T.C.
MİLLÎ EĞİTİM BAKANLIĞI

ELEKTRİK ELEKTRONİK TEKNOLOJİSİ

MİKRODENETLEYİCİYLE ANALOG İŞLEMLER

523EO0022

Ankara, 2012

- Bu modül, mesleki ve teknik eğitim okul/kurumlarında uygulanan Çerçeve Öğretim Programlarında yer alan yeterlikleri kazandırmaya yönelik olarak öğrencilere rehberlik etmek amacıyla hazırlanmış bireysel öğrenme materyalidir.
- Millî Eğitim Bakanlığınca ücretsiz olarak verilmiştir.
- PARA İLE SATILMAZ.

İÇİNDEKİLER

AÇIKLAMALAR

KOD	**523EO0022**
ALAN	**Elektrik-Elektronik Teknolojisi**
DAL/MESLEK	**Dal Ortak**
MODÜLÜN ADI	**Mikrodenetleyici ile Analog İşlemler**
MODÜLÜN TANIMI	Bu modül mikrodenetleyiciler ile analog uygulama devreleri yapılması ile ilgili bilgi ve becerilerin kazandırıldığı bir öğrenme materyalidir.
SÜRE	40/24
ÖN KOŞUL	Bu modülün ön koşulu yoktur.
YETERLİK	Mikrodenetleyici ile analog işlemleri yapmak
MODÜLÜN AMACI	**Genel Amaç** Bu modül ile gerekli ortam sağlandığında, mikrodenetleyici ile analog işlemler yapabileceksiniz. **Amaçlar** **1.** Mikrodenetleyici ile ADC/DAC dönüştürme yapabileceksiniz. **2.** Mikrodenetleyici ile analog uygulama devreleri yapabileceksiniz. **3.** Picbasic pro programı ile programlama yapabileceksiniz. **4.** Picbasic pro programı ile uygulama devreleri yapabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	**Ortam:** Elektrik-elektronik laboratuvarı, işletme, kütüphane, ev, bilgi teknolojileri ortamı vb. **Donanım:** Bilgisayar, projeksiyon cihazı, çizim ve simülasyon programları, kataloglar, deney setleri, çalışma masası, avometre, bread board, eğitmen bilgi sayfası, havya, lehim, elektrikli almaçlar, anahtarlama elemanları, yardımcı elektronik devre elemanları, elektrik-elektronik el takımları
ÖLÇME VE DEĞERLENDİRME	Modül içinde yer alan her öğrenme faaliyetinden sonra verilen ölçme araçları ile kendinizi değerlendireceksiniz. Öğretmen modül sonunda ölçme aracı (çoktan seçmeli test, doğru-yanlış testi, boşluk doldurma, eşleştirme vb.) kullanarak modül uygulamaları ile kazandığınız bilgi ve becerileri ölçerek sizi değerlendirecektir.

GİRİŞ

Sevgili Öğrenci,

Günümüzde pek çok iş, yapısında mikroişlemci bulunan makineler ile gerçekleştirilir. Mikroişlemci (CPU) ile kontrol edilen sistemlerde bunun dışında RAM, Bios, I/0 ünitesi gibi ek birimlere ihtiyaç duyulur. Bu şekilde hem maliyet artar hem de işlemler zorlaşır. Bütün bu olumsuzlukları ortadan kaldıran devre elemanı mikrodenetleyicidir. Pek çok firma mikrodenetleyici üretmektedir fakat Microchip firmasının PIC (Pripheral Interface Controller -çevre birimlerini kontrol eden ünite) adını verdiği denetleyici uygulamalarda geniş yer bulmaktadır.

Bu modüldeki birkaç uygulamada PIC 16F84 kullanılmıştır. 16F84 entegresi analog işlemler için sınırlı olduğundan dolayı çoğu uygulamada 16F877 entegresi tercih edilmiştir.

Modüldeki uygulamaları takip ederek analog devreler geliştirebilir ve programlayabilirsiniz.

Her uygulamada ayrı bir konu anlatılmaktadır. Uygulamalar basitten karmaşığa doğru sıralanmış ve sizin anlayabileceğiniz sadelikte işlenmiştir.

Modülün amacı, programlama mantığını öğrenerek elektronik devre uygulamalarını mikrodenetleyiciyle çabuk, doğru ve kolayca çözme yeteneğini kazandırmaktır.

Bu modülün sonunda program yazabilecek, bu programa ait devreyi yapabilecek ayrıca kendi program ve devrelerinizi üretme yeteneğine sahip olabileceksiniz.

ÖĞRENME FAALİYETİ–1

AMAÇ

Mikrodenetleyici ile ADC/DAC dönüştürme yapabileceksiniz.

ARAŞTIRMA

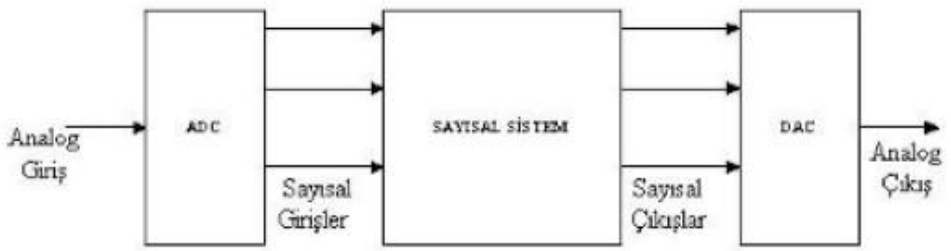
- Analog veriyi öğrenmelisiniz.
- ADC ve DAC devre çeşitleri ile ilgili bilgi edinmelisiniz.
- ADC ve DAC işlemlerinde niçin mikrodenetleyicinin kullanıldığını araştırmalısınız.
- Araştırma işlemleri için Mikrodenetleyici ile Dijital İşlemler modülünü gözden geçirebilir ve internet ortamından yararlanabilirsiniz.

1. ANALOG VERİ

1.1. Genel Bilgiler

Sıfırdan sonsuza kadar devamlı olarak değişim gösteren büyüklük **analog büyüklük** olarak tanımlanır. Bilindiği gibi görülen ve duyulan büyüklüklerin tamamı, analog bilgi tipindedir. Çünkü bu değerler sürekli değişmektedir.

Fiziksel bir büyüklük bilgi şekline dönüştürülürken bilgiyi temsil eden işaret doğrudan doğruya fiziksel büyüklüğün benzeri ise oluşan işaret **analog işaret** olarak adlandırılır. Bu analog işaretlerin algılanması ve değerlendirilmesi, ancak insanoğlu tarafından mümkünken bilgisayarlar ve mikroişlemciler tarafından mümkün değildir. Dijital sistemlerin dış dünya ile bağlantılarını sağlamak için ölçülen fiziksel büyüklüklerin dijital sistemin anlayabileceği sayısal değerlere dönüştürülmeleri gerekir. Analog bilgiyi sayısal değerlere dönüştüren elemanlara **analog dijital çevirici** (ADC Analog Dijital Converter) adı verilir.

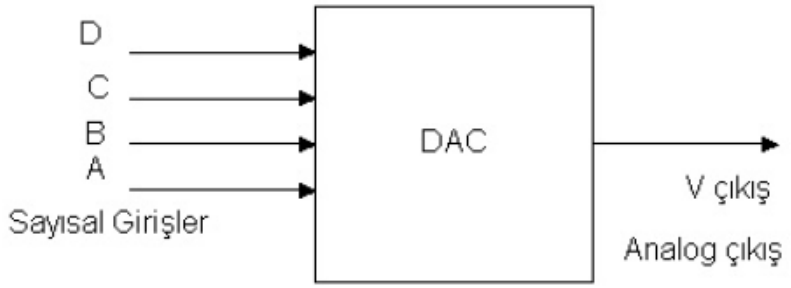


Şekil 1.1: Sayısal bir sistemin blok diyagramı

Dijital bir bilginin analog işaretlere çevirmesi için kullanılan elemanlara dijital analog çevirici (DAC Dijital Analog Converter) adı verilir. Birçok elektronik cihaz sayısal sistemin giriş ve çıkışına bağlanarak kullanılır. Bu cihazalar sayısal sistemlerde A/D ve D/A dönüştürücüler ile birlikte kullanılabilir. Şekil 1.1'de bu sistemlerin blok diyagramı gösterilmiştir.

1.2. Dijital Analog Çeviriciler

Dijital bilgi sinyalini, sayısal değerine orantılı olarak gerilim veya akıma dönüştüren devrelere dijital analog çevirici denir. Bu gerilim veya akım, girişteki değerlere göre değişen bir analog sinyaldir. Şekil 1.2'de 4 bit girişli DAC'ın blok diyagramı görülmektedir.



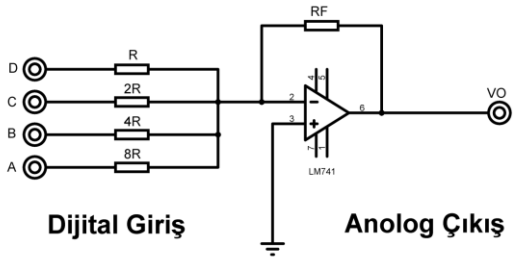
Şekil 1.2: DAC sistemin blok diyagramı

D/A çevirme işlemlerinde genellikle ağırlık dirençli DAC, R-2R merdiven tipi DAC veya PWM (PulseWidth Modulation - Darbe genişlikli modülasyon) metodu kullanılır.

1.2.1. Ağırlık Dirençli DAC Devresi

Şekil 1.3'te D/A çeviricinin basit bir devresi görülmektedir. Devrede OP-AMP' toplayıcı olarak kullanılmaktadır. D, C, B, A dijital girişlerin ağırlıklarının toplamı kadar çıkışta gerilim elde edilir.

