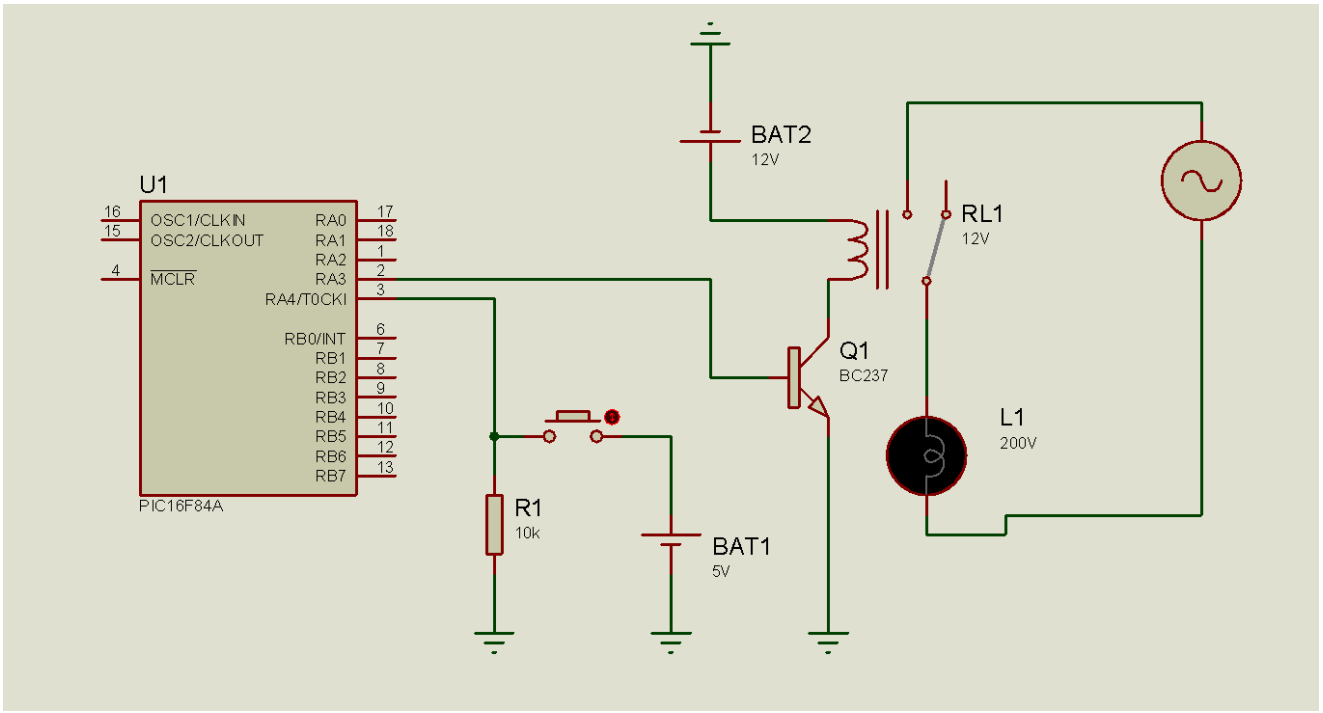
Mikrodenetleyici ile Röle kontrolü

Röle:

Röle küçük gerilimle yüksek gerilimleri kontrol edebilen mekanik bir elektronik anahtarlardır. Röleler birçok sistemi otomatikleştirmesine rağmen dezavantajları da bulunmaktadır.

Bunlar;

- Gürültü çıkartırlar.
- Yüksek akımda anahtarlar yapışabilir.
- Mekanik olmasından dolayı açma kapama sırasında ark oluşabilir.
- Arızalanması durumunda sökülüp takılması maliyet oluşturabilir.
- Değiştirilmesinden dolayı maliyeti ortaya çıkar.



Optokuplör

Mikrodenetleyicilerde yüksek gerilimle çalışılırken doğrudan bağlamak yerine yalıtım yapılması gerekmektedir. Yani fiziki bağlantı yerine ışıkla iletişim kurularak yapılabilir. Bu işlem için optotransistör kullanılabilir. Bu işleme optokuplaj olayı denilir.

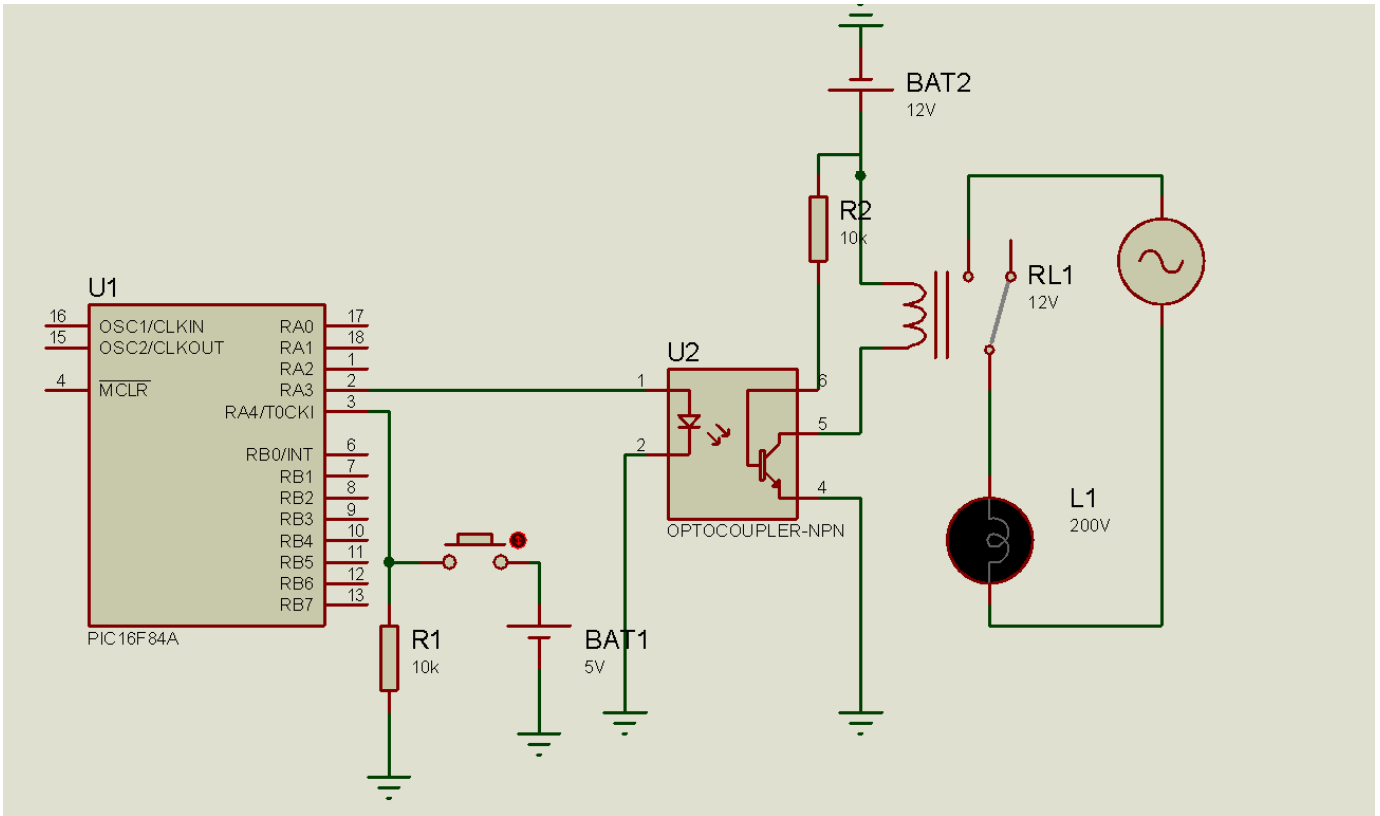
Avantajları:

- Yarı iletken olduğundan dolayı sıkça bozulmaz.
- İki sistemi birbirinden ayırdığından dolayı sistemi kontrol eden entegreler zarar görmez.

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Röle kontrolü için gerekli ASM kodu.

```
#include"C:\Users\a\Desktop\role\main.h"

#byte PA=05
#byte TPA=85
#byte STATUS=3

void main()
{

set_tris_a(0XF0);
output_a(0x00);
#asm
BASLA:
BTFSS PA,4
GOTO  BASLA
PAS:
BTFSC PA,4
GOTO  PAS
BSF   PA,3
GOTO  BASLA
#endasm
}
```

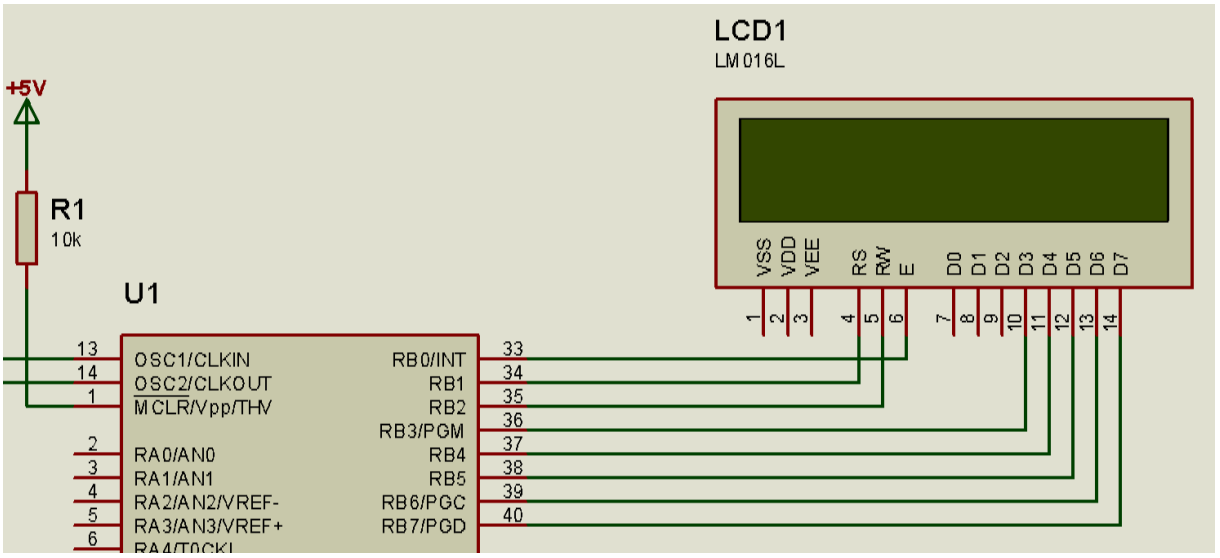


Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Lcd

İçerisinde sıvı bir yapıyla çalışan ekran türüdür. Genel olarak iki gruba ayırabiliriz. Grafik ve karakter lcdlerdir. Grafik lcd'ler piksel mantığına göre çalışır. Grafik komutları ile görüntü oluşturulur. İstenilen resim gösterilebilir. Karakter lcd'ler ise karakter ile çalışır.
İletişim yöntemine göre ise seri, 8 bit paralel, 4 bit paralel ve tek kablo yapılı olarak bulunmaktadır.
8 bit paralel demek ; 8 adet veri yolunun tamamının kullanılması ile olur. 4 bit paralel demek ise d4-d7 arasında veri yolu kullanan türüdür. Tek bit LCD'lerde ise bütün işlem bir hat üzerinden yapılır.
Genellikle pic ile LCD kontrolünde D portu ya da B portu tercih edilebilir. Bunu programda belirtmemiz gerekmektedir. CCS'de #define use_portb_lcd TRUE şeklinde.
Aşağıda iki satırlık bir LCD ekran görülmektedir.



VSS toprak
VDD +5V
VEE ekran karakter parlaklığını ayarlamada kullanılır.
RW Okuma yazma
D0-D7 data uçları
E Seçme ucu
RS Satır seçme ucu

Ekrana bilgi yazmak için CCS'de;

LCD kullanmak için aşağıdaki satırları ana programdan önce tanımlama yapılmalıdır.

```
#define use_portb_lcd TRUE
#define LCD TYPE 1
#define <lcd.c>
```

Bilgi yazmak için;

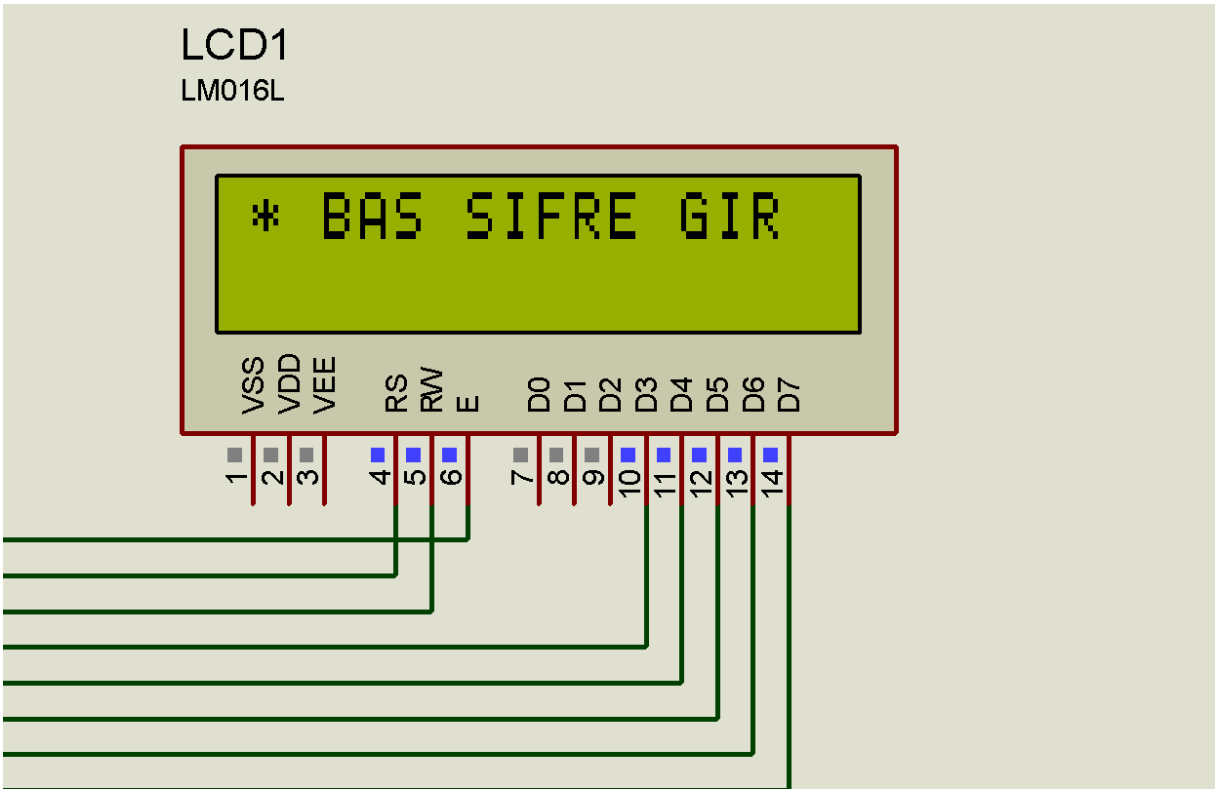
```
char veri='a';
Printf(lcd_putc,"\f%c",veri);
```

Komutumuzdaki % ifadesi ile veri türü belirlenir, bazıları şunlardır.
%s karakter dizisini yazdırmada kullanılır.
%d tam sayıları yazdırmada kullanılır.
%c tek karakterleri yazdırmada kullanılır.