Çıkış gerilimi , $V\text{çk} = -\frac{\text{Rf}}{8\text{R}}\ \text{Vref}\ (\text{A} + 2\text{B} + 4\text{C} + 8\text{D})$ olarak bulunur.

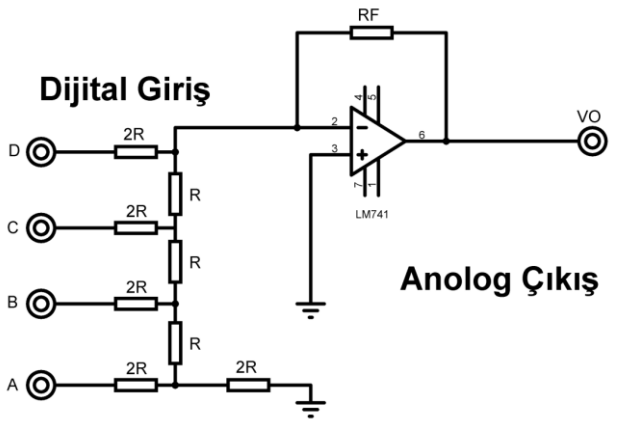


Şekil 1.3: Ağırlık dirençli DAC devresi

1.2.2. R-2R Merdiven Tipi DAC Devresi

Dijital bilginin analog bilgiye çevrilmesinde en fazla kullanılan yöntemdir. R- 2R merdiven tipi devresi Şekil 1.4'te gösterilmiştir. Çıkış gerilimi aşağıdaki gibi hesaplanır.

$$Vçk = -\frac{\mathrm{Rf}}{16\mathrm{R}}\,\mathrm{Vref}\,(\mathrm{A} + 2\mathrm{B} + 4\mathrm{C} + 8\mathrm{D})$$



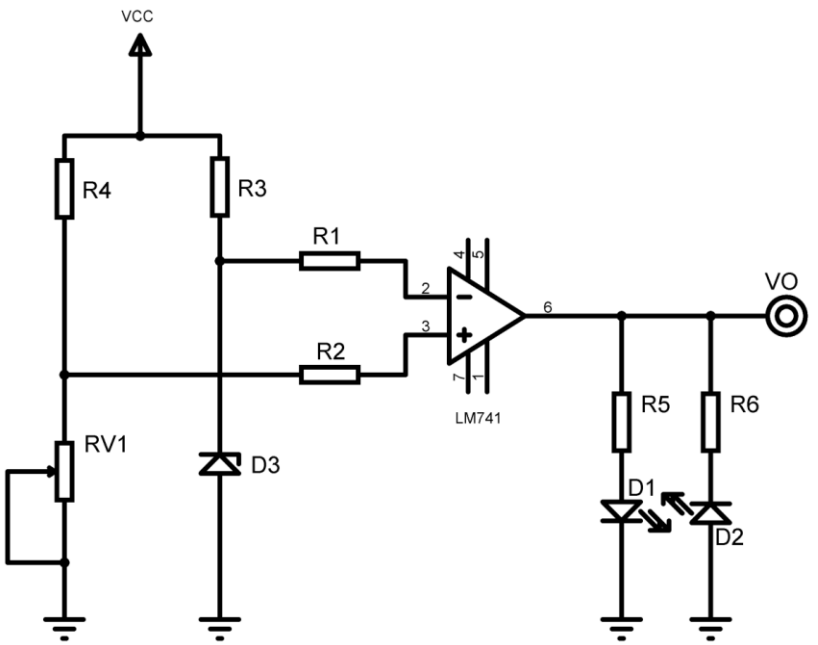
Şekil 1.4: R-2R DAC devresi

1.3. Analog Dijital Çeviriciler

Analog dijital çeviriciler analog giriş gerilimini alır ve belirli bir süre içinde analog giriş değerini gösteren sayısal çıkış kodu üretir. Şekil 1.5'te analog dijital sistemin blok diyagramı gösterilmiştir.

Şekil 1.5: ADC sistemin blok diyagramı

Basınç, sıcaklık gibi fiziksel değişimi veya akım, gerilim gibi analog sinyalleri mikrodenetleyici sistemlere direk uygulanmadığından A/D çeviriciler kullanılır. ADC devrelerinin temeli karşılaştırıcıya dayanır. Karşılaştırıcı devresinde OP-AMP'ın girişlerinden birine referans gerilimi diğerine ise giriş gerilimi uygulanır. Şekil 1.6'da OPAMP'lı karşılaştırıcı devresi görülmektedir.



Şekil 1.6: OP-AMP' la yapılan karşılaştırıcı devresi

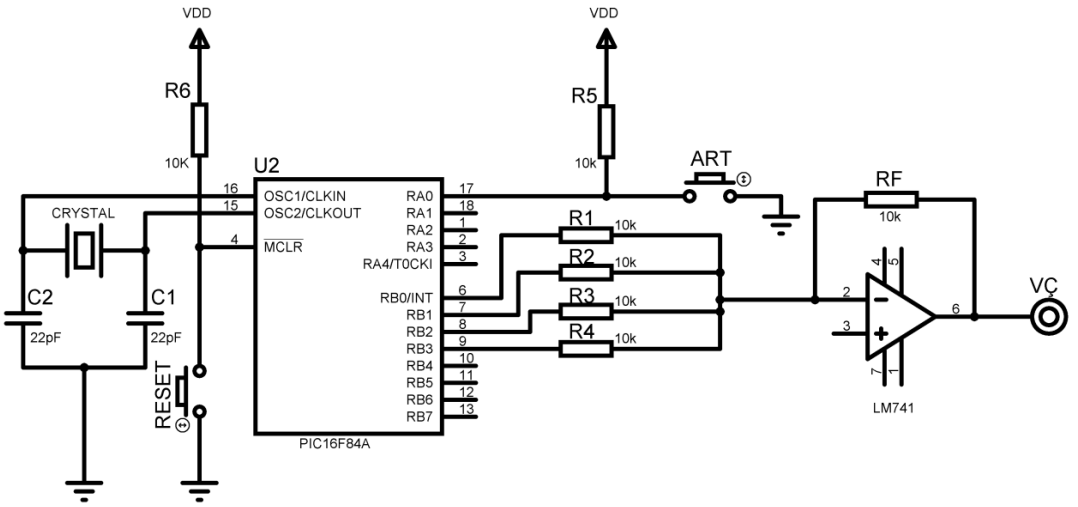
Şekil 1.6'daki devrede zener gerilimi referans gerilimini belirler. Referans gerilimi OP-AMP'ın "+" girişidir. Giriş gerilimini potansiyometre 0-12V arasında değiştirir. Giriş gerilimi OP-AMP'ın "-" girişidir. Giriş gerilimi referans geriliminden büyük olursa çıkış -V olur ve D3 LED'i yanar. Giriş gerilimi referans geriliminden küçük olursa çıkış +V olur ve D2 LED'i yanar. İki gerilim eşit olursa çıkış 0V olur.

1.4. PIC ile DAC Uygulama Devreleri

Bu uygulamalarda kullanılan PIC 16F84'ün analog girişi bulunmadığından bu işlemler DAC veya ADC devreleri yardımıyla yapılır. Şekil 1.1'de bulunan blok diyagramda bu yapı gösterilmiştir.

1.4.1. Ağırlık Dirençle DAC Uygulama Devresi

PIC ile yapılan uygulamada A portunun 0. bitine bağlı olan bir butona her basıldığında çıkıştaki gerilimi artıran dijital analog çevirici uygulamasıdır. Dijital analog çevirici olarak ağırlık dirençli DAC devresi kullanılmıştır. Şekil 1.7'de uygulama devresi gösterilmiştir.



Şekil 1.7: PIC 16F84A ile DAC uygulama devresi

```
;========DAC UYGULAMA PROGRAMI=============

        LIST P=16F84
        INCLUDE "P16F84.INC"

SAYAC1 EQU H'0C'        ; Gecikme alt programlarında kullanılan değişken
SAYAC2 EQU H'0D'        ; Gecikme alt programlarında kullanılan değişken
SAYAC3 EQU H'0E'        ; Gecikme alt programlarında kullanılan değişken
ART    EQU H'0F'        ; programlarda kullanılan değişken

:Portları Ayarla……
        CLRF    PORTB           ;PORTB' yi temizle
        BSF     STATUS,5        ;BANK1' e geç
        CLRF    TRISB           ;PORTB çıkış
        MOVLW   H'FF'           ;W <-- H'FF'
        MOVWF   TRISA           ;PORTA giriş
        BCF     STATUS,5        ;BANK0 a geç

; Start butonuna basılıncaya kadar bekle……
BUTON
        BTFSC   PORTA,0         ;PORTA nın 1.biti 0 mı?
        GOTO    BUTON           ;Hayır, tekrar test et

; Değişkeni artır ve porta gönder……..
        MOVLW   H'00'           ;W ← H'00'
```

```
        MOVWF   ART             ;ART =W
DON
        MOVF    ART,W           ;W← ART
        ANDLW   B'00001111'     ;Portun düşük olan bitlerini sıfırla
        MOVWF   PORTB           ;PORTB' ye bilgiyi gönder
        INCF    ART,F           ;ART değişkeninin içeriğini bir artır
        CALL    GECIKME         ;GECIKME alt programını çağır
        GOTO    BUTON           ;Butonu kontrol et

; gecikme alt programı……
GECIKME
        MOVLW    H'FF'          ;W<--H'FF'
        MOVWF   SAYAC1          ;SAYAC1 <--W
D1
        MOVLW   H'FF'           ;W<--H'FF'
        MOVWF   SAYAC2          ;SAYAC2 <--W
D2
        DECFSZ  SAYAC2,F        ;Sayac2 bir azalt ve sıfır mı? kontrol et
        GOTO    D2              ;Hayır D3 e git
        DECFSZ  SAYAC1,F        ;Sayac1 bir azalt ve sıfır mı?
        GOTO D1                 ;Hayır D1 e git
        RETURN
        END
```

Devrede butona her basılışta PORTB'nin yüksek bitlerindeki dijital bilgi artırılır. Portun çıkışına bağlanan DAC devresi ile PIC' in çıkışındaki dijital bilgi analog sinyale dönüştürülür.

1.4.2. PWM Metodu ile DAC Uygulama Devresi

PWM (Pulse Width Modulation) darbe genişlik modülasyonudur. PWM sinyali kare dalga bir sinyaldir. Bu kare dalga sinyalin darbe genişliği artırılıp azaltılarak PWM sinyal elde edilir. PWM yöntemi motor hız kontrolü, lambanın parlaklık ayarı gibi uygulamalarda kullanılmaktadır. Darbelerin genişliği arttıkça motorun hızı veya lambanın parlaklığı artar, darbelerin genişliği azaldıkça orantılı olarak azalır.

PWM sinyalin darbe genişliği çıkış geriliminin ortalama değeriyle doğru orantılıdır.

Darbe genişliği %50 ise çıkış gerilimi $\frac{5V}{2} \rightarrow 2{,}5V$

Darbe genişliği %25 ise çıkış gerilimi $\frac{5V}{4} \rightarrow 1{,}25V$

Darbe genişliği %75 ise çıkış gerilimi $5V\frac{3}{2} \rightarrow 3{,}75\text{V}$ olur.

PWM sinyal kare dalga olduğundan çıkış sürekli "1" ve "0" olarak değiştirilir. Burada önemli olan gecikme alt programının süresidir. Darbe genişliği %50 olursa tek gecikme alt programı kullanılır. Diğer durumlarda ise iki gecikme alt programı kullanılır.

Gecikme döngüsünde sayaç **h'FF'** kullanıldığında sayacın sıfırlanması için programın 256 kez çalıştırılması gerekir. Döngüde kullanılan sayacın alacağı değerler aşağıda gösterilmiştir.

- 2.5V'luk çıkış için gerekli süre

256 X (%50) = 128 → h'80' (%50 darbe genişliği için gerekli süre)

- 1.25V'luk çıkış için gerekli süre

256 X (%25) = 64 → h'40' (%25 darbe genişliği için gerekli süre)

- 1 V'luk çıkış için gerekli süre

256 X (%20) = 51.2= 51 → h'33' (%20 darbe genişliği için gerekli süre)

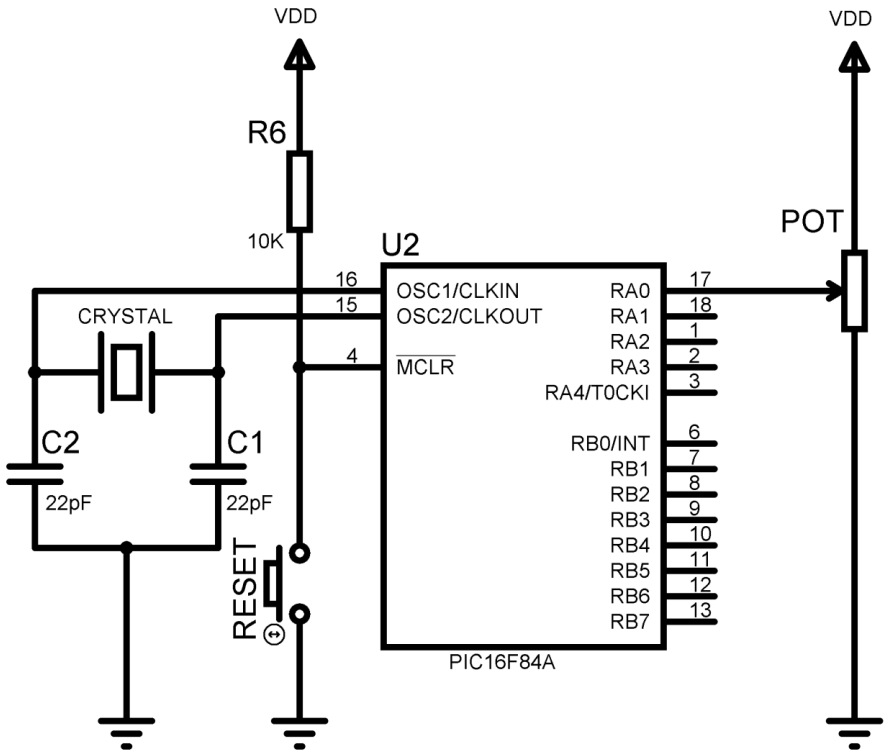
Aşağıda asm programı verilen devrede PortB bağlı olan 1. bit çıkışında 2V'luk bir gerilim üretilmektedir. Gerilim 2V olduğundan darbe genişliği %40 olmalıdır. Yani kullanılan alt programların sayaçları h'66' ve h'9A' olmalıdır.

```
;========PWM UYGULAMA PROGRAMI=============

        LIST        P=16F84
        INCLUDE     "P16F84.INC"

SAYAC1 EQU H'0C'            ;Gecikme alt programlarında kullanılan değişken
SAYAC2 EQU H'0D'            ;Gecikme alt programlarında kullanılan değişken

;Portları Ayarla……
        CLRF        PORTB           ;PORTB'yi temizle
        BSF         STATUS,5        ;BANK1'e geç
        CLRF        TRISB           ;PORTB çıkış
        BCF         STATUS,5        ;BANK0'a geç

; Programın başlangıcı……
BASLA
        BSF         PORTB, 1        ;Çıkışı "1" yani 5V yap
        CALL        GECIKME1        ;Darbe genişliği %40 süresini bekle
        BCF         PORTB,1         ;Çıkışı "0" yani 0V yap
        CALL        GECIKME2        ;%60 süresince 0V olarak bekle
```

```
        GOTO        BASLA           ;BASLA etiketine git
GECIKME1
        MOVLW       H'66'           ;W<--H'66"
        MOVWF       SAYAC1          ;SAYAC1 <--W
D1
        DECFSZ      SAYAC1,F        ;Sayac1 bir azalt ve sıfır mı? Kontrol et
        GOTO        D1              ;Hayır D1'e git
        RETURN
GECIKME2
        MOVLW       H'9A'           ;W<--H'9A'
        MOVWF       SAYAC2          ;SAYAC2 <--W
D2
        DECFSZ      SAYAC2,F        ;Sayac2 bir azalt ve sıfır mı? Kontrol et
        GOTO        D2              ;Hayır D2'e git
        RETURN
        END
```

1.5. PIC ile ADC Uygulama Devresi

Analog bir sinyal genellikle bir sensör aracılığıyla sağlanır. Uygulamalarda analog sinyal, gerilim bölücü direnç (potansiyometre) ile elde edildiğinde Şekil 1.8'deki bağlantı yapılır. Bu bağlantı ile PIC'e bağlı olan bir direncin değeri ölçülür. Ölçülen değer PORT'un çıkışlarında binary olarak görülür.



Şekil 1.8: PIC'e analog giriş bağlantısı

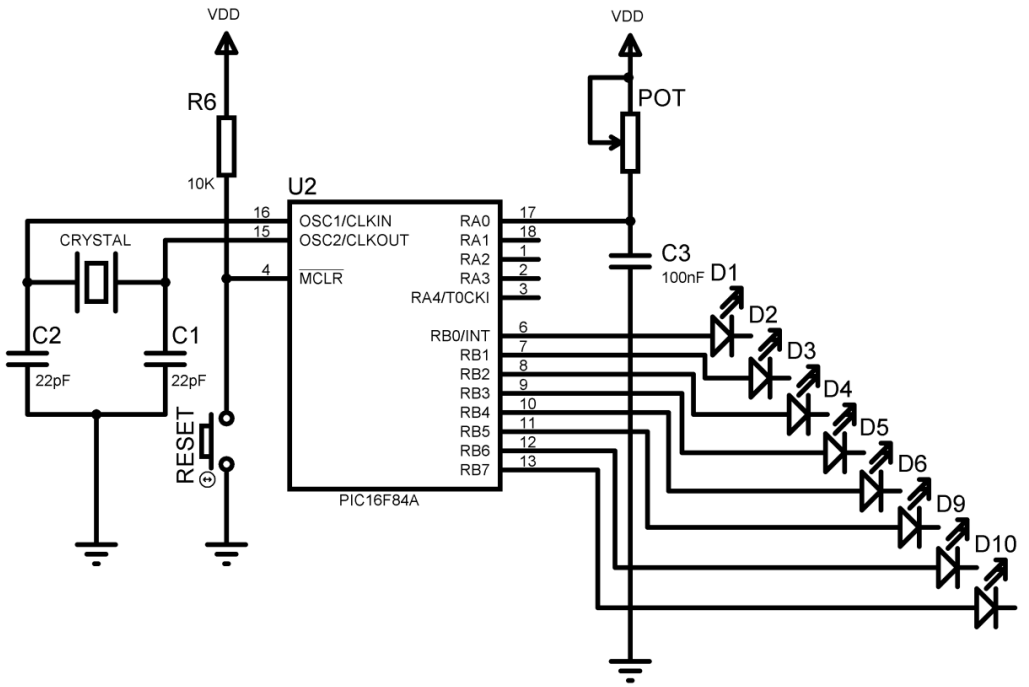
A/D çevrim metodu ile yapılan ölçümde değeri bilinmeyen bir direncin, bir kondansatörü V değerine ne kadar sürede şarj ettiği bulunur. Bunun için TMR0 sayıcısı şarj süresini ölçmek için kullanılır. A/D çevrim metodunun bağlantısı Şekil 1.9'da gösterilmiştir. Ölçme işlemi aşağıdaki gibi yapılır.

- PortA çıkış olarak yönlendirildikten sonra kondansatöre "0" bilgisi gönderilerek deşarj edilir.