Komutumuzdaki \ ifadesinde ise 3 adet kullanım vardır.
\f ekranı temizleme yapar
\n yeni satıra geçer
\b imleci bir karakter geri götürür.

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Aşağıdaki yazıyı yazmak için ccs'de şu yazılmalıdır.
Printf(lcd_putc,"* BAS SIFRE GIR");



Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Kesme

Çoğu zaman bir işlem yaparken başka bir işlemi de yapmak isteyebiliriz. Örneğin konuşurken duymak gibi. Fakat işlemcilerle bir işlem yaparken başka bir işlem yapmak mümkün değildir. Fakat iki işlemi bir arada yapmak ihtiyacı doğabilir. Bu işlemi yapmak istersek kesme kullanmadan yapmak mümkün değildir. İşlemcilerde ve mikrodenetleyicilerde bu işlemi yapmak için kesmeler kullanılır. Bir işlem yapılırken başka bir işlem için, yapılan iş kesilip diğer iş yapılır. Kesmeler birden fazla süreci aynı anda yürütmek için kullanılan en iyi yöntemdir.

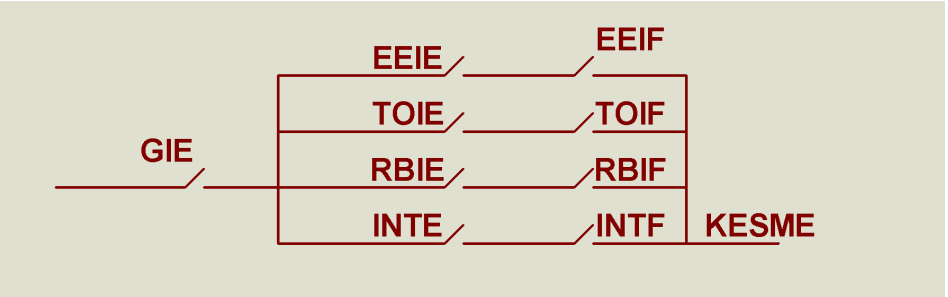
16f84'de 4 tane kesme bulunmaktadır. Bunlar;

- Eeprom kesmesi
- Timer0 kesmesi
- B4-B7 değişim kesmesi
- B0 kesmesi.

Kesmeleri ayarlamak için intcon kaydedicisini ayarlamak gerekmektedir. İçeriği aşağıda görülmektedir. Bir kesmenin oluşabilmesi için genel kesmenin aktif edilmesi gerekmektedir.

7							0
GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF
Genel kesme	Eeprom Aktif etme	Timer0 aktif etme	Rb0 Aktif etme	B4-b7 Aktif etme	Timer0 Kesme belirteci	B4-b7 Kesme belirteci	B0 kesme belirteci

Kesme programının çalışma şekli;



Eğer RB0 kesmesi yapıcaksa şöyle olmalıdır.

1	0	0	1	0	0	0	0
GIE			INTE				

Intcon içeriği "90" olmalıdır.

Eğer TMR0 kesmesi yapmak istersek;

1	0	1	0	0	0	0	0
GIE		TOIE					

Intcon içeriği "A0" olmalıdır.

Eğer RB4-RB7 arasındaki portlarda bir değişme olmuşsa çalışan kesme için;

1	0	0	0	1	0	0	0
GIE				RBIE			

Intcon içeriği "88" olmalıdır.

Eğer dış kesme oluşmuşsa aşağıdaki gibi bir durum oluşur;

1	0	0	0	0	0	0	1
GIE							RBIF

Intcon içeriği "81" olur. Fakat kesmeyi tekrar aktifleştirmek için

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

1	0	0	1	0	0	0	0
GIE			INTE				

Intcon içeriği “90” olmalıdır.

Pic’in çalışma Mantığı

İlk defa çalışmaya başlayan pic 00 adresinden başlayarak çalışmaya başlar.

Eğer kesme oluşmuşsa 01-04 arasındaki adreslere bakar. Programda kesme programı yazılacaksa daha önceden bu adreslerin içeriği düzenlenmelidir.

İstenilen adrese konumlanmak için ORG komutu kullanılır. Aşağıda ALARM adındaki kesme programının nasıl yazıldığı görülmektedir.

ORG H’00’	00	GOTO ANA
GOTO ANA		
ORG H’04’		
GOTO ALARM		
	04	GOTO ALARM
ANA		ANA
…..		…..
GOTO ANA		GOTO ANA
ALARM		
…..		ALARM
RETFIE		…
END		RETFIE
		END

Örnek:

RB4-RB7 arasındaki port değişimini kontrol eden programı yazınız.

```
#include<p16F84.inc>
ORG H'00' ;Program başlangıç adresini ayarla
GOTO START ;Ana programa geç
ORG H'04' ;Kesme alt programı başlangıç adresini ayarla
GOTO KESME
        START ; Ana program başlangıcı
        BSF STATUS, 5 ; BANK1’e geçilir
        BCF TRISA, 0 ; PORTA.0 bit çık
        BCF STATUS, 5 ; BANK0’a geçilir
        CLRF    PORTA
        MOVLW b'10001000'
        MOVWF INTCON ;Kesme kontrol yazmacını harici kesme için ayarla
DUR
        GOTO DUR ;ve normal şartlarda hiçbir işlem yapmadan sonsuz
KESME
        COMF   PORTA
        MOVLW b'10001000'
        MOVWF INTCON ;Kesme kontrol yazmacını harici kesme için ayarla
        RETFIE ;Kesme alt programı sonu

END ;Program sonu
```

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Seri iletişim

Genel olarak iki tür iletişim bulunmaktadır. Bunlar seri ve paraleldir.

Seri iletişim, iletişim yöntemlerine göre ve zamanlama açısından iki durumu bunmaktadır.

İletişim yöntemleri:

- **Tek yönlü:**

Bu iletişim yöntemi, alıcı ve gönderici arasında yapılan iletişimdir. Örneğin tv ve kumanda iletişimi

- **Sıralı tek yönlü:**

Bu iletişim yönteminde alıcı ve gönderici sırasıyla değişen yöntemdir. Örneğin telsiz iletişimi.

- **Çift yönlü:**

Bu iletişimde alıcı ve gönderici aynı anda bilgi gönderip alabilir.

Zamanlama yöntemi:

- **Eş zamansız(asenkron):**

Bu iletişimde zamanlama gönderilen bilgiye ve gönderici-alıcının hızlarına bağlı olarak değişen ve zamanın sabit bir birim olmadığı durularda kullanılan yöntemdir. Örneğin gönderici birim saniyede 10 iş yaparken alıcı 20 iş yapması gibi düşünebiliriz.

- **Eş zamanlı(senkron):**

Bu iletişimde zamanlama gönderen ve alan için yapılan işlem aynı anda başlar ve biter. Örneğin birim saniyede her zaman 20 iş yapması ve bu işlemin gönderici ve alıcı tarafından aynı şekilde tamamlanması olayı gibi düşünebiliriz.

Seri iletişim nasıl olur:

Seri iletişim 8 birlik veriler halinde gerçekleştirilir. Fakat giden bilgi sadece 8 bitlik bilgiden ibaret olmaz. Bilgi gönderme başlangıç biti. Bilgi bitiş bit ve zamanlama bilgileri de bulunur. Bilgi 8 bit

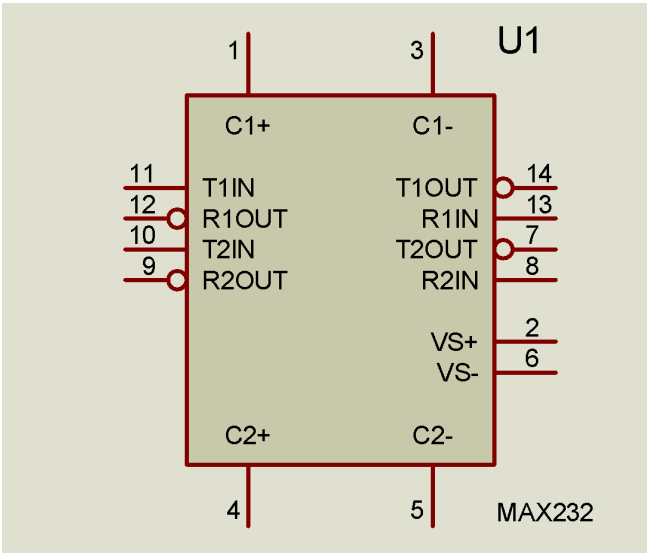
Olarak paket halinde gitmez. İletişim hızına göre (bound) 1 bit bir bit şeklinde gider. Bunlar gönderilirken mutlaka gidecek bilgi 8 bit birer bit şeklinde ayrılıp gönderilmelidir. Bu işlemi alıcıda bit bit alıp 8 bitlik paketler haline getirir.

- **Seri iletişim hızı:**

İletişim hızı bound olarak ifade edilir. Saniyede gönderilecek ya da alınacak bit sayısını ifade eder. Genel olarak 9600 kullanılır. Bu sayı saniyede alınacak ya da gönderilecek bilginin saniyedeki sayısını ifade eder.

- **Seri iletişim nasıl gerçekleştirilir:**

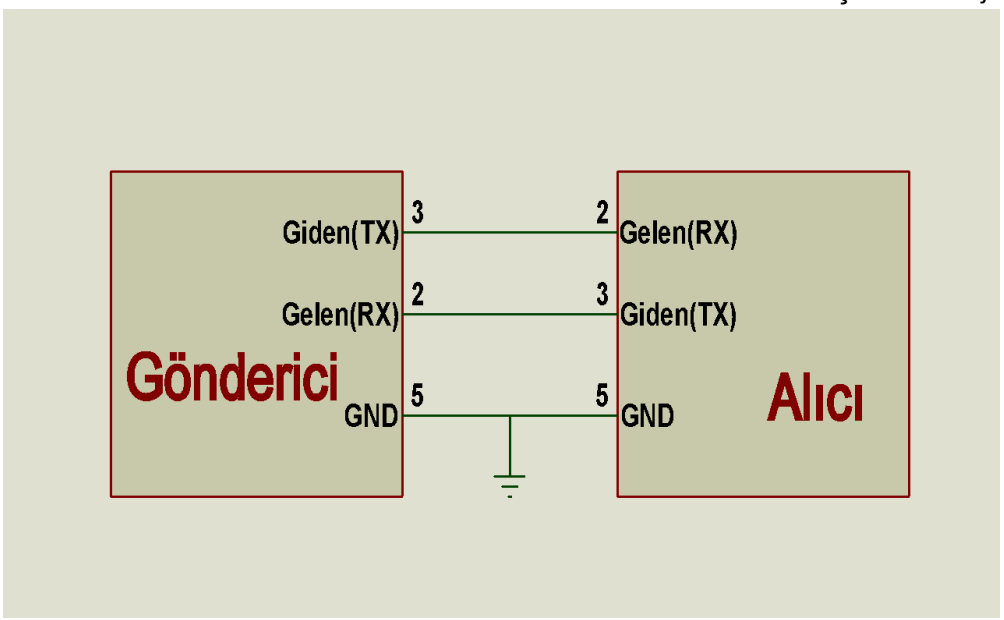
Seri iletişim genel olarak doğrudan alıcı ve verici bağlanarak gerçekleştirilmez. Arada gerilim ve hız dengesini sağlayacak yani el sıkışma dediğimiz olayı ve tamponlanma yapacak entegreler kullanılır. Bu max232 entegresidir. Bu entegre bağlanmazsa alıcı ya da gönderici zarar görmesi muhtemeldir. Yapısı aşağıda görülmektedir.



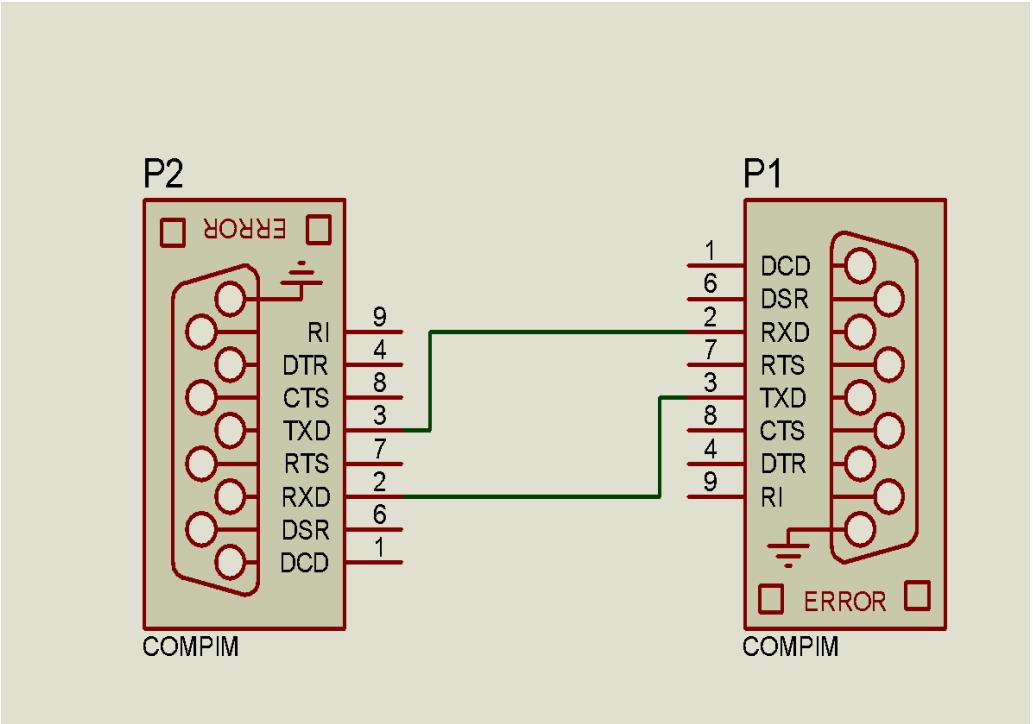
Seri iletişim bağlantı aparatı

Bu aparatın iki türü bulunmaktadır. Bu 9 pin ve 25 pin olarak piyasada bulunmaktadır. Gelişen teknoloji ile seri portu olmayan bilgisayarlar için satılan dönüştürme aparatı bulunmaktadır. Bu usb seriport şeklindedir. Basit bir iletişim için 3 uç kullanılması yeterlidir. Şekli aşağıda görülmektedir.