- Kondansatörün deşarj süresinden sonra PortA giriş olarak yönlendirilir ve TMR0 sayıcısı çalıştırılır.

- PortA'daki gerilim V değerine ulaştığında TMR0 sayıcısı okunur.

- Şarj süresi direncin değeriyle doğru orantılıdır. Direnç büyüdükçe şarj süresi de büyüyeceğinden TMR0 registeri içerisinde daha büyük bir sayı okunur.



Şekil 1.9: PIC'e A/D çevirim metodu bağlantısı

```
;========ADC UYGULAMA PROGRAMI=============
        LIST        P=16F84
        INCLUDE     "P16F84.INC"
        ORG         H'00'
        GOTO        BASLA
        ORG         H'04'
        GOTO        KESME

;Portları Ayarla……
        CLRF        PORTB               ;PORTB'yi temizle
        BSF         STATUS,5            ;BANK1'e geç
        CLRF        TRISB               ;PORTB çıkış
        CLRF        TRISA               ;PORTA çıkış
        BCF         STATUS,5            ;BANK0'a geç
        MOVLW       B'10100000'         ;TMR0 registerini kur
        MOVWF       INTCON
        BSF         STATUS,5            ;BANK1'e geç
        MOVLW       B'11010001'         ;TMR0 aktif, dâhili komut saykılı
        MOVLW       OPTION_REG
        BCF         STATUS,5            ;BANK0
```

```
        BCF     PORTA,0         ;Kondansatörü deşarj et
        CLRF    TMR0            ;TMR0 zamanlayıcı başlat
DON
        BTFSS   TMR0,7          ;deşarj bitti mi?
        GOTO    DON             ;Hayır bekle
        BSF     STATUS,5        ;BANK1
        BSF     TRISA,0         ;PORTA 0 giriş
        BCF     STATUS,5        ;BANK0
        CLRF    TMR0            ;TMR0 yeniden başlat
DON2
        BTFSS   PORTA,0         ;0.bit "1" mi?
        GOTO    DON2            ;Hayır geri dön
        MOVF    TMR0,W          ;Evet, TMR0'ı oku
        MOVWF   PORTB           ;Kondansatör dolma süresi göster
        BCF     INTCON,5
DONGU
        GOTO DONGU
KESME
        BCF     INTCON,5        ;TMR0 kesmesini iptal et
        MOVLW   H'AA'
        MOVWF   PORTB           ;TMR0 dolma süresini göster
BEKLE
        GOTO BEKLE
        END

; Programın başlangıcı……
BASLA
        BSF     PORTB, 1        ;Çıkışı "1" yani 5V yap
        CALL    GECIKME1        ;Darbe genişliği %40 süresini bekle
        BCF     PORTB,1         ;Çıkışı "0" yani 0V yap
        CALL    GECIKME2        ;%60 süresince 0V olarak bekle
        GOTO    BASLA           ;BASLA etiketine git
GECIKME1
        MOVLW   H'66'           ;W<--H'66'
        MOVWF   SAYAC1          ;SAYAC1 <--W
D1
        DECFSZ  SAYAC1,F        ;Sayac1 bir azalt ve sıfır mı? kontrol et
        GOTO    D1              ;Hayır D1'e git
        RETURN
GECIKME2
        MOVLW   H'9A'           ;W<--H'9A'
        MOVWF   SAYAC2          ;SAYAC2 <--W
D2
```

```
DECFSZ      SAYAC2,F        ;Sayac2 bir azalt ve sıfır mı? kontrol et
GOTO        D2              ;Hayır D2'e git
RETURN
END
```

UYGULAMA FAALİYETİ

Şekil 1.9 daki uygulama devresini kurunuz.

İşlem Basamakları	Öneriler
➢ Kurulacak sistem için ihtiyaçları tespit ediniz.	➢ Kullandığınız devre elemanlarının özelliklerini internetten araştırınız.
➢ İhtiyacı karşılayacak mikrodenetleyiciyi seçiniz.	➢ Mikrodenetleyici olarak PIC 16F84A kullanınız.
➢ Analog veri için gerekli hesaplamaları yapınız.	➢ Matemetiksel işlemleri dikkatli şekilde yapınız.
➢ Sistemin mikrodenetleyici programını yazınız.	➢ Programı yazdıktan sonra MPLAB ile PROTEUS programlarında deneyiniz.
➢ Programı mikrodenetleyiciye yükleyiniz.	➢ Programı mikrodenetleyiciye yüklerken kullanılan pic programlayıcıya uygun yazılım kullanınız.
➢ Çevre elemanları ve analog veri sağlayan elemanlar ile devreyi kurunuz.	➢ Devrenin montajını yapmadan önce breadboard üzerinde kurarak deneyiniz.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet,** kazanamadıklarınız için **Hayır** kutucuklarına (X) işareti koyarak öğrendiklerinizi kontrol ediniz

Değerlendirme Ölçütleri		Evet	Hayır
1.	Kurulacak sistem için ihtiyaçları tespit edebildiniz mi?		
2.	İhtiyacı karşılayacak mikrodenetleyiciyi seçebildiniz mi?		
3.	Analog veri için gerekli hesaplamaları yapabildiniz mi?		
4.	Sistemin mikrodenetleyici programını yazabildiniz mi?		
5.	Programı mikrodenetleyiciye yükleyebildiniz mi?		
6.	Çevre elemanları ve analog veri sağlayan elemanlar ile devreyi kurabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda "Hayır" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "Evet" ise "Ölçme ve Değerlendirme"ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1.( ) Dijital bir bilginin analog işaretlere çevirmesi için kullanılan elemanlara DAC adı verilir.

2.( ) DAC devreleri dirençlerin farklı bağlanması ve opamplar yardımıyla yapılır.

3.( ) Dalga genişlik modülasyonuna PCM denir.

4.( ) PIC ile gerçekleştirilen ADC devresinde ölçüm için RC elemanları kullanılır.

5.( ) PIC DAC devresinde gerilim değerini arttırmak için potansiyometre kullanılır.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ–2

AMAÇ

Mikrodenetleyici ile analog uygulama devreleri yapabileceksiniz.

ARAŞTIRMA

- 16F877 entegresinin analog dijital dönüştürme özelliğini araştırmalısınız.
- LM 35 entegresinin çalışmasını öğrenmelisiniz.
- DC motorun yapısını ve çalışmasını öğrenmelisiniz.
- Araştırma işlemleri için Mikrodenetleyici ile Dijital İşlemler modülünü gözden geçirebilir ve internet ortamından yararlanabilirsiniz.

2. UYGULAMA DEVRELERİ

2.1. Pic 16f877 Entegresinin Özellikleri

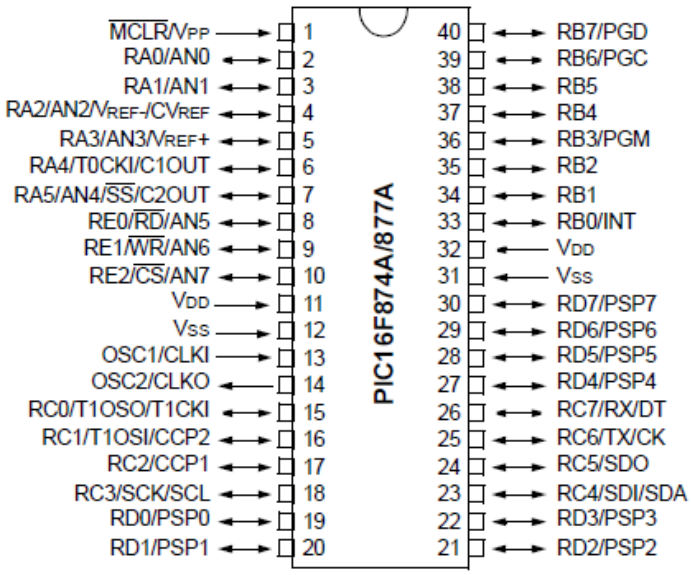
PIC16F877, en popüler PIC işlemcisi olan 16F84'ten sonra kullanıcılara yeni ve gelişmiş olanaklar sunmasıyla hemen göze çarpmaktadır. Program belleği FLASH ROM olan F877'de, F84'te olduğu gibi elektriksel olarak silinip yazılmaktadır. PIC 16F877 ile 16F84 arasındaki farklar tablo 2.1'de verilmiştir.

16F877 mikrodenetleyicisi 4 veya 20 MHz osilatör ile çalışmaktadır. Bu sayede daha hızlı bir sistemler kurulabilir. 5 adet giriş/çıkış portu (A, B, C, D, E) bulunmaktadır. A portu 6 I/O, B portu 8 I/O, C portu 8 I/O, D portu 8 I/O ve E portu 3 I/O sahiptir. Port sayısı fazla olduğundan aynı anda LCD, klavye, motor gibi elemanlar çalıştırılabilir. Üç tane timer'ı bulunur. Şekil 2.1' de PIC 16F877'nin bacak bağlantısı gösterilmiştir.

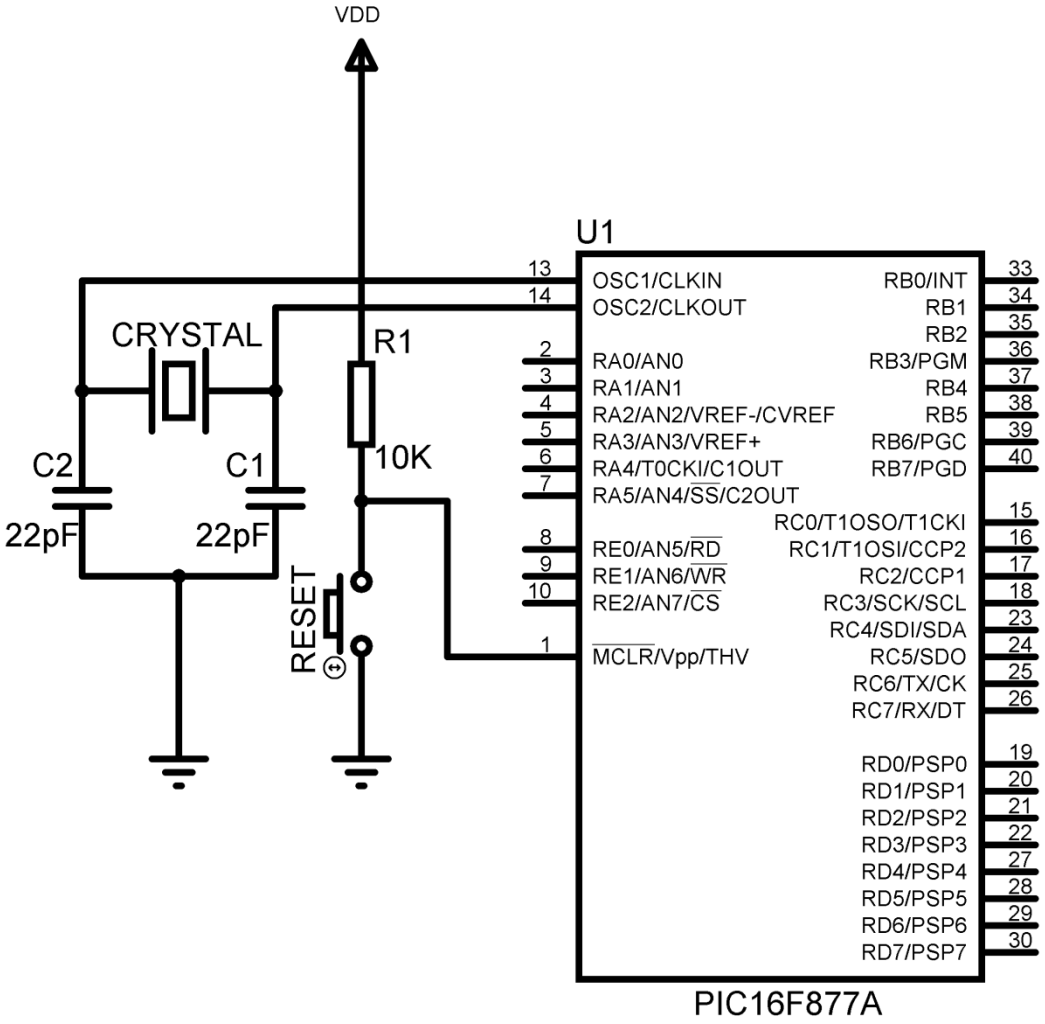
Özellikler	**PIC16F84**	**PIC16F877**
Çalışma hızı	DC-10 Mhz	DC-20Mhz
Program belleği	1Kx14 word Flash ROM	8Kx14 word Flash ROM
EEPROM belleği	64 byte	256 byte
Kullanıcı RAM	68 x 8 byte	368 x 8 byte
Giriş / Çıkış port sayısı	13	33
Timer	Timer0	Timer0, Timer1, Timer2
A / D çevirici	YOK	8 kanal 10 bit

Capture / Comp / PWM	YOK	16 bit Capture 16 bit Compare 10 bit PWM çözünürlük
Seri çevresel arayüz	YOK	SPI (Master) ve I2C (Master / Slave) modunda SPI portu (senkron seri port)
Paralel slave port	YOK	8 bit, harici RD,WR ve CS
USART / SCI	YOK	9 bit adresli

Tablo 2.1: 16F84 ile 16F877 mikrodenetleyicilerinin karşılaştırılması



Şekil 2.1: PIC 16F877 mikrodenetleyicisinin bacak bağlantısı



Şekil 2.2: PIC 16F877 mikrodenetleyicisinin minimum bağlantıları

Pin Adı	Görevi
OSC1/CLKIN	Osilatör clock girişi (kristal)
OSC2/CLKOUT	Osilatör kristal çıkış ucu
MCLR/Vpp	Resetleme girişi
RA0/AN0 RA1/AN1 RA2/AN2/V_{REF} RA3/ AN3 RA4/TOCK1 RA5/SS/AN4	Port A iki yönlü giriş/çıkış portudur. Analog giriş olarak kullanılabilir.
	Bu pin TMR0 için clock girişi olarak da kullanılabilir.
	SSP Slave seçme pini veya analog giriş/çıkış olabilir.
RB0/INT RB1 RB2 RB3/PGM RB4 RB5 RB6/PGC RB7/PGD	Dış kesme girişi olarak seçilir. Port B iki yönlü giriş/çıkış portudur.
	Düşük akımla programlamada da kullanılabilir.
	Kesme girişi olarak seçilebilir.
	Kesme girişi olarak seçilebilir.
	Kesme girişi olarak seçilebilir. Seri programlamada clock girişidir.
	Kesme girişi olarak seçilebilir. Seri programlamada data pinidir.
RC0/T1OSO/T1CK1	Timer1 osc. girişi veya saat girişi olarak kullanılabilir.
RC1/T1OSI/CCP2	Timer1 osc. girişi / Capture2 girişi/Compare2 çıkışı /PWM2 çıkışı
RC2/CCP1 RC3/SCK/SCL	Timer1 osc girişi/ Capture 1 girişi/ Compare 1 çıkışı / PWM1 çıkışı
RC4/SDI/SDA	SPI ve I_2 modunda, seri saat giriş/ çıkışı SPA moda SPI giriş verisi veya I_2 C moda I/O için kullanılır.
RC5/SDO	SPA moda SPI çıkış verisi için seçilebilir.
RC6/TX/CX	USART asenkron gönderme ya da senkron saat için kullanılır.
RC7/RX/DT	USART asenkron alma veya senkron veri için kullanılır.
	Port C iki yönlü giriş çıkış portudur.
RD0/ PSP0 RD1/PSP1 RD2/PSP2 RD3/PSP3 RD4/PSP4 RD5/PSP5 RD6/PSP6 RD7/PSP7	Port D iki yönlü giriş çıkış portudur. PSP bitleridir.
RE0/RD/AN5	Analog giriş ya da PSP okuma kontrolü olarak kullanılabilir.
RE1/WR/AN6	Analog giriş ya da PSP yazma kontrolü olarak da kullanılabilir.
RE2/CS/AN7	Analog giriş ya da PSP seçim kontrolü için kullanılabilir.
V_{ss}	Toprak
V_{DD}	Pozitif kaynak

Tablo 2.2: PIC16F877 mikrodenetleyicisinin pinlerinin görevleri

2.2. A/D Çevirici Uygulama Devresi

Daha önceden de bahsettiğimiz gibi analog bilgiyi dijital bilgiye dönüştürmek için PIC16F84 kullanıldığında ek devrelere ihtiyaç duyulmaktadır. Bundan dolayı bu uygulamada PIC 16F877 entegresi kullanılmaktadır.

PIC16F877'de 8 tane 10 bitlik A/D çevirme kanalı bulunur. A / D kanalları için RA4 hariç diğer A ve E portları kullanılır. A/D çevirme işlemi dört adet kaydediciyle yapılmaktadır.

Aşağıda ilgili registerlar ve adresleri gösterilmiştir.

ADRESH	*0x1E* ;	A / D sonuç kaydedicisi (high register)
ADRESL	*0x9E* ;	A / D sonuç kaydedicisi (low register)
ADCON0	*0x1F* ;	A / D kontrol kaydedicisi 0
ADON1	*0x9F* ;	A / D kontrol kaydedicisi 1

ADCON0: 8 bitlik bir A/D kaydedicisidir ve tablo 2.3'te iç yapısı gösterilmiştir.

ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

Tablo 2.3: ADCON0 kaydedicisinin iç yapısı

- **ADCS1-ADCS0:** A/D dönüştürme clock sinyali seçme bitidir. Sinyal seçme işlemi ADCON1 registeri ile birlikte kullanılır. ADCS2 (ADCON2 kaydedicisinin 6.biti), ADCS1 ADCS0 bit üçlüsü clock sinyali seçmek için kullanılan bitlerdir. Tablo 2.4'te bitlerin durumu ve clok sinyalleri gösterilmiştir.

ADCON1 (ADCS2)	**ADCON0** (ADCS1-ADCS0)	**Dönüştürme clock sinyali**
0	00	$F_{osc}/2$
0	01	$F_{osc}/8$
0	10	$F_{osc}/32$
0	11	F_{RC} (Sinyal iç RC devresi tarafından sağlanır.)
1	00	$F_{osc}/4$
1	01	$F_{osc}/16$
1	10	$F_{osc}/64$
1	11	F_{RC} (Sinyal iç RC devresi tarafından sağlanır.)

Tablo 2.4: ADCON0 kaydedicisinin bit durumları

- CHS2-CHS1-CHS0 bitleri analog giriş kanalı seçme bitleridir.

000 : Kanal 0 (AN0/RA0)
001 : Kanal 1 (AN1/RA1)
010 : Kanal 2 (AN2/RA2)
011 : Kanal 3 (AN3/RA3)
100 : Kanal 4 (AN4/RA5)
101 : Kanal 5 (AN5/RE0)
110 : Kanal 6 (AN6/RE1)
111 : Kanal 7 (AN7/RE2)

- GO/DONE biti dönüştürme işlemi durum bitidir.

1: A/D dönüştürme işlemi sırasında donanım tarafından aktive edilir. Sonra tekrar resetlenir.

0: A/D dönüştürme işlemi yapılmıyor.

- ADON: A/D dönüştürücü modülü açma biti

1: A/D modülü çalıştırıldı.
0: A/D modülü çalışmıyor.

ADCON1: 8 bitlik bir A/D kaydedicisidir ve tablo 2.5'te bitleri görülmektedir.

ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

Tablo 2.5: ADCON1 kaydedicisinin iç yapısı

- ADFM biti dönüştürme işlemi sonunda 10 bitlik sayının formatını belirler.

1: sayı sağa kaydırılır ve düşük 8 biti ADRESL'de, üst 2 biti de ADRESH registerinde tutulur. ADRESH registerinin üst 6 biti sıfırlanır. Normal A/D dönüşüm işlemlerinde ADFM=1 kullanılır.

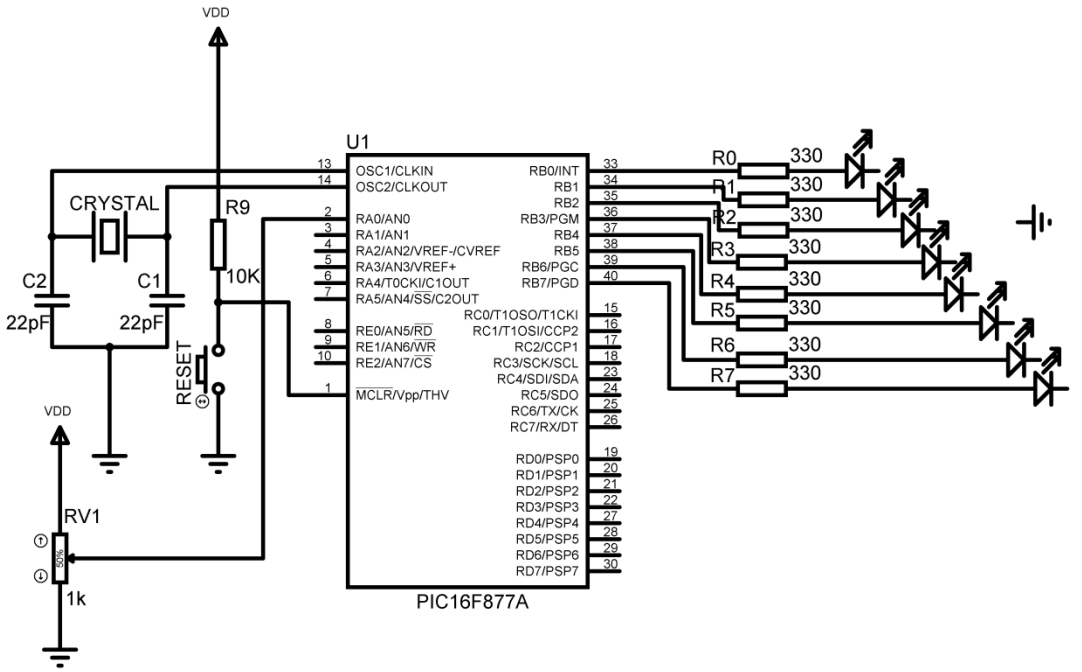
0: Sayı sola kaydırılır ve ADRESL registerinin düşük 6 biti sıfırlanır. Sayı 10 bitlik olduğundan 6 biti sıfırlanır.

- PCFG3 – PCFG0 bitleri A/D portlarının durumunu belirler. Bu bitler ile hangi portların analog ve hangi portların sayısal giriş-çıkış için kullanılacaklarını belirler ve analog portlarda kullanılacak referans gerilimleri seçeriz. En yaygın kullanılan tüm bitlerin 0 olmasıdır. Bu durumda 8 tane analog kanal seçilmiş olur ve her kanalın referans gerilimi VDD olarak ayarlanır.

2.2.1. Devrenin Malzemeleri

- PIC 16F877 4 Mhz mikrodenetleyici
- X1= 4Mhz kristal
- C1 = C2 = 22pf
- R1 = R2…R8 = 330Ω direnç
- RV1= 1K potansiyometre
- R9= 10K,
- Buton
- 8 Adet LED Diyot

2.2.2. Devrenin Şeması



Şekil 2.3: PIC 16F877 ile A/D uygulama devresi

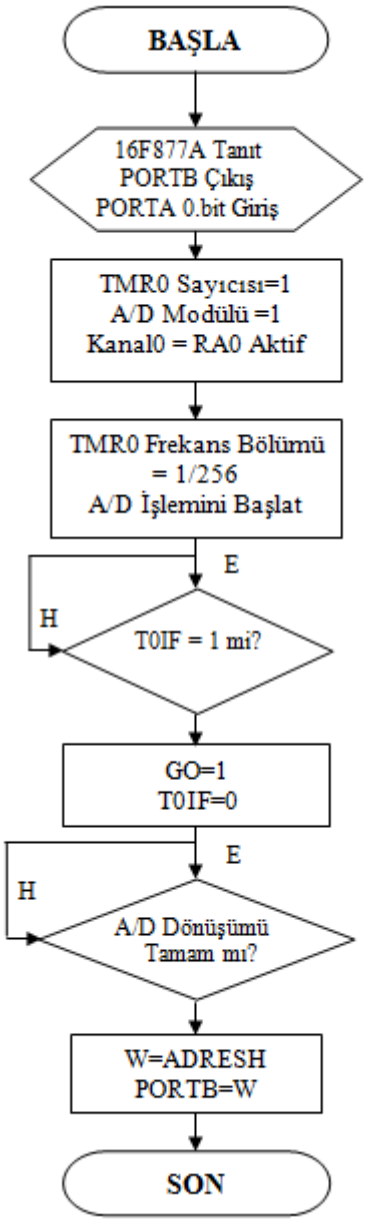
2.2.3. Devrenin Asm Programı

```
;=====ANALOG DİJİTAL ÇEVİRİCİ UYGULAMA PROGRAMI=========
        LIST        P=16F877
        INCLUDE     "P16F877.INC"
        BCF         STATUS, 5           ;STATUS 5. biti =0
        BCF         STATUS, 6           ;STATUS 6.biti=0 BANK 0 a geç
        MOVLW       B'00100000'         ;TMR0 sayıcısını aktif yap
        MOVWF       INTCON
```

```
        CLRF    PORTA           ;Port A sıfırla
        CLRF    PORTB           ;Port B sıfırla
        MOVLW   B'01000001'     ;A/D çevirici modülü aktif
        MOVWF   ADCON0          ;Kanal 0 (RA0) aktif
        BSF     STATUS, 5       ;STATUS 5. biti = 1
        BCF     STATUS, 6       ;STATUS 6. biti =0 BANK 1 e geç
        MOVLW   B'10000111'     ;Frekans bölme değeri TMR0
        MOVWF   OPTION_REG      ;Frekans bölme sayısı 1/256'dır.
        MOVLW   B'00000000'     ;Sayıyı sola kaydır.
        MOVWF   ADCON1          ;Sayıyı ADRESH a yükle.
        MOVLW   B'00000001'     ;Port A.0 = giriş
        MOVWF   TRISA
        CLRF    TRISB           ;Port B çıkış
        BCF     STATUS, 5
        BCF     STATUS, 6       ;BANK 0 a geç

;================PROGRAM BAŞLANGICI=====================
BASLA
        BTFSS   INTCON, T0IF    ;TMR0 sayıcının sayımı bitti mi?
        GOTO    BASLA           ;Hayır BASLA etiketine git
        BCF     INTCON, T0IF    ;Evet sayıcıyı yeniden kur
        BSF     ADCON0, GO      ;A/ D dönüşümünü başlat
DON
        BTFSS   PIR1, ADIF      ;A/D dönüşümü tamam mı?
        GOTO    DON             ;Hayır geri dön
        MOVF    ADRESH,W        ;Evet, ADRESH →W aktar
        MOVWF   PORTB           ;Aktarılan sayıyı çıkışa gönder
        GOTO    BASLA           ;BASLA etiketine geri dön
        END
```

2.2.4. Akış Diyagramı

Programda ilk olarak PIC tanıtılır. TMR0 sayıcısı kurulur ve A/D dönüşüm için ADCON0, ADCON1 registerleri ayarlanır. Devrede analog bilgi olarak 5V bir kaynak ile potansiyometre kullanılmıştır. Potansiyometrenin değişimiyle kanal 0'ın (Port A 0.bit) giriş gerilimi değişmektedir. Giriş gerilimi 10 bitlik dijital veriye dönüştürülür. Bu verinin en yüksek bitleri ADRESH registerinde, kalan 2 biti ise ADRESL registerinde saklanır. ADRESH registerindeki bilgi çıkışa aktarılarak girişteki analog bilginin çıkıştaki dijital dizilimi görülür. Giriş gerilimi değiştirildikçe çıkıştaki dijital veri de değişmektedir.



Tablo 2.6: A/D çevirici devresinin akış diyagramı

Programın başlangıcında TMR0 sayıcısı kontrol edilir. Bu sayıcı 1/256 frekans bölme sayısı ile 00'dan FF'ye kadar sayar, FF sayısına ulaşıldığında INTCON registerinde T0IF bayrağı "1" olur. T0IF 1 olduğunda A/D çevrime başlanır ve T0IF sıfırlanarak sayıcı yeniden kurulur. A/D çevirimin bitip bitmediğini anlamak için de PIR1 registerinin ADIF bayrağı kontrol edilir. ADIF "1" ise A/D çevrim bitmiştir.