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

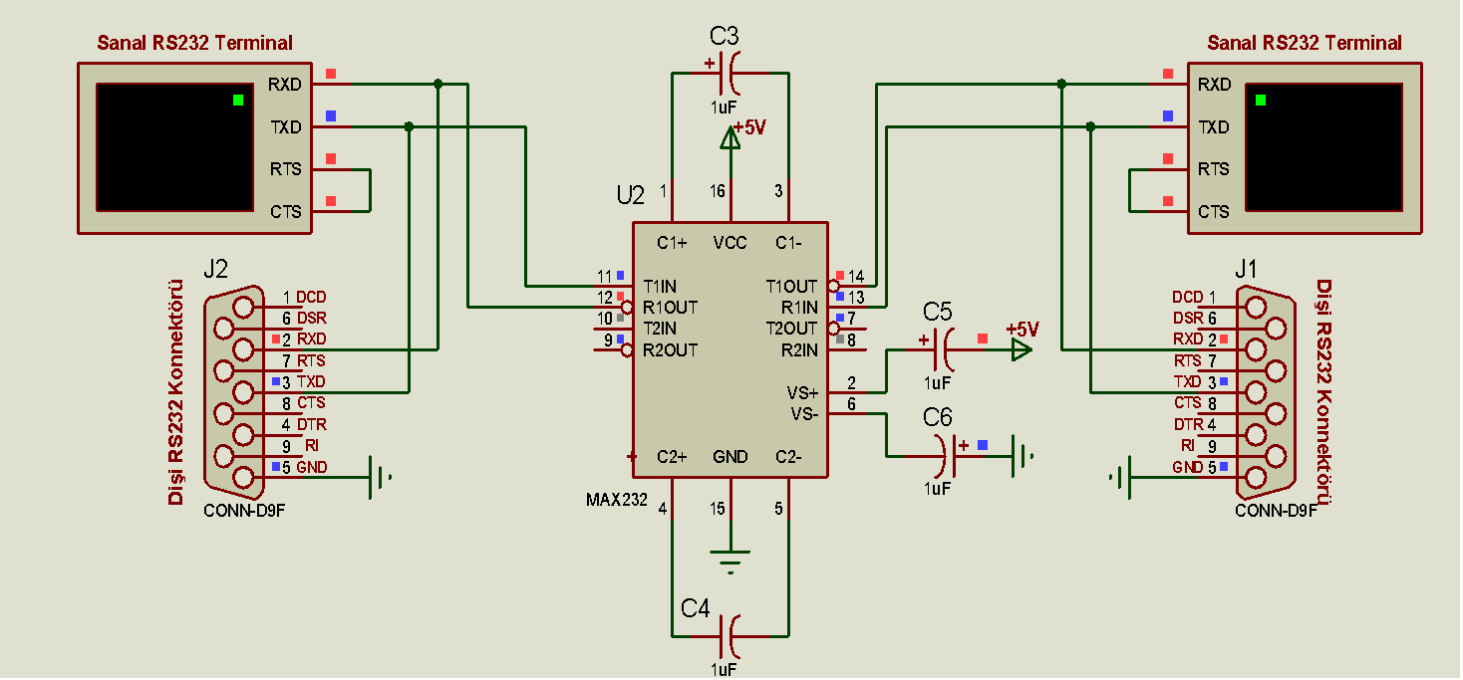


Aynı bağlantının 9pinli port ile bağlantısı aşağıda görülmektedir.



Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Yukarıda görülen p1 ve p2 cihazı şekildeki gibi haberleşmesi birçok sıkıntıya sebep olduğunu yukarıda max232 de bahsetmiştik. Sorunsuz bir bağlantı için aşağıda şekil görülmektedir.



Yukarıda iki tane sanal terminal arasında iletişim için simülasyon programında kullanmak için çizilmiş bir çalışmadır. Doğru bir iletişim için max 232 kullanmamız gerekmektedir.

Pic ile seri iletişim :

Pic ile seri iletişim yapmak için yüksek seviyeli diller ile çok kolay yapılabilmektedir. Fakat asm ile yapıldığında işlemlerin nasıl olduğu konusunda bir fikrimiz olacaktır. Seri iletişimde bilgiler tek tek gönderildiğini unutmadan örneğimizi verilim.

TX MOVLW D'8' MOVWF SAYAC **TX_LOOP** BTFSS TVERI,0 BCF PORTA,0 BTFSC TVERI,0 BSF PORTA,0 RRF TVERI,1 DECFSZ SAYAC,1 GOTO TX_LOOP **RETURN**	Bu programda TX adında etiketle alt program olarak tanımlanmıştır. Program içerisinde kullanılmak istendiğinde CALL TX dememiz gerekmektedir. Bu program gönderilen veri 8 bir olmasından dolayı sayaç adındaki değişkene 8 değeri yüklenmiştir. Bu sayı bize bilgiyi 8 defada göndermemize yarayacaktır. Gönderilen veri hep 0 bitte tutuluyor. Veri bir bit gönderildikten sonra tekrar 0 bite bilgi kaydırılarak devam eden bir döngü ile veri gönderimi yapılıyor. Bilgi ise TVERI adındaki bellek gözünde tutulmaktadır.

Bu işlem CCS C ile daha kolay yapılabilmektedir.

Seri iletişim veri gönderimi
Printf(veri); formatlı veri gönderimi yapılır.
Putc(); tek karakterlik bilgi gönderimi yapılır.
Puts(); birden fazla karakter gönderimi yapılabilir.
Seri iletişimde veri alım :
Gets(veri); birden fazla bilgi alma işlemini yapar.
Getch();tek karakterlik bilgi alama işlemini yapar.

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

I^2C(Inter IntegratedCircuit)

Seri haberleşme veri yolu azlığı ve maliyet açısından birçok alanda tercih edilen bir iletişim yöntemidir. Bundan dolayı birçok firma seri iletişim yöntemini kullanan cihazlar üretmektedir. Bun entegrelerin seri iletişim yöntemine I^2C yöntemi denilir.

Rs-232 Nedir? Rs-485 Ne demektir? Rs232 nasıl çalışır? Rs485 nerede kullanılır?

RS232, RS422, RS423 VE RS485 bilgisayarlar ve diğer elektronik cihazlarda kullanılan seri haberleşme metotlarıdır.

Kuşkusuz ki RS232 bu seri haberleşme metotları arasında en iyi bilinenidir, hemen hemen tüm bilgisayarlarda bir RS232 çıkışı bulunmaktadır.

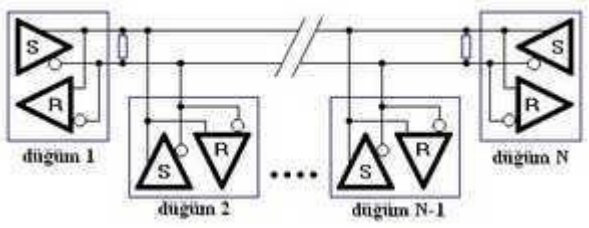
RS232 maksimum 20 kbps veri iletim hızında sadece bir DTE (data terminal cihazı) nin sadece bir DCE (data haberleşme cihazı) ye bağlanabildiği bir haberleşme ara yüzüdür ve bu iki cihaz arasındaki maksimum kablo uzunluğu 15 metre olabilir. Bu mesafe ilk zamanlarda yeterli gelmekteydi ancak daha sonra teknoloji ve buna bağlıolarak ihtiyaçlar değişti :

- Daha uzun mesafede haberleşme
- Birden fazla DTE bağlama
- Daha hızlı haberleşme

RS485 EIA tarafından tanımlanmış çok yönlü bir seri haberleşme standardıdır. Yukarıda belirtilen ihtiyaçların hepsini sağlar. Bu yüzden birden fazla cihazın birbirleriyle haberleşmesi gereken veri işleme, ve kontrol uygulamalarında yoğun bir şekilde kullanılır.

RS232 nin en temel problemi sinyal hattı üzerindeki gürültüden kolay etkilenir olmasıdır. RS232 protokolü alıcı ve verici arasındaki data ve handshakeline voltajlarını ortak bir toprak hattı kullanarak karşılaştırır. Toprak hattındaki herhangi bir voltaj artımı felaket sonuçlar doğuracaktır. Bu yüzden RS232 tetikleme seviyesi +/- 3volta ayarlanmıştır. Bu nedenle mesafe arttığında gürültü hızla artar. RS485 standardında ise sinyal referansı için ortak sıfır kullanılmaz. Bu sebeple RS485 alıcı ve verici ünite arasındaki voltaj seviye farkı bir problem oluşturmaz. RS485 sinyalleri değişkendir ve her bir sinyal Sig+ ve Sig- hatları üzerinde iletilir. RS485 alıcısı sinyal hattı üzerindeki kesin voltaj seviyesi yerine iki hat arasındaki voltaj farkını karşılaştırır. Bu sayede bir çok haberleşme sorunun temeli olan toprak döngüsü önlenmiş olur.

RS485 in network yapısı data işleme ve kontrol uygulamalarında yoğun bir şekilde kullanılmasının ana nedenidir. 12 kohm giriş direnci ile networke 32 cihaza kadar bağlantı yapılabilir. Daha yüksek giriş direnciyle bu sayı 256 ya kadar çıkarılabilir. RS485 tekrarlayıcıları ile bağlanabilecek cihaz sayısı birkaç bine, haberleşme mesafesinde birkaç kilometreye çıkabilir. RS485 bunun için ayrıca bir donanım istemez yazılım kısmıda RS232 den zor değildir.



Yukarıdaki resim RS485 network yapısını göstermektedir. N kadar düğüm çok noktalı RS485 networküne bağlanmıştır. Hattın iki ucundaki R dirençleri 100 ohm seçilerek yansıma önlenmiş olur böylece daha yüksek hız ve daha uzun mesafeye erişilmiş olur.

<u>RS485 in belli başlı teknik özellikleri</u>

Maksimum sürücü sayısı : 32
Maksimum alıcı sayısı : 32
Çalışma şekli : HalfDuplex
Network Yapısı : Çok noktalı bağlantı
Maksimum Çalışma Mesafesi : 1200 metre
12 m kablo uzunluğunda maksimum hız : 35 Mbps
1200 m kablo uzunluğunda maksimum hız : 100 kbps
Alıcı giriş direnci : 12 kohm
Alıcı giriş duyarlılığı :+/- 200 mvolt

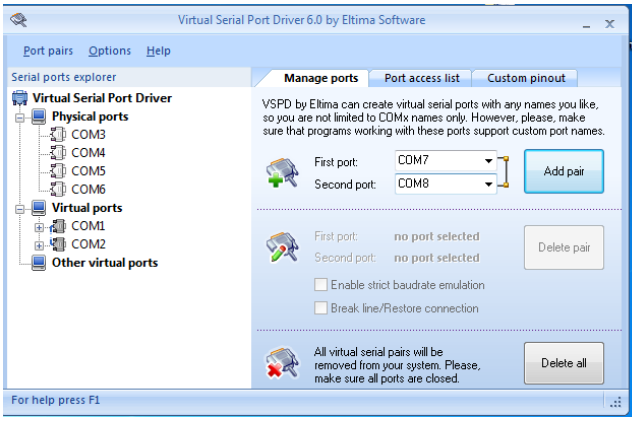
Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Alıcı giriş aralığı : -7…12 volt
Maksimum sürücü çıkış voltajı : -7…12 volt
Minimum sürücü çıkış voltajı (yük bağlı durumda) : +/- 1.5 volt

Seri İletişim için kullanılan programlar

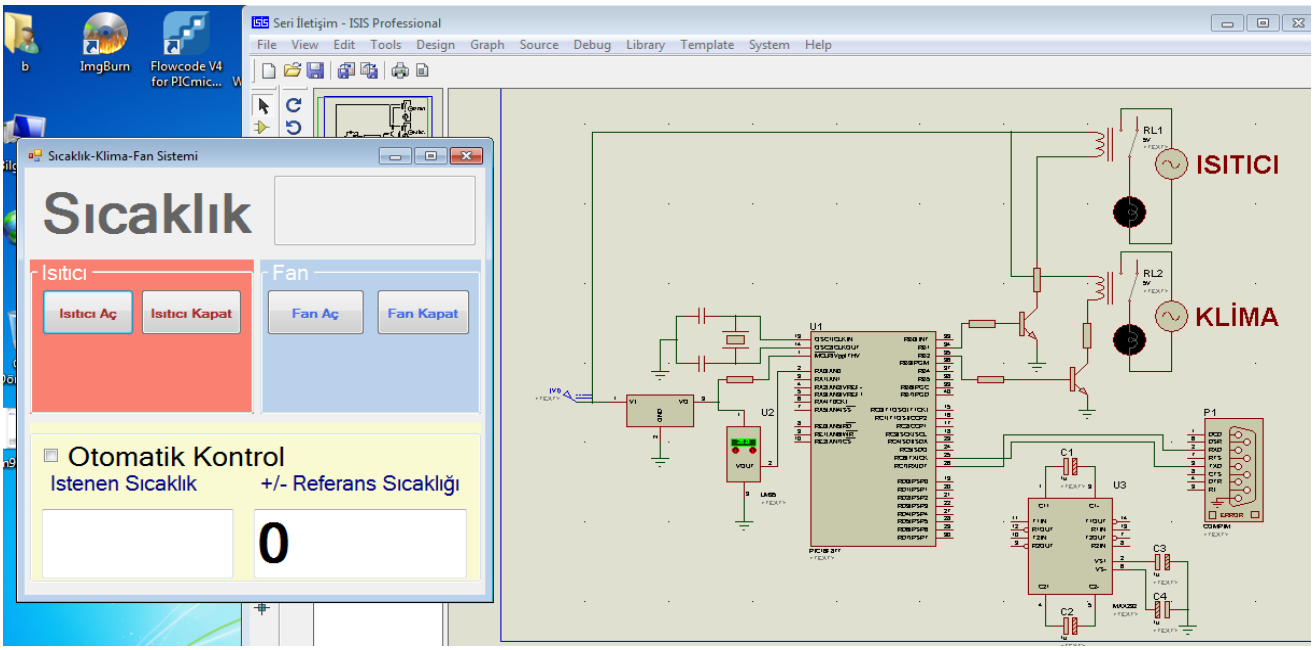
Bilgisayarda simülasyon yapmak istediğimizde port çakışmasıyla karşılaşırız. Bunun için sanal portlar oluşturabiliriz. Bunun için viturualserial port programları bulunmaktadır. Aşağıda bu amaçta bir program görülmektedir. Aşağıda sanal com1 ve com2 bulunmaktadır. C# yazılmış bir programla isis aynı anda çalıştırılıp çalıştırılabilir.