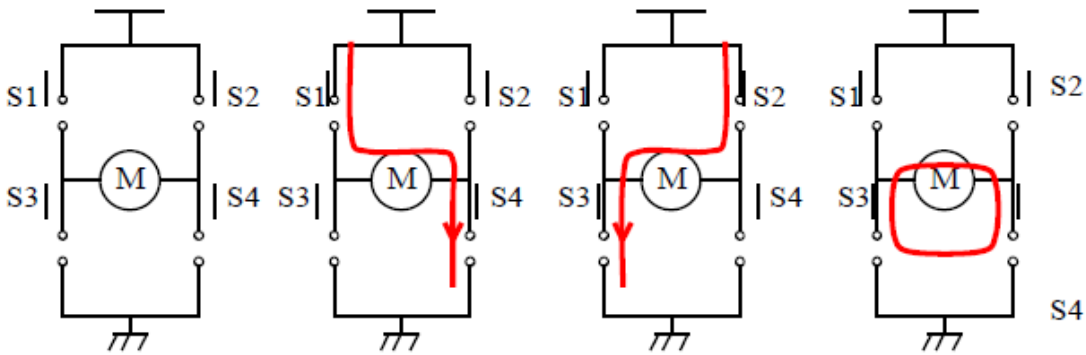
PIR1, Kesmeleri kontrol etmek için kullanılan özel bir registerdir. Bu registerin 6. biti ile A/D dönüşümünün bitip bitmediği kontrol edilir.

INTCON, registerinin 5. biti ile TMR0 registeri aktif hâle getirilir. 2. biti ile de sayıcıda taşma olup olmadığı kontrol edilir.

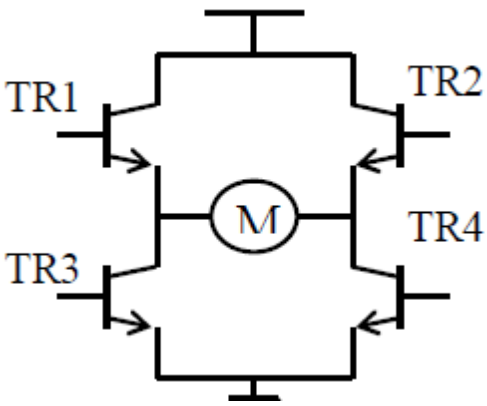
2.3. DC Motor Yön ve Hız Kontrol Devresi

DC motorlar uçlarına uygulanan DC gerilim ile çalışır. Genellikle besleme gerilimi motor gövdesinde yazar. Uygulanan gerilimin değeri değiştirilerek motorun hızı ve kutupları değiştirilerek motorun yönü ayarlanır.

Şekil 2.4'te motorun yön kontrolü gösterilmiştir. DC motorların yönü anahtarlama ile değiştirilir. Şekil 2.5'te motorun yön kontrolünün bağlantı şeması gösterilmiştir.



Şekil 2.4: Motor yön kontrol temel devreleri



Şekil 2.5: Motorun transistörlü yön kontrol temel devresi

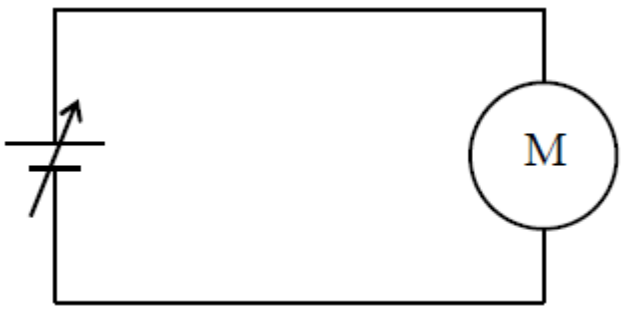
Şekil 2.4.a'da motor boştadır. Devrede tüm anahtarlar açık olduğundan motor besleme gerilimi almaz.

Şekil 2.4.b'de S1 ve S4 anahtarları kapalı konumdadır. Bu durumda motor saat yönünde döner.

Şekil 2.4.c'de S2 ve S3 anahtarları kapalı olduğundan motor saatin tersi yönünde döner.

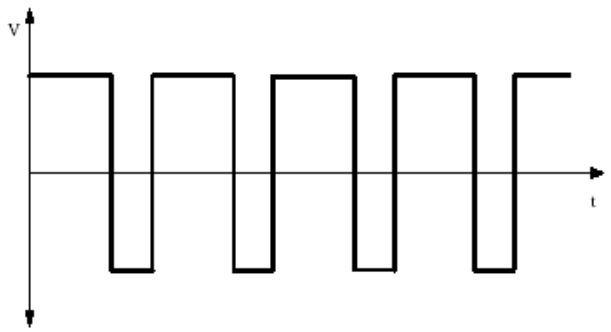
Şekil 2.4.d'de S2 ve S4 anahtarları kapalı konumdadır. Bu durumda motor her iki yönden şase potansiyeli aldığından motor durur. Bu durum çalışan motoru bulunduğu konumda tutmak için kullanılır.

- DC motorun hız kontrolünde DC motorun hız kontrolünde iki yöntem kullanılır. Birincisi gerilim kontrolü, ikincisi ise faz kontrolüdür. Gerilim kontrolü şekil 2.6'da gösterildiği gibi mikrodenetleyiciye ihtiyaç duyulmadan ayarlı bir güç kaynağı ile yapılmaktadır.



Şekil 2.6: DC motorun ayarlı güç kaynağı ile hız kontrol devresi

İkinci yöntemde PWM sinyal kullanılır. Öğrenme Faaliyeti-1'de PWM sinyalinin oluşturulmasını PIC 16F84 ile yapmıştık. Bu uygulama devresinde 16F877 entegresi kullanılacaktır. Çünkü bu mikrodenetleyicinin *CCP1 (RC1)* ve *CCP2 (RC2)* bacakları PWM çıkış üretir. Üretilen PWM sinyalin peryodu sabit fakat darbe genişliği değişebilen bir kare dalga sinyaldir. Kare dalga sinyalin darbe genişliği (duty saykılı) azalırsa ortalama gerilim azalır ve motorun hızı düşer. Sinyalin darbe genişliği artarsa motorun hızı artar. Şekil 2.7'de kare dalga sinyal gösterilmiştir.



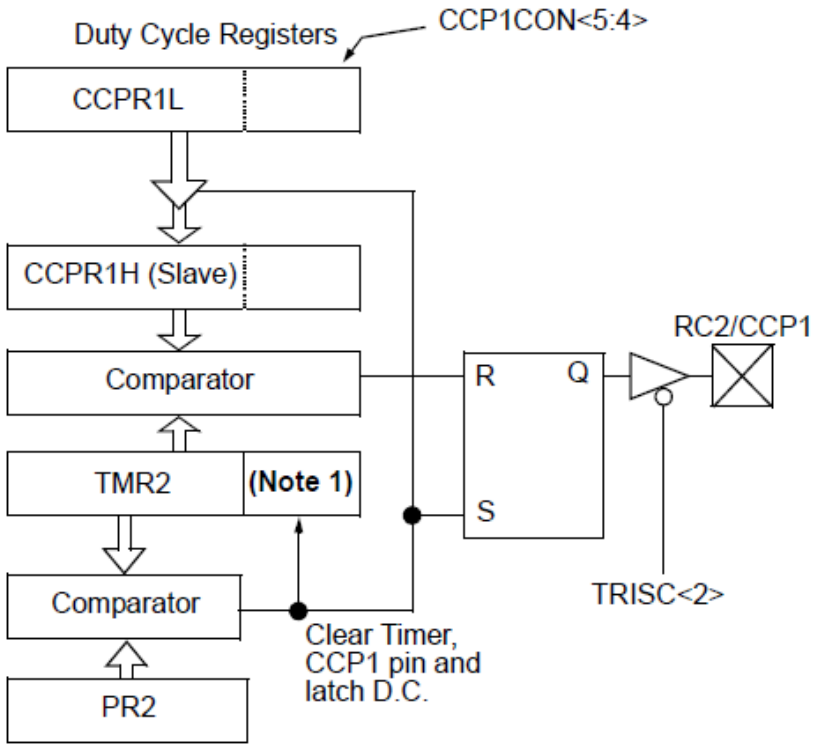
Şekil 2.7: PWM sinyali

PWM çalışma maduna geçebilmek için aşağıdaki aşamalar takip edilerek Capture Compare PWM(CCP) modülü düzenlenmelidir.

- PWM periyodu PR2 registerine yazarak başlanır.
- PWM görev saykılı CCPR1L register ve CCP1CON <5:4> bitlerine yazılır.
- CCP1 pini TRISC<2> biti temizlenerek çıkış yapılır.
- Timer2 (TMR2) prescale değeri girilir ve TMR2'nin T2CON biti set edilerek TMR2'nin çalışması sağlanır.
- PWM operasyonu için CCP1 modülü ayarlanır.

Darbe Genişlik Modülasyon (PWM) modunda, CCPx pini 10 bit kadar kararlı PWM çıkışı üretir. CCP1 pini PORTC data latch üzerinden çıkışa aktarılmıştır. CCP1 pinini çıkış yapmak için TRISC<2> biti mutlaka temizlenmelidir.

Bir PWM çıkışında, bir periyotluk süre dâhilinde çıkışın yüksek seviyede bulunduğu zamana duty cycle (görev saykılı) denir.



Şekil 2.8: Mikrodenetleyici PWM blok diyagramı

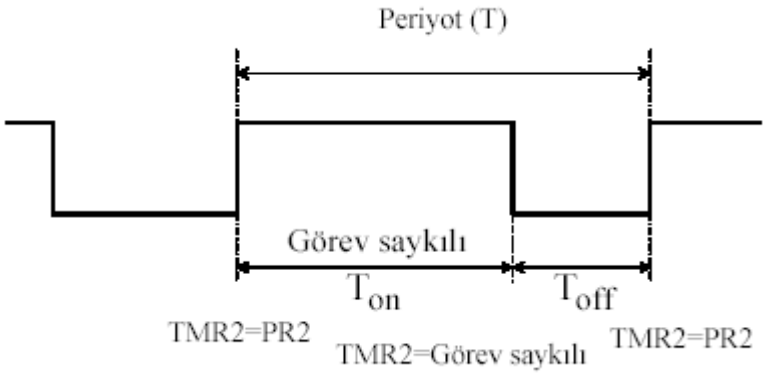
PWM periyodunun ve görev süresinin hesaplanması:

PWM periyodu,PR2 registerine yazılarak belirtilir ve eşitlik aşağıdaki formülle hesaplanabilir.