C# kodu	CCS kodu

`if(!serialPort1.IsOpen) serialPort1.Open();`
 	
`serialPort1.Write("v");`
`gl = serialPort1.ReadLine();`
`textBox1.Text = gl;`
 	
`serialPort1.Close();`
`}`
	`bilgi=read_adc();`
	`sicaklik=bilgi*2;`
	`//putc(sicaklik);`
	`gb=getc();`
	`itoa(sicaklik,10,gd);`
	`if (gb=='v')`
	`{`
	`//output_high(PIN_D1);`
	`puts(gd);`
	
	`}`

Aşağıdaki program C# ile yazılmış progra ile isis programı arasında iletişimin yapıldığı bir resimdir.



Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Bellek Kulanımı

Bellek pic içerisinde farklı amaçlar için birden fazla bulunmaktadır. Bu bellekler harward mimarisinden dolayı bizim pek alışık olmadığımız bir mantıkta dizayn edilmiştir. Normal bir bilgisayarda(vonneuman) hdd, ram bulunur. Kalıcı veriler hdd'de saklanır. Fakat harward mimarisinde işlemler böyle olmaz. Temel üç adet hafıza bulunur.

Bunlar;

- Program hafızası
- Özel amaçlı yarı static(kalıcı) ram
- Tasarımcının kullanacağı özel eeprom

Biz özel amaçlı verileri kaydetmek için eeprom yapıdaki ram kullanacağız. CCs'de okuma ve yazma komutları şunlardır;

write_eeprom(adres,k); istenilen adrese bilgi yazmak için kullanılır.
k= read_eeprom(adres); istenilen adresten bilgi okumak içim kullanılır.

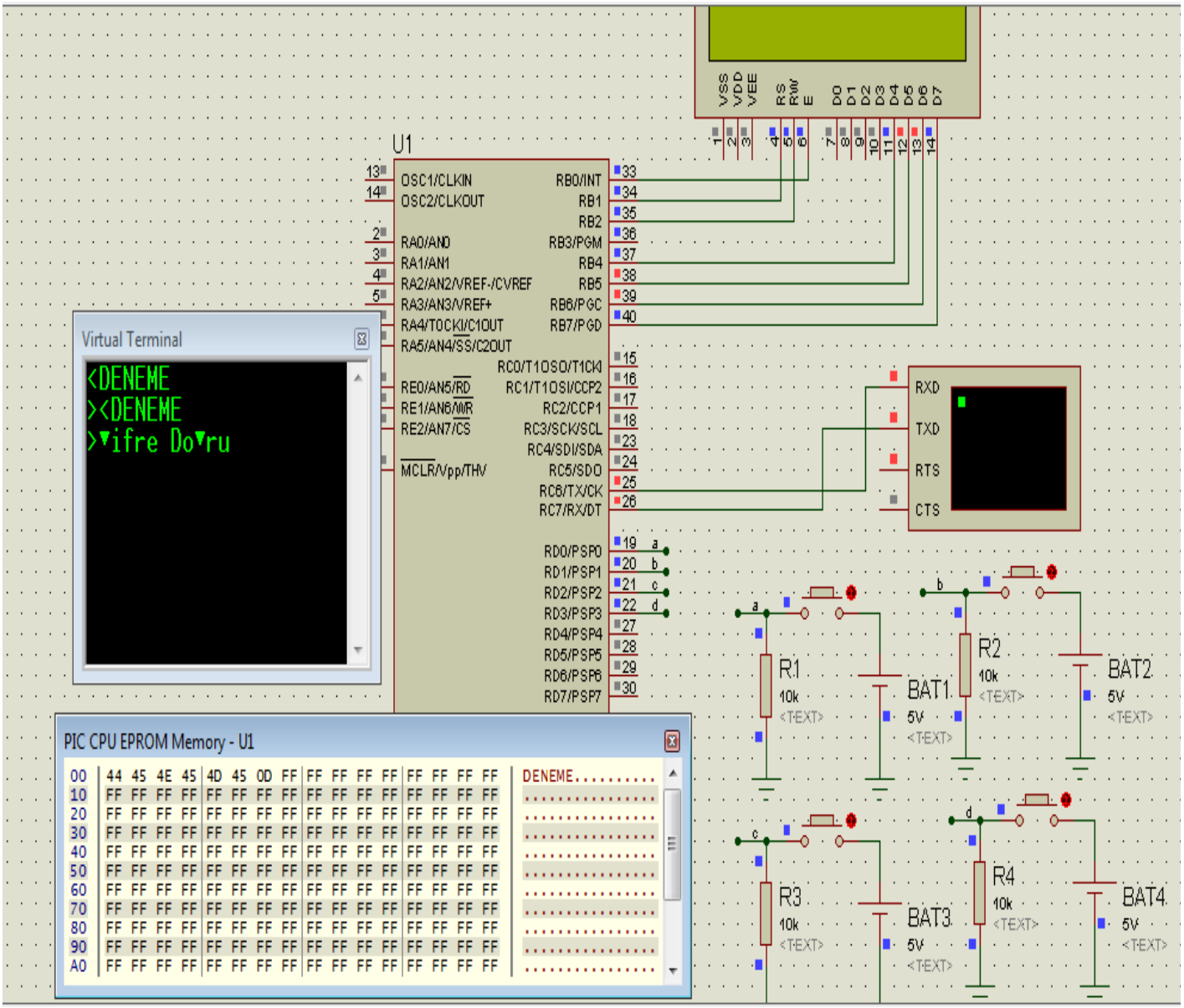
Aşağıda bilgi yazma, okuma karşılaştırma yapan fonksiyonlar bulunmaktadır.

```
#include<main.h>
#include<string.h>
#defineuse_portb_lcd TRUE
#define LCD_TYPE 1
#include<lcd.c>
char klavye[15];
char bellek[15];
/*************** klavye oku **********/
voidklavyeoku()
        {
int adres=0;
char k;
printf("<");
do
        {
        k=getch();
printf("%c",k);
        klavye[adres]=k;
        adres++;
        } while(k!=0x0d);
printf(">");
        }
/***************** bellek yaz ********/
voidbellekyaz()
        {
int adres=0;
char k;
printf("<");
do
        {
        k=getch();
printf("%c",k);
write_eeprom(adres,k);
        adres++;
        } while(k!=0x0d);
printf(">");
        }
/************ bellek oku ************/
voidbellekoku()
        {
int adres=0;
char k;
printf("<");
do
        {
        k= read_eeprom(adres);
printf("%c",k);
        bellek[adres]=k;
         adres++;
         } while(k!=0x0d);
printf(">");
         }
/********** karşılaştır ************/
voidkarsilastir()
         {
//printf("%s\n",klavye);
//printf("%s\n",bellek);
if(!strcmp(klavye,bellek))
         {
printf("şifre Doğru");
         }
else
         {
printf("şifre Yanlış");
         }

         }

/***************** ana program ************/
void main()
{
lcd_init();
setup_adc_ports(NO_ANALOGS);
setup_adc(ADC_OFF);
   setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
setup_comparator(NC_NC_NC_NC);
setup_vref(FALSE);
//  ana program bloğu
while(true)
   {
if(input(PIN_D0))   {
while(input(PIN_D0));   bellekyaz();   }
if(input(PIN_D1))   {
while(input(PIN_D1));   bellekoku();   }
if(input(PIN_D2))   {
while(input(PIN_D2));   klavyeoku();   }
if(input(PIN_D3))   {
while(input(PIN_D3));    karsilastir(); }
   }

}
```

Bu programın aşağıda çalışır halini görmekteyiz.

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen



Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Klavye Kullanımı

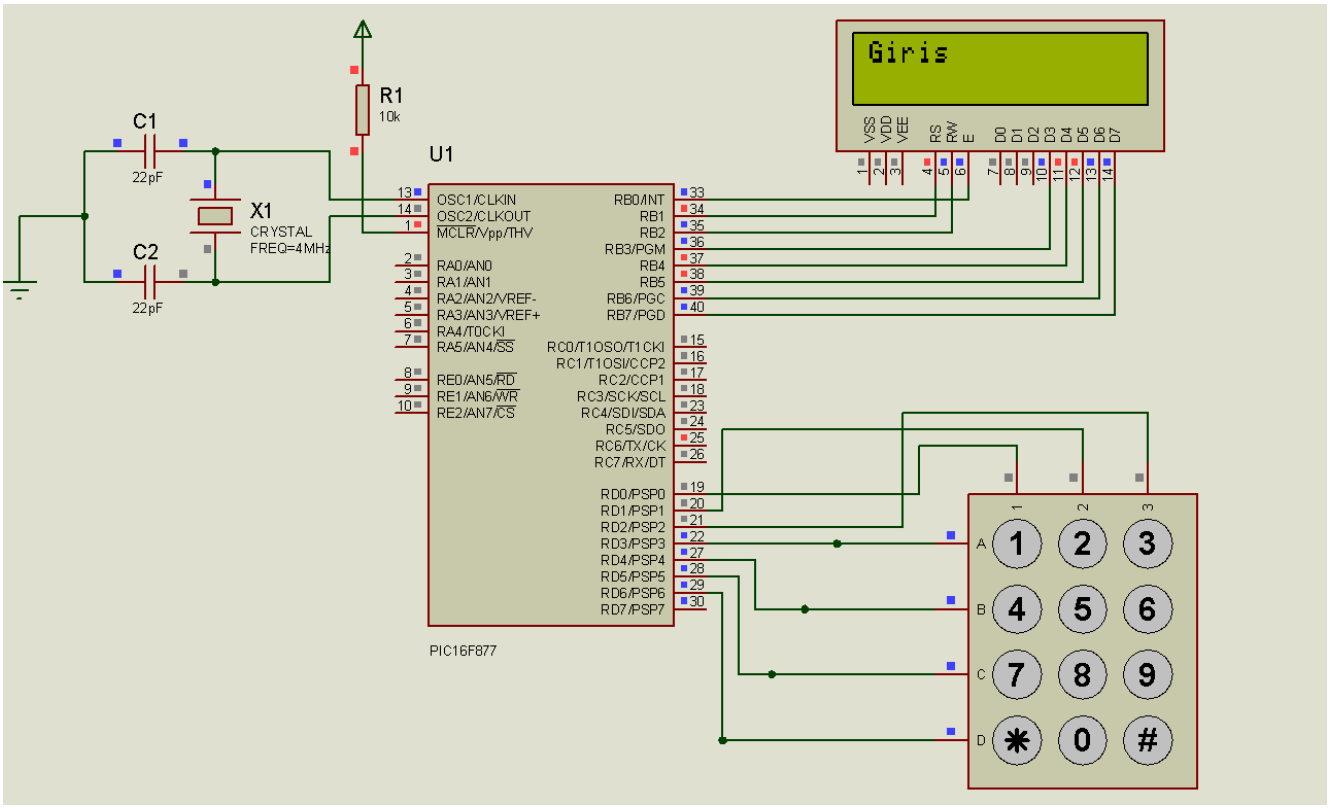
Bir keypad basit bir anahtarlama ve matris kesişiminden oluşur. Satırdan verilen bilgi sütundan alınabilmesi için satır ve sütunun kesişen noktasındaki tuşa basılması gerekmektedir. Kısacası if(şart) ile satır ve sütun aynı anda True olması durumunda sadece bir tuşa basılmıştır. Bunu test ederek hangi tuşa basıldığı bulunur. Bunun için çeşitli tuş kontrol fonksiyonları yazılabilir. Aşağıda keypad_oku() adında fonksiyonla bu işlem gerçekleştirilmiştir.

```
#include"F:\2012-2013-pic\tus\main.h"
#defineuse_portb_lcd TRUE
#define LCD_TYPE 1
#include<lcd.c>

chartus=0; // karakter tipinde değişken
tanımlanıyor
chardrm=0;
//*********** Keypad Tarama Fonksiyonu
*********
charkeypad_oku() // Fonksiyon ismi
{
tus=0;
drm=0;
output_high(pin_d3); // 1. satır lojik-1
yapılıyor
if (input(pin_d0))
{   while(input(pin_d0));tus=1;drm=1; }
if (input(pin_d1))
{  while(input(pin_d1));tus=2;drm=1; }
if (input(pin_d2))
{   while(input(pin_d2));tus=3;drm=1;}
output_low(pin_d3);

output_high(pin_d4); // 2. satır lojik-1
yapılıyor
if (input(pin_d0))
 {   while(input(pin_d0));tus=4;drm=1;}
if (input(pin_d1))
 {   while(input(pin_d1));tus=5;drm=1;}
if (input(pin_d2))
 {  while(input(pin_d2));tus=6;drm=1;}
output_low(pin_d4); // 2. satır lojik-0
yapılıyor

output_high(pin_d5); // 3. satır lojik-1
yapılıyor
if (input(pin_d0))
  {   while(input(pin_d0));tus=7;drm=1;}
if (input(pin_d1))
      {  while(input(pin_d1));tus=8; drm=1;}
if (input(pin_d2))
      {   while(input(pin_d2));tus=9;drm=1;
}
output_low(pin_d5); // 3. satır lojik-0
yapılıyor
output_high(pin_d6); // 3. satır lojik-1
yapılıyor
if (input(pin_d0))
 {   while(input(pin_d0));tus=0xE; drm=1;}
if (input(pin_d1))
 {  while(input(pin_d1));tus=0;drm=1; }
if (input(pin_d2))
 {  while(input(pin_d2));tus=0xF;drm=1; }
output_low(pin_d6); // 3. satır lojik-0
yapılıyor

returndrm; // Fonksiyon "drm" değeri ile
geri döner
}

void main()
{
lcd_init();
output_d(0x00); // D portu çıkışı
sıfırlanıyor
while(true)
        {
/*************** * TUŞUNUNUN KONTROLÜ****/
if((keypad_oku())&&(tus==14))
              {
delay_ms(500);
if((keypad_oku()==1)&&(tus==15))
        {
printf(lcd_putc,"\fDegistir");
//şifre değiştirme fonksiyonuna gidilecek
        }
              }
/************************************/
if((keypad_oku()==1)&&(tus==14))
       {
printf(lcd_putc,"\fGiris");
//şifre giriş edilecek fonksiyona gidilecek
       }
/*********# TUŞUNUN KONTROLÜ YAPILIYOR****/
if((keypad_oku()==1)&&(tus==15))
        {
printf(lcd_putc,"\fKontrol");
//şifre kontrol edilecek fonksiyona
Gidilecek
        }
/************************/
        }
}
```

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Yukarıdaki programın simülasyonda çalıştırılmış hali aşağıdaki şekilde görülmektedir.



Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

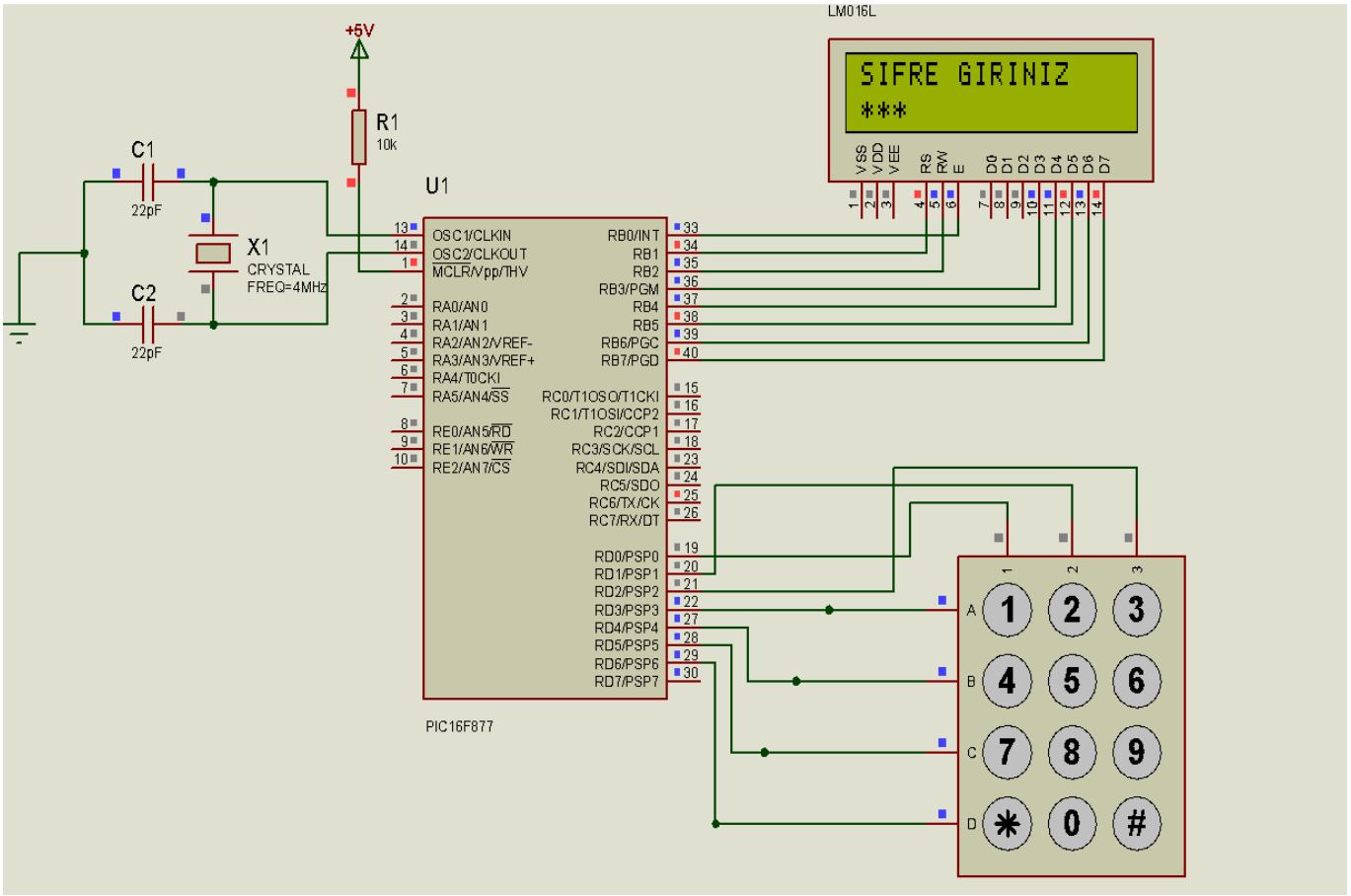
Bellek ve Klavye Kullanımı

Bu programda klavye ve bellek kullanımı yapılmıştır. Yukarıdaki klavye ve bellek kullanım programlarının birleştirilmiş halidir.

```
#include"main.h"
#include<string.h>
#defineuse_portb_lcd TRUE
#define LCD_TYPE 2
#include<lcd.c>

chartus=0; // karakter tipinde değişken tanımlanıyor
chardrm=0;
char klavye[15];
char bellek[15];

//*********** Keypad Tarama Fonksiyonu *********
charkeypad_oku() // Fonksiyon ismi
{
tus=0;
drm=0;
output_high(pin_d3); // 1. satır lojik-1 yapılıyor
if (input(pin_d0))
 {  while(input(pin_d0));tus=1;drm=1; }
if (input(pin_d1))
 {  while(input(pin_d1));tus=2;drm=1; }
if (input(pin_d2))
{   while(input(pin_d2));tus=3;drm=1;}
output_low(pin_d3);

output_high(pin_d4); // 2. satır lojik-1 yapılıyor
if (input(pin_d0))
 {   while(input(pin_d0));tus=4;drm=1;}
if (input(pin_d1))
 {   while(input(pin_d1));tus=5;drm=1;}
if (input(pin_d2))
 {  while(input(pin_d2));tus=6;drm=1;}
output_low(pin_d4); // 2. satır lojik-0 yapılıyor

output_high(pin_d5); // 3. satır lojik-1 yapılıyor
if (input(pin_d0))
 {   while(input(pin_d0));tus=7;drm=1;}
if (input(pin_d1))
 {  while(input(pin_d1));tus=8; drm=1;}
if (input(pin_d2))
 {   while(input(pin_d2));tus=9;drm=1; }
output_low(pin_d5); // 3. satır lojik-0 yapılıyor

output_high(pin_d6); // 3. satır lojik-1 yapılıyor
if (input(pin_d0))
 { delay_ms(1000);tus=0xE; drm=1;}
if (input(pin_d1))
 {  while(input(pin_d1));tus=0;drm=1; }
if (input(pin_d2))
 {   delay_ms(1000);tus=0xF;drm=1; }
output_low(pin_d6); // 3. satır lojik-0 yapılıyor

returndrm; // Fonksiyon "drm" değeri ile geri döner
}
/*************** klavye oku ***************/

voidklavyeoku()
int i;
void main()
{
lcd_init();
        {
int adres=0;
char k;
//  printf(lcd_putc,"\f");
do
        {
if (keypad_oku()&&(tus!=14))
        {
        k=tus;
printf(lcd_putc,"*");
        klavye[adres]=k;
        adres++;
        }
        } while((tus!=15));

        }
/************** bellek yaz ***********/
voidbellekyaz()
        {
int adres=0;
char k;
//  printf(lcd_putc,"\f");
do
        {
if (keypad_oku()&&(tus!=14))
        {
        k=tus;
printf(lcd_putc,"*");
write_eeprom(adres,k);
        adres++;
        }
        } while( tus!=0x0F);
tus=0;
        }
/*************** bellek oku***********/
voidbellekoku()
        {
int adres=0;
char k;
do
        {
        k= read_eeprom(adres);
// printf("%c",k);
       bellek[adres]=k;
        adres++;
        } while(k!=0x0F);
        }
/************* karşılaştır **********/
charkarsilastir()
        {
bellekoku();
if(!strcmp(klavye,bellek)) return 1;
elsereturn 0;
        }
```

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

```
output_d(0x00); // D portu çıkışı sıfırlanıyor
printf(lcd_putc,"\f* BAS SIFRE GIR\n");
while(true)
        {
/*************************************/
if((keypad_oku()))
                {
if((tus==0))
                {
do
                {
if(tus==14) break;
keypad_oku();
                }while((tus!=15));

printf(lcd_putc,"\fMEVCUT SIF.GIR.\n");
klavyeoku();
if(karsilastir())
                {//
printf(lcd_putc,"DOGRU")printf(lcd_putc,"\fYENI
SIFRE\n");
bellekyaz();
printf(lcd_putc,"\fSIFRE KAYDEDILDI.\n");
printf(lcd_putc,"* BAS SIFRE GIR\n");
                }
else
                { printf(lcd_putc,"YANLIS");}
// tus=0;
                }
/*************************************/
if((tus==14))
                {
       printf(lcd_putc,"\fSIFRE GIRINIZ\n");
klavyeoku();
//şifre giriş edilecek fonksiyona gidilecek
                }
/*************************************/
if((tus==15))
                {
printf(lcd_putc,"\fSIFRE..");
if(karsilastir())
                { printf(lcd_putc,"DOGRU\n");
for(i=0;i<15;i++){bellek[i]=0;klavye[i]=0;}
                }
else
                { printf(lcd_putc,"YANLIS\n");}
printf(lcd_putc,"* BAS SIFRE GIR");

//printf(lcd_putc,"\fGiris Yapıldı");
//şifre kontrol edilecek fonksiyona gidilecek
                }

/*************************************/
        }
        }
}
```



Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Adc ve Dac işlemleri

Analog

Analog bilgiler dış dünyadan ölçebildiğimiz büyüklüklerin birçoğu analog bilgidir. Örneğin ses, ısı, nem, uzunluk, basınç ve rüzgâr vb. bir bilginin analog mu yoksa dijital mi olduğunu anlamak için en kolay yol şudur; Eğer ölçülen iki değer arasında üçüncü bir değer söyleyebiliyorsak bu değer analog bilgidir. Örneğin 1gr ile 2gr arasında 1,5gr, 1,1gr vb. Kısacası birçok değer söyleyebiliriz. Bu tür değerlere analog bilgi denilir.

Dijital

Bilgisayar sistemleri dijital bilgilerle çalışırlar. Bu bilgiler 0 ve 1 ile temsil edilir. 0 ile 1 arasında 0,5 gibi değerler temsil edilemez. Bundan dolayı iki değer arasında üçüncü bir değer temsil edilemediğinden dijital bilgidir. Dijital bilgiler genel olarak 2v ve üzeri gerilimler lojik 1 ve 2v altındaki değerler 0 kabul edilir. Bu bilgiler tam ve kesin değildir. Cihaza göre değişebilir. Bu bilgiler genel bilgilerdir.

Çevremizde ölçtüğümüz bilgiler analog bilgilerdir. Biz bunları bilgisayara aktarmak istediğimizde dijitale dönüştürmeliyiz. Fakat birçok değer kaybolacaktır. Örneğin 1gr büyüklüklerde değer dönüşümü yapan dijital dönüştürücü 1.1gr, 1.2gr, 1.3 ……1.9gr değerleri hiç yansıtamayacaktır. Sadece 1gr ve 2gr ölçülebilecektir. İşte bu durumda adc işleminde birçok kayıplar oluşacaktır. Kısacası adc işlemi yapılan bilgi dış dünyadaki bilgiye göre çok kötü bir bilgidir. Fakat artık çok yüksek hassasiyette ölçüm yapan adc'ler bulunmaktadır. Yine de gerçek bilgi olmayacaktır.

Çevrim prensibi

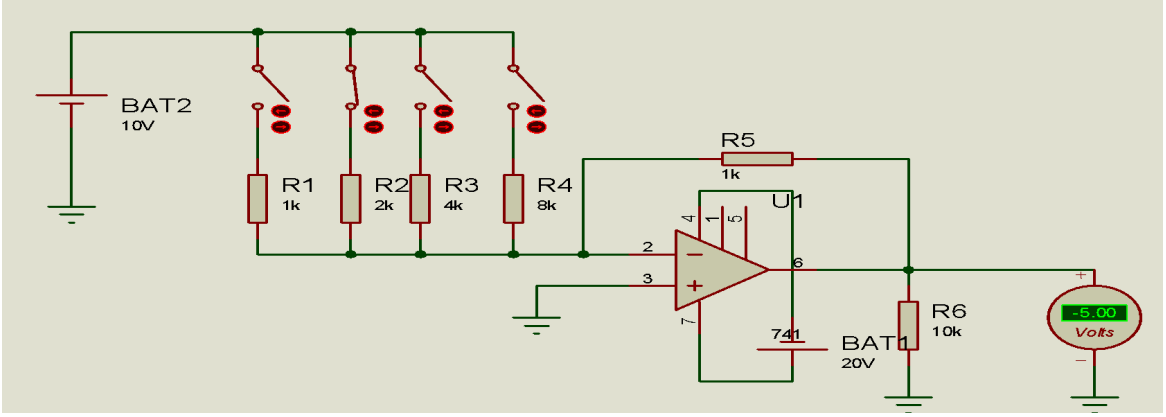
Dış dünyadaki bilgiler analogdur. Biz bunları bilgisayar sistemlerinde kullanmak istersek analogdan dijitale çevirmemiz gerekmektedir. Bu işleme ADC(analog todigitalConvert) denilir. Bunun tam terside mümkündür bu işlem ise DAC(digitalto analog Convert) denilir. Birçok sistemde adc ve dac birimleri bulunur. Örneğin ses bilgisini duyduğumuz hoparlör dac işlemi sonucu ses bilgisini duyuyoruz. Ses bilgisini ise basit bir mikrofonla kayıt etme işlemi ise adc işlemine girer. Bu işlem bizim adımıza ses kartı yapar. Yani ses kartı üzerinde adc ve dac birimleri bulunmaktadır. Sadece adc ve dac sinyallerini işleyen işlemciler üretilmiştir. Bunlara DSP(digitalSignalprocess) dijital sinyal işleyen işlemci denilir.

Dijital Analog Çevirici(DAC)

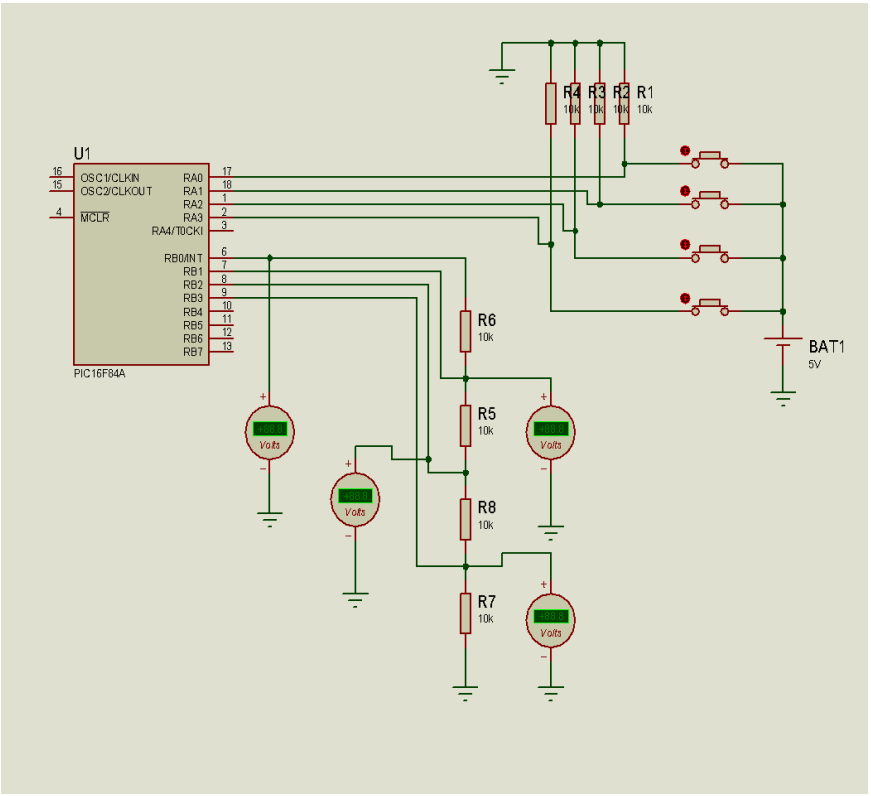
Bilgisayarda bilgiler 1 ve 0 olarak tutulur. Bu bilgileri analog bilgiye çevirmek gerekebilir. Örneğin mp3 dosyasını ses olarak duymamız gerekir. Bir resmi kâğıda basmamız gerekecektir. Bu işlemler DAC işlemidir.

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen



Pic'te basit bir dijital bilginin analog bilgiye çevrilmesi aşağıdaki gibidir.



Bt3	Bt2	Bt1	Bt0	çıkış
0	0	0	0	0mV
0	0	0	1	80mV
0	0	1	0	60mV
0	1	0	0	40mV
1	0	0	0	20mV
0	0	1	1	140mV
0	1	1	1	180mV
1	1	1	1	200mV

Görüldüğü gibi dijital bilgi analog bilgiye çevrilirken özel bir işlemden geçiriliyor.

ADC(Analog toDigitalConvert)

Analog bilgiler dijitale çevrilirken belirli sınırlar içerisinde çevrilir. Örneğin 16f877 10bit çevirme yapabilir.
Bu şu anlama gelir;
Minimum bilgi 00 0000 0000 $0=2^0=0V$
Maksimum bilgi 11 1111 1111 $1023=2^{10}=5Vref$
Buradaki 5V aslında sabit değildir. Pic'e verilen referans gerilimi kadardır. Bu gerilim Vref şeklinde ifade edilir. Fakat hem +Vref hemde –Vref vardır. Buradan şu sonuca varabiliriz.
Ölçülen en küçük birim(ölçek)=+Vref- (-Vref)/çevrilen bit sayısı
Ölçek=maksimum değer/bit sayısı

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Ölçek=5000mV/2^{10} = 4,88~=5mV

10bit	9bit							1bit	0bit	Desimal sayı	giriş
0	0	0	0	0	0	0	0	0	0	0	0mV
0	0	0	0	0	0	0	0	0	1	1	5mV
0	0	0	0	0	0	0	0	1	0	2	10mv
-	-	-	-	-	-	-	-	-	-	-	-
1	1	1	1	1	1	1	1	1	0	1022	4995mV
1	1	1	1	1	1	1	1	1	1	1023	5000mV

Bit sayısı ne kadar fazla olursa o kadar hassasiyet artar. Ölçülen birim küçülür. Buda iyi bir ölçüm yapmayı sağlar.
CCS'deadc işlemi;
`bilgi=read_adc(); komutu ile yapılır. Ayrıntılarını incelememiz gerekmektedir.`
ADC işlemi için iki adet kaydedici kullanılmaktadır. Bunlar ADCON0 ve ADCON1 kaydedicileridir. ADC işlemi yapmadan bu kaydedicilerin içeriği ayarlanmalıdır. Aşağıda içerikleri gösterilmiştir.

ADCON0 Kaydedicisi

7.bit		5			2	1	0.bit
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	-	
ADCS1-0: ADC işleminin ne kadar zaman aralıkla yapılacağını belirleme 00 Fosc/2 01 Fosc/8 10 Fosc/32 11 RC osilatör		8 Adet analog kanaldan hangisinin seçileceğini belirleme yapmada kullanılır. Değiştirilmezse varsayılan Analog 0 kanalı seçilir. 000 Analog 0 kanalı …. 111 Analog 7 kanalı			ADC işleminin yapılıp yapılmadığını ifade eder. 1 ADC başladı 0 ADC işlemi bitti	boş	1 ad açık(aktif) 0 ad kapalı

Analog 0 kanalı için ADCON0=h'41' olmalıdır.

Daha sonrada ADCON0,2 biti kontrol edilmelidir.

ADCON1 Kaydedicisi

7.bit							0.bit
ADFM	-	-	-	PCFG3	PCFG2	PCFG1	PCFG0

Bit-7: **A/D** Sonuç Format Seçme biti dir.
1 olur ise sonuç sağa hizalanmış , **ADRESH** içeriğine yüksek 2 bit 1. Ve 0. Bite alınır diğerleri ise ADRESL içerisine alınır. Son 6 biti sıfır olur.

0 olur ise sonuç sola hizalanmış olur. En yüksek 8 bit ADRESH içerisine, geri kalan iki bit ise **ADRESL** nin7. Ve 6. Bitine yazılır. İlk 6 biti 0 olur.
Bit 6-4 arası kullanılmaz ve 0 olarak okunur.

Bit -3-0 arası **PCFG3** – **PCFG0 A/D** portu ayarlama kontrol bitleridir. İşte bu bitleri ayarlayarak portların seçimleri yapılır. Aşağıdaki tabloya bakınız.

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

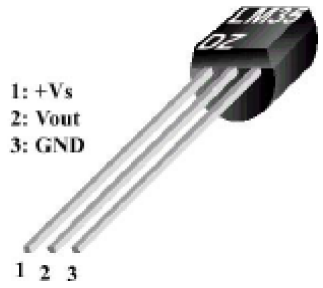
PCFG3: PCFG0	AN7(1) RE2	AN6(1) RE1	AN5(1) RE0	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AN0 RA0	VREF+	VREF-	Kanal/Refs
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	RA3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	RA3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	RA3	VSS	2/1
011x	D	D	D	D	D	D	D	D	-	-	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	RA3	RA2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	RA3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	RA3	RA2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	RA3	RA2	3/
1101	D	D	D	D	VREF+	VREF-	A	A	RA3	RA2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	RA3	RA2	½

Şimdi tablo üzerinde biraz kafa yoralım.

Şayet **PCFG3:PCFG0** bitlerini **0000** olarak verir isek bu durumda **RA0-RA3 , RA5, RE0-RE2** bacaklarının tamamı **ANALOG** olarak ayarlanmış olacak ve artı referans Voltajı VDD den eksi referans voltajı ise VSS yani**GND** den alınacaktır.
Aşağıdaki programda ADCON1=”8E” alındı.

LM 35 DZ Isı Sensörü

LM35: National firması tarafından üretilenLM35 sıcaklık sensörü, analog tipte olup °C derece başına 10 mili volt gerilim üretir. Yani, sıcaklığın her 1°C artışına karşılık, çıkış gerilimi 10mV artar. Örneğin, 20°C için çıkış gerilimi 200mV iken, 100°C için çıkış gerilimi 1V’dur. LM35 sıcaklık sensörünün pek çok tipi vardır. Her bir modelin sıcaklık ölçüm aralığı ve doğruluğu farklıdır. Örneğin, LM35DZ adlı sensör, 0°C ile 100°C arasındaki sıcaklıkları ölçer ve son derece doğrusal bir karakteristiğe sahiptir. Fiyatı ucuz olduğundan ve kolayca temin edilebildiğinden dolayıçoğu sistemde bu sensör tercih edilir. LM35DZ sıcaklık sensörünün bacak bağlantıları Şekil 2’de görülmektedir [2].



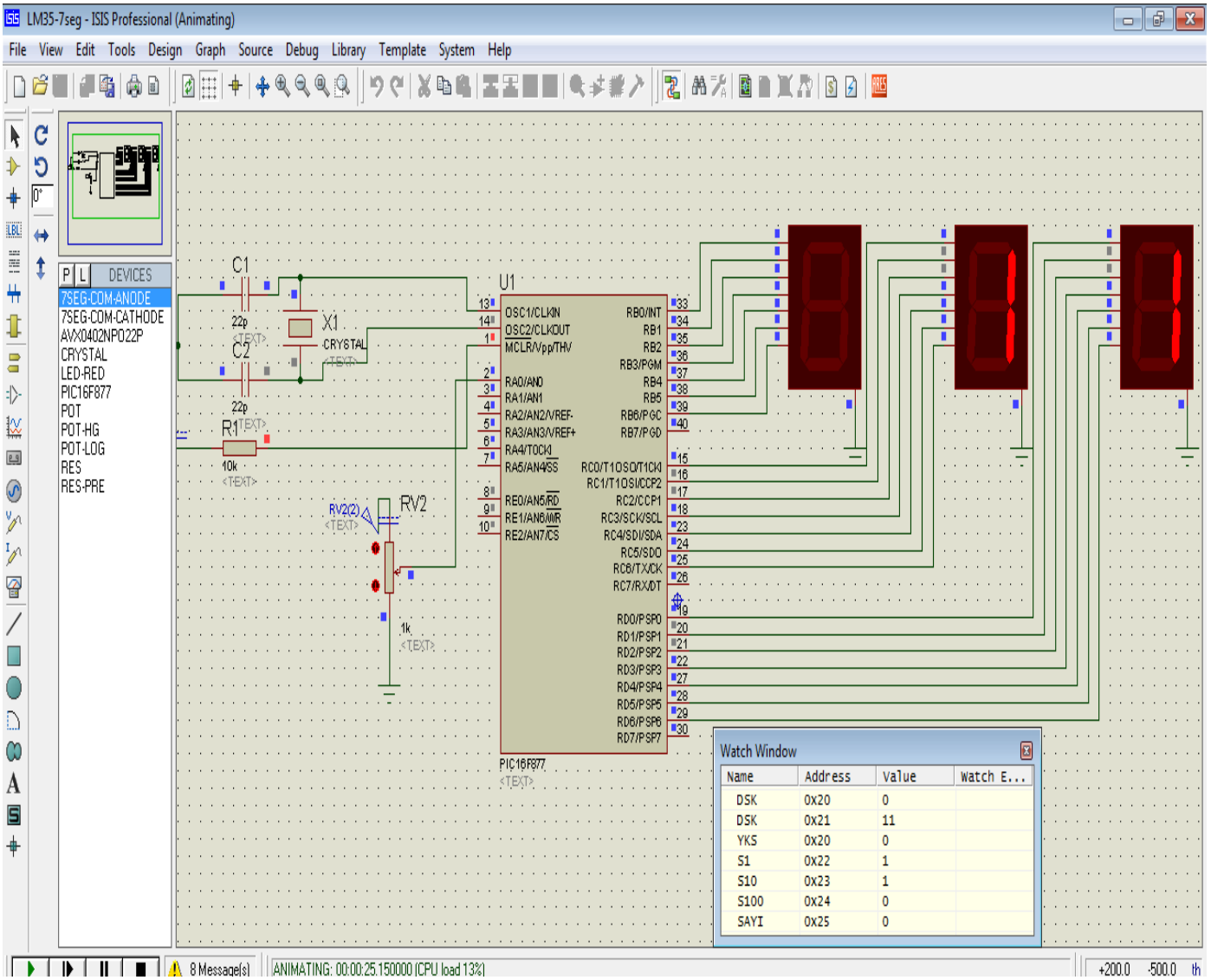
Devre üzerinde bulunan LM35 sensörü dışarıdan algılanan sıcakdeğerini doğrudan kullanamaz. Çünkü bu sensor her 1dereceye karşılık 10mVanalog sinyal üretir. Bu üretilen sinyal PIC mikro denetleyicinin A0 portundan alınır (A veya E portu olabilir). Bu alınan değer PIC 16F877 içerisinde bulunan ADC devresi tarafından sayısala dönüştürülür. Sayısala çevirme için bir referans gerilimine ihtiyaç duyulur. Bu gerilim PIC16F877 veya PIC16f628 kullanılarak yada harici bir kaynaktan sağlanabilir. Harici kaynak kullanıldığında katsayı hesaplanmasında dikkatli olunmalıdır.

PIC16F877 veya PIC16f628 ADC işlemi yapmasından dolayı çok tercih edilen mikrokontrollerdir.

Lm35 ile adc işlemi yapma

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen



Bu programda üç basamaklı sayı binary(ikilik) olarak alınıp, desimal(onlu) sayıya çevrilmiştir. Bunun için basamakla alt programı kullanılmıştır. Ayrıca adc işleminden okunan sayı küçük hassasiyet hatası da olsa basit bir yöntemle bol adındaki alt programla sayı bir kaydırılarak değer bulunmuştur. Programda birçok işlem aynı anda yapılmıştır.*Burada dikkat edilmesi gereken durum şudur yaklaşık 5mV 1 sayısına karşılık gelmektedir. Bizim için 1 derece 10mV karşılık gelmektedir. Yani 10mV için 2 sayısı karşılık gelir. Bizim ölçtüğümüz değer iki kat olacaktır. Yani 20ºC için 40 değerini okuruz. Bunu ise bir defa sağa kaydırma işlemi ile düzeltebiliriz.*

Bunlar;

- İkilik sayının desimal(onlu) sayıya çevrilmesi
- Üç adet 7 parçalı ekran kullanma
- Sayılar için çevrim tablosunun kullanılması
- Gecikme alt programı
- ADC işlemi yapma
- ADC işleminin sonucunu okuma

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

```
        INCLUDE "P16F877.INC"

        YKS     EQU     H'20'
        DSK     EQU     H'21'
        S1      EQU     H'22'
        S10     EQU     H'23'
        S100    EQU     H'24'
        SAYI    EQU     H'25'
        SAYAC1  EQU     H'26'
        SAYAC2  EQU     H'27'
        SAYAC3  EQU     H'28'

;*****************************
        BSF     STATUS,5
        CLRF    TRISB
        CLRF    TRISC
        CLRF    TRISD
        MOVLW   H'01'
        MOVWF   TRISA
        MOVLW   H'8E'
        MOVWF   ADCON1
        BCF     STATUS,5
        MOVLW   H'41'
        MOVWF   ADCON0
CEVIR
        BSF     ADCON0,2
BEKLE
        BTFSC   ADCON0,2
        GOTO    BEKLE

        MOVF    ADRESH,0
        MOVWF   YKS
        BSF     STATUS,5
        MOVF    ADRESL,0
        BCF     STATUS,5
        MOVWF   DSK

CALL    BOL
        CALL    BASAMAK

;       MOVF    DSK,0
;       MOVWF   PORTB
;*****************1 LER BASAMAGI
        MOVF    S1,0
        CALL    TABLO
        MOVWF   PORTD
;*****************10 LAR BASAMAĞI
        MOVF    S10,0
        CALL    TABLO
        MOVWF   PORTC
        CALL    ZAMAN
        GOTO    CEVIR
;****************************** SAYIYI 2 YE BOL
BOL
        BCF     STATUS,0
        RRF     DSK,1
        RETURN
```

```
;************************ SAYIYI BASAMKLARA AYIR
BASAMAK
        CLRF    S1
        CLRF    S10
        CLRF    S100
        MOVF    DSK,0
;       MOVLW   D'247'
        MOVWF   SAYI
;************************************
TEKRAR
;************************************
        INCF    S1,1
        MOVLW   d'10'
        SUBWF   S1,0
        BTFSS   STATUS,2
        GOTO    DEVAM
;*****************************
        CLRF    S1
        INCF    S10,1
        MOVLW   D'10'
        SUBWF   S10,0
        BTFSS   STATUS,2
        GOTO DEVAM
;*****************************
        CLRF    S10
        INCF    S100
DEVAM
;*****************************************
        DECFSZ  SAYI,1
        GOTO    TEKRAR
;*****************************************
        RETURN

;********TABLO*********

TABLO
        ADDWF   PCL,1;
        RETLW   3FH;0
        RETLW   06H;1
        RETLW   5BH;2
        RETLW   4FH;3
        RETLW   66H;4
        RETLW   6DH;5
        RETLW   7DH;6
        RETLW   07H;7
        RETLW   7FH;8
        RETLW   6FH;9
        RETLW   77H;A
        RETLW   7CH;B
        RETLW   39H;C
        RETLW   5EH;D
        RETLW   79H;E
        RETLW   71H;F

;*****************************
ZAMAN   MOVLW   D'200'
                MOVWF   SAYAC1
TIMER1  MOVLW   D'100'
                MOVWF   SAYAC2
TIMER2
                MOVLW   D'100'
                MOVWF   SAYAC3
TIMER3
                DECFSZ  SAYAC3,F
                GOTO    TIMER3
                DECFSZ  SAYAC2,F
                GOTO    TIMER2
                DECFSZ  SAYAC1,F
                GOTO    TIMER1
                RETURN
END
```

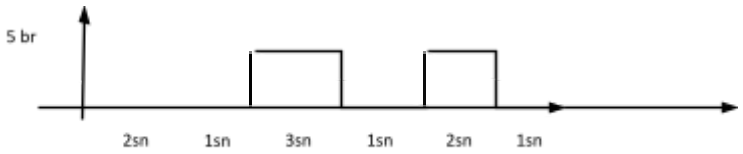
Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

PWM(pulse width modulation)darbe genişlik modülasyonu(değişiklik)

Pic'de çıkış 1 ya da 0'dır. Fakat 1 ya da 0 olma durumunu pic ile kontrol etmek mümkündür. Bu işleme pwm denilir. Aslında pwm ortalama iş miktarını verir. Aslında yapılan çıkışın 0'da kalma süresini ya da 1'de kalma süresini değiştirmedir. Bu işlem zaman programına müdahale ile olur.

Çıkış şöyle bulunur;
Vçık=(1 olma zamanı*5V+0 olma zamanı*0) /(1olma zamanı+0 olma zamanı)

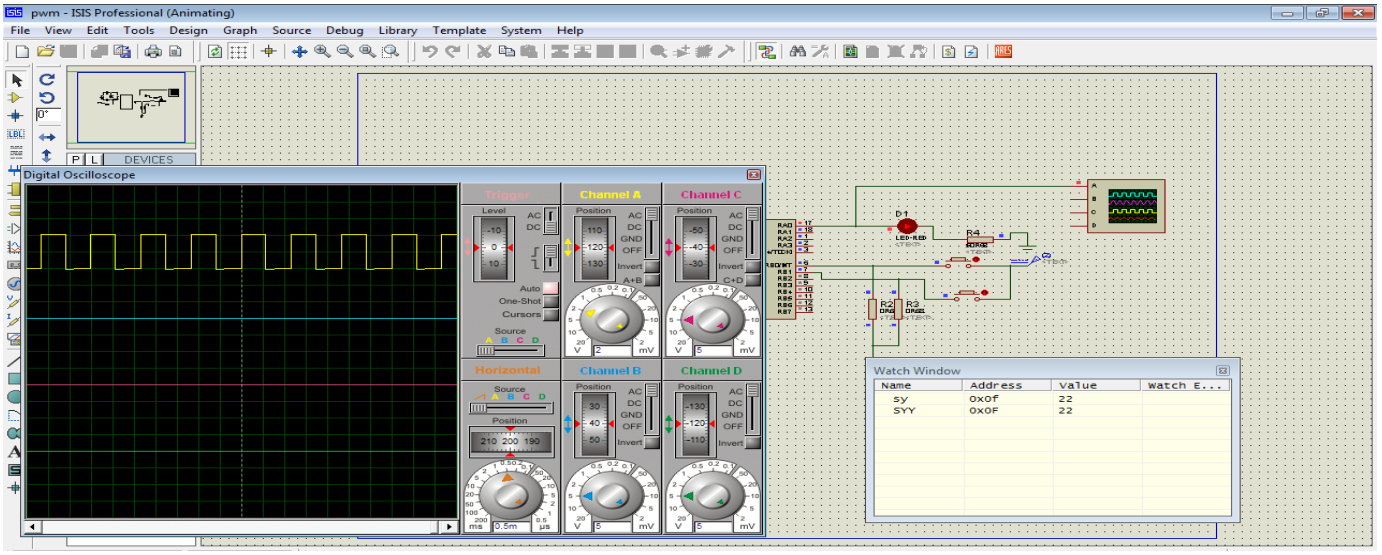


İş=(5br*2sn+5br*3sn+5br*2sn)/10=3,5br

Örnek:

İş yapma süresi 5sn ve toplam iş süresi 20sn, iş büyüklüğü 5br ise ortalama iş nedir?

İŞ=5sn*5br+15sn*0/20sn=1,25br'dir



Aşağıdaki programda zamanlama alt programının süresi artırılıp azaltılmıştır.

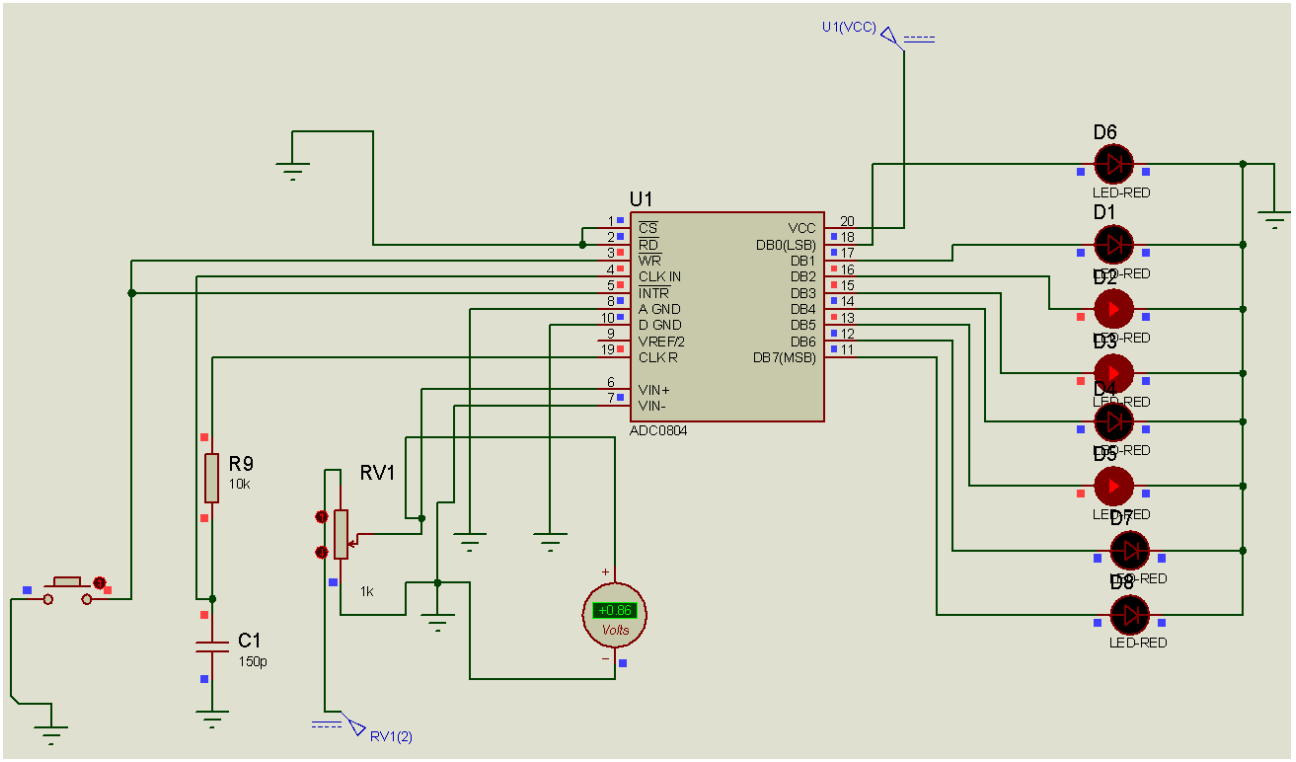
```
LIST        P=16F84
STATUS EQU  03H
PORTA EQU   05H
PORTB EQU   06H
TRISB EQU   86H
TRISA EQU   85H
SAYAC1      EQU  H'0C'
SAYAC2      EQU  H'0D'
SAYAC3      EQU  H'0E'
SAYI        EQU  H'0F'
            CLRF SAYI
            BSF  STATUS, 5 ;BANK1'e geçilir
            MOVLW        H'FF'
            MOVWF        TRISB ;PORTB tüm uçlar giriş olacaktır
            CLRF TRISA ;PORTA tüm uçlar çıkış olacaktır
            BCF  STATUS, 5 ;BANK0'a geçilir
            CLRF PORTA

BASLA
;*****************************************************
'           BSF          PORTA,0
            CALL ZAMAN
            CALL ZAMAN
            BCF          PORTA,0
            CALL ZAMAN
            CALL ZAMAN
BT1
            BTFSS PORTB,0 ;PORTB 0. bitini test et
            GOTO BT2
BT11
            BTFSC PORTB,0
            GOTO BT11
;           CALL ZAMAN
            INCF SAYI,1
            GOTO BASLA

;*****************************************************
BT2         BTFSS PORTB,1 ;PORTB 0. bitini test et
            GOTO BASLA
BT22
            BTFSC PORTB,1
            GOTO BT22
            DECF SAYI,1

            GOTO BASLA

GOTO BASLA ;Başa dön
;*****************************************************
;*****************************************************
ZAMAN

            MOVF    SAYI,0
            MOVWF   SAYAC1
TIMER1
            DECFSZ  SAYAC1,1
            GOTO    TIMER1

            RETURN
;*********************************
```

	END ;Program sonu.

CCs'depwm;

Mikroişlemci Uyumlu ADC'ler

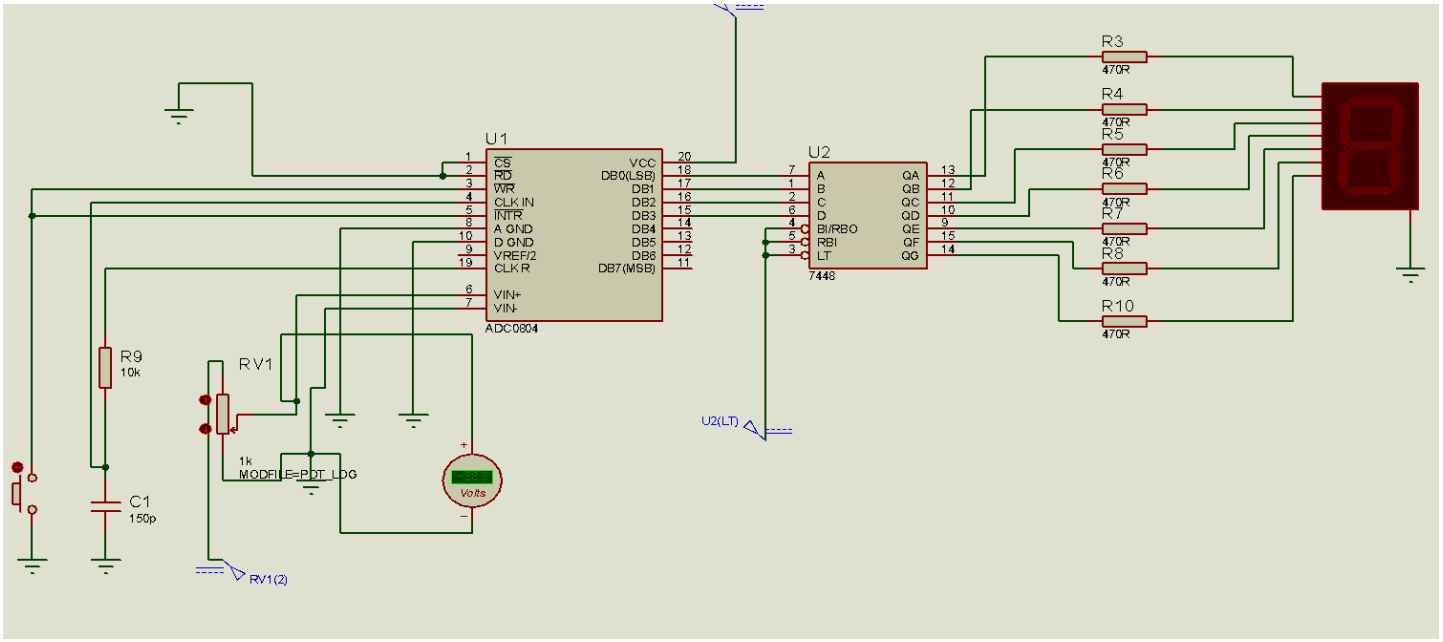
Adc804 entegresi adc işlemi yapan bir entegredir. Bu entegre 8bit adc işlemi yapar. Aşağıda şekli görülmektedir. Bu entegre mcu uyumlu bir entegredir. Bağlı olan buton yerine bir mcu'ya bağlanıp kullanılabilir. Bazı durumlarda harici adc entegresini kullanmak gerekirse rahatlıkla adc804 kullanılabilir. Fakat pic ailesinde 16f8xx serilerinde bulunmaktadır. Ayrıca pwm için 16f6xx serileri(16f628) kullanılabilir.



Aşağıda 7 parçalı ekran bağlı adc804 kullanılmıştır.

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen



Step Motor

Step motorlar dijital pals ile çalışan, temel mantık olarak fırçasız motorlardır. İçerisinde bobinler sayesinde manyetik alan oluşturularak rotor hareketini sağlayan motor türleridir. Bu motorda ufak hatalar olsa da genelde motor hareketi sabit ve doğrudur. Adım miktarı üretici firma tarafından verilir. Daha çok FDD, CD ve dvd sürücülerinde bulunur.

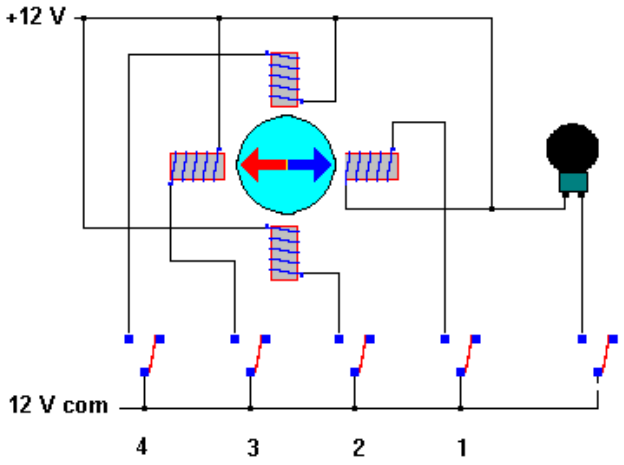
Avantajları;

- Pic ile kontrol edilebilir.
- Durma ve başlama hızında gecikme olmaz.
- İstenilen pozisyona rahatlıkla getirilebilir.

Dezavantajları;

- Maliyeti ve imalatı yüksektir.

İç yapısı;



Step motor hareket tablosu

4.	3.	2.	1. bobin	Adım
0	0	0	1	1.adım
0	0	1	0	2.adım
0	1	0	0	3.adım
1	0	0	0	4.adım

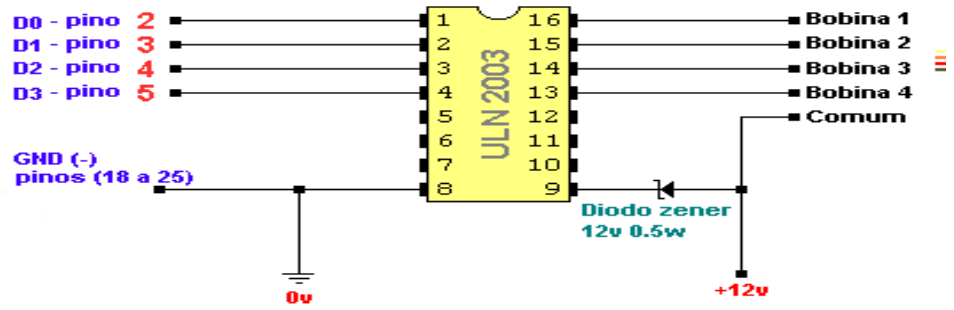
Dalga sürme tablosu				
4.	3.	2.	1.	Adım
0	0	0	1	1.adım
0	0	1	1	
0	0	1	0	2.adım

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

0	1	1	0	
0	1	0	0	3.adım
1	1	0	0	
1	0	0	0	4. adım
1	0	0	1	

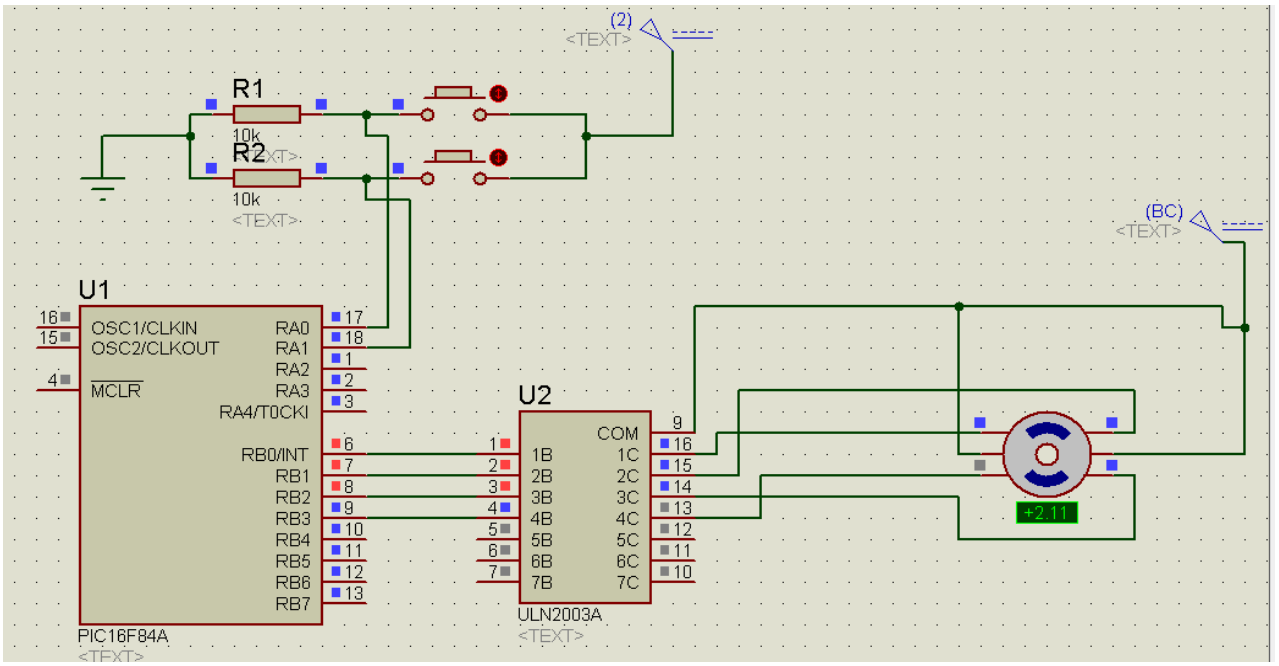
Motor Sürme Entegresi

Mikrodenetleyicilerde 20mA'lik çıkış alınabilmektedir. Fakat kontrol edeceğimiz cihaz röle vb. bir şeyse transistörle yükseltme yapmamız gerekmektedir. Fakat bu işlem çoğu zaman devreyi karmaşıklaştırmaktadır. Bu gibi durumlar step motorlar içinde geçerlidir. Doğrudan step motor kontrol edilemez. Bu problemi çözmek için ULN serisi entegreler kullanılır. Çıkış gerilimleri yapılarında darligtontaransistör barındıran uln entegrelerde yaklaşık 500mA'dir. Şekli aşağıda görülmektedir.



Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Örnek:Tam tur döndürme ile motor ileri-geri çalıştırma



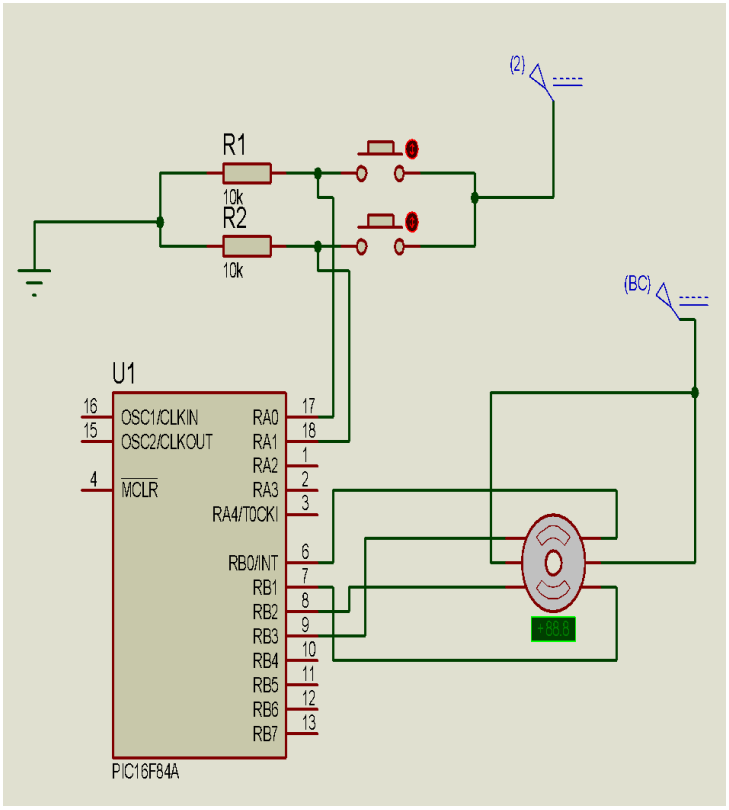

```
LIST P=16F84
PCL   EQU h'02'
STATUS EQU h'03'
PORTA EQU h'05'
PORTB EQU h'06'
TRISA EQU h'85'
TRISB EQU h'86'
STEP EQU h'20'
SAYAC1     EQU   H'0C'
SAYAC2     EQU   H'0D'
SAYAC3     EQU   H'0E'
SAYI  EQU   H'0F'

ADIM  EQU   H'1C'
           CLRF PORTA
           CLRF PORTB
           BSF STATUS, 5 ; Bank1
           MOVLW B'00000011' ;
PORTA 0. ve 1. bit giriş
           MOVWF TRISA
           CLRF TRISB
           BCF STATUS, 5 ; Bank0
           MOVLW h'00'
           MOVWF STEP
MAINPROG
BT1        BTFSS PORTA, 0
           GOTO  BT2
           CALL  ILERI
BT2        BTFSS PORTA, 1
           GOTO  BT1
           CALL GERI
           GOTO MAINPROG
```

```
ILERI
           MOVLW B'00001110'
           MOVWF PORTB
           CALL  ZAMAN
           MOVLW B'00001101'
           MOVWF PORTB
           CALL  ZAMAN
           MOVLW B'00001011'
           MOVWF PORTB
           CALL  ZAMAN
           MOVLW B'00000111'
           MOVWF PORTB
           CALL  ZAMAN
           RETURN

GERI
           MOVLW B'00000111'
           MOVWF PORTB
           CALL  ZAMAN
           MOVLW B'00001011'
           MOVWF PORTB
           CALL  ZAMAN
           MOVLW B'00001101'
           MOVWF PORTB
           CALL  ZAMAN
           MOVLW B'00001110'
           MOVWF PORTB
           CALL  ZAMAN

           RETURN
           END
```

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

CCS'de step motor kontrol kodu




```
#include"F:\stp-motor-ccs\main.h"

int8 sure;
void main()
{


setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
sure=100;
while(true)
{

if (input(PIN_A0)==1)
{
output_b(0xA);
delay_ms(sure);
output_b(0X6);
delay_ms(sure);
output_b(0X5);
delay_ms(sure);
output_b(0X9);
delay_ms(sure);

}


if (input(PIN_A1)==1)
{
output_b(0b00000111);
delay_ms(sure);
output_b(0b00001011);
delay_ms(sure);
output_b(0b00001101);
delay_ms(sure);
output_b(0b00001110);
delay_ms(sure);
output_b(0b00000111);
delay_ms(sure);
}

}


}
```

Sistem Kontrol Uygulamaları Notları

Bayram KARAHAN
Bilişim Teknolojileri Öğretmen

Kaynaklar

[1] Bil386 Mikrobilgisayarlı Sistem Tasarımı
[2] http://www.biltek.tubitak.gov.tr/gelisim/elektronik/dosyalar/5/5.pdf (Haziran 2007)
[3] www.datasheetcatalog.com/datasheets_pdf/L/M/3/LM35DZ.shtml (Mayıs 2007)
[4] www.microchip.com