T=PWM period = [(PR2) + 1] . 4 . TOSC . (TMR2 prescale değeri)

PWM frekansı periyodun tersi olup eşitlik aşağıdaki formül ile hesaplanır.

f=1/T



Şekil 2.9: Mikrodenetleyici PWM çıkış sinyali

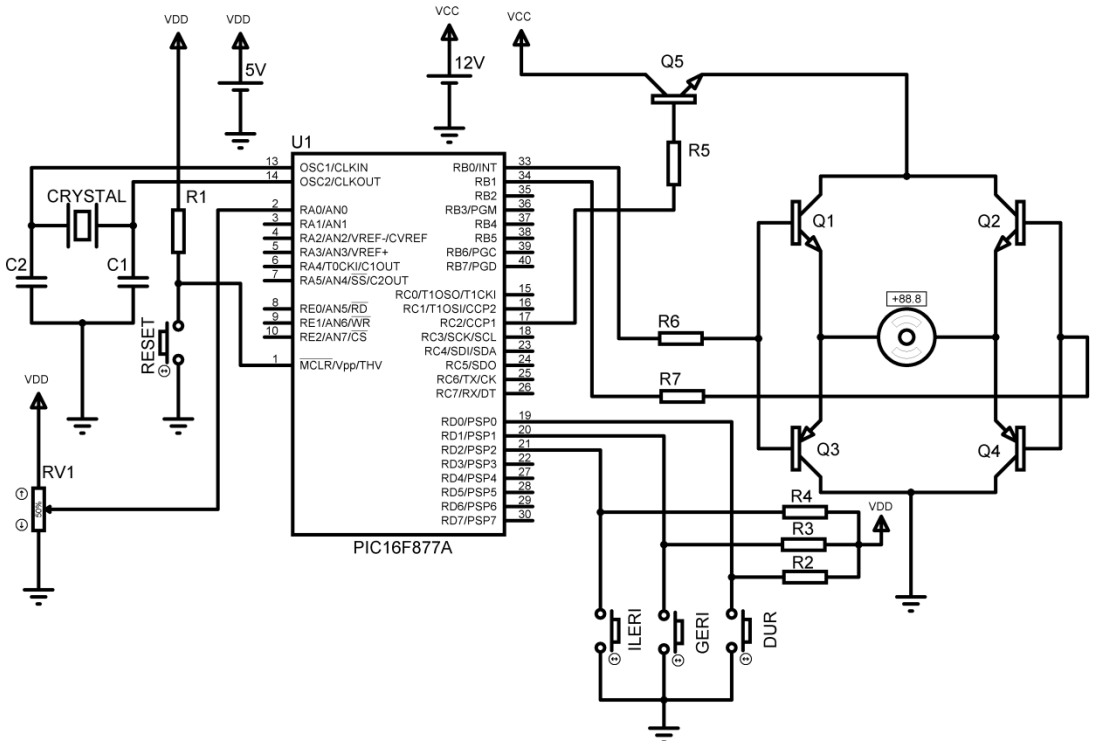
TMR2, görev saykılı'na (Ton) eşit oluncaya kadar CCPR1L'de lojik1 bilgisi görülür. TMR2 içeriği görev saykılına eşit olduğunda seviye lojik 0'a düşer. TMR2 içeriği sıfırlanmadan PR2'ye eşit oluncaya kadar çalışmaya devam eder. Böylece, Şekil 2.9'da gösterildiği gibi bir periyotluk süre tamamlanmış olur. Burada,

f = PWM sinyalinin frekansını (Hz)
T = PWM sinyalinin bir saykılını (s) gösterir.
Bir saykıllık PWM süresi ise; **T**=Ton+Toff
Ton = Pozitif PWM sinyal süresini veya PWM görev saykılını (s)
Toff = Sıfır veya negatif PWM sinyal süresini (s) göstermektedir.

2.3.1. Devrenin Malzemeleri

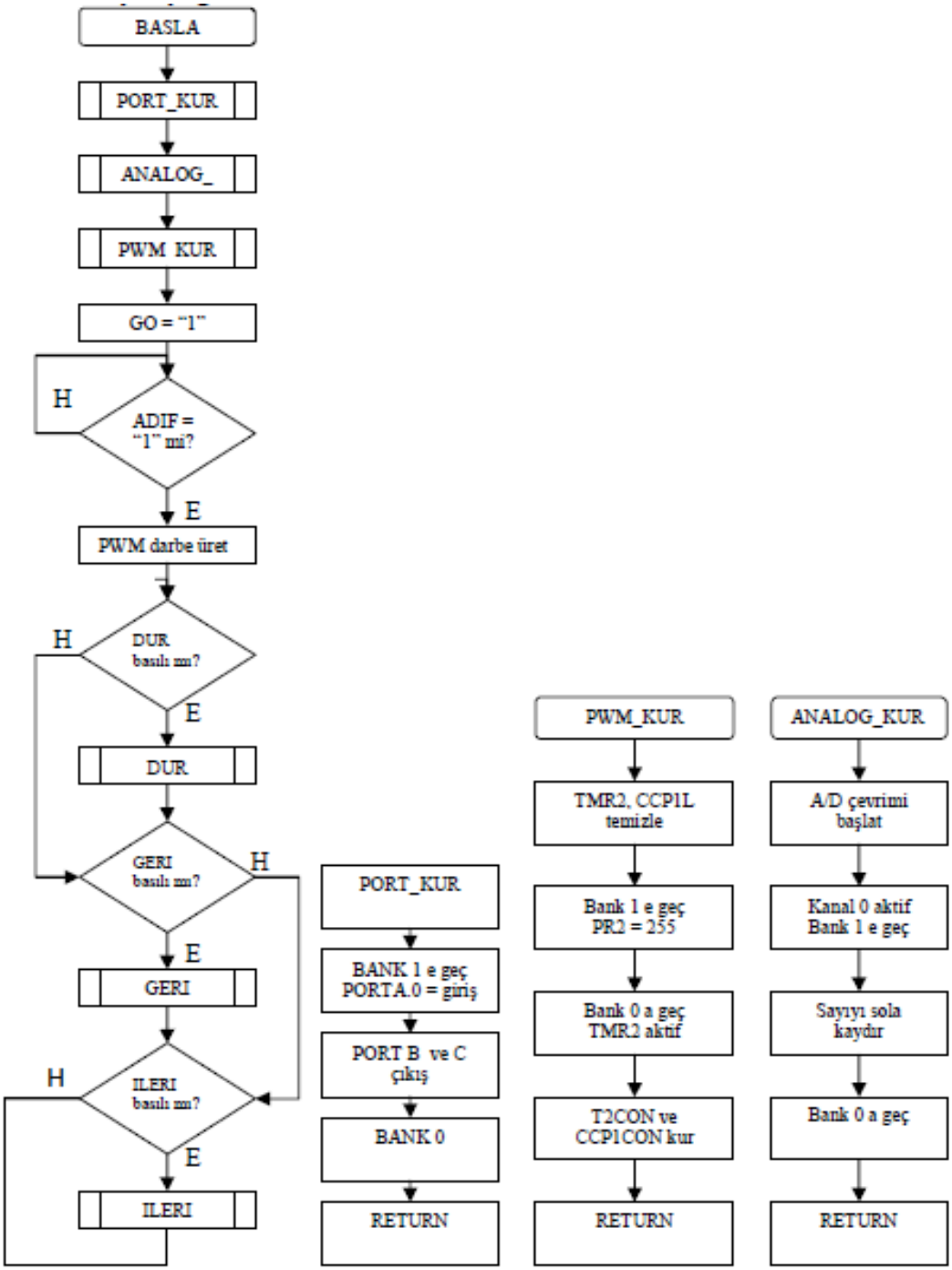
- PİC 16F877 4 Mhz mikrodenetleyici
- X1= 4Mhz kristal
- C1 = C2 = 22pf
- R1=R2=R3=R4= 10K, R5=R6=R7=1K
- DC motor, buton
- RV1= 1K potansiyometre
- Q1=Q2= BD 135
- Q3=Q4= BD140, Q5= BC237

2.3.2. Devrenin Şeması



Şekil 2.10: Mikrodenetleyici ile DC motor hız ve yön kontrol devresi

2.3.3. Akış Diyagramı



Tablo 2.7: DC motor hız ve yön kontrol devresinin program akış diyagramı

2.3.4. Devrenin Asm Programı

```
;====MOTOR HIZ VE YÖN KONTROL UYGULAMA PROGRAMI=========
        LIST        P=16F877
        INCLUDE     "P16F877.INC"
        CALL        PORT_KUR        ;PORT_KUR alt programına git
        CALL        ANALOG_KUR      ;ANALOG_KUR alt programına git
        CALL        PWM_KUR         ;PWM_KUR alt programına git
        BSF         ADCON0,GO       ;A/D çevrimi başlat
DON
        BTFSS       PIR1,ADIF       ;A/D çevrim bitti mi?
        GOTO        DON             ;Hayır geri dön
        MOVF        ADRESH,W        ;Evet sayıyı W'ye aktar
        MOVWF       CCPR1L          ;PWM darbeyi üret
TUS_ARA
        BTFSS       PORTD,0         ;DUR butonuna basılı mı?
        CALL        DUR             ;evet DUR alt programına git
        BTFSS       PORTD,1         ;GERI butonuna basılı mı?
        CALL        GERI            ;evet GERI alt programına git
        BTFSS       PORTD,2         ;ILERI butonuna basılı mı?
        CALL        ILERI           ;evet ILERI alt programına git
        GOTO        TUS_ARA         ;Hayır tuş ara

PORT_KUR;=============================================
        BSF         STATUS,5        ;Bank 1'e geç
        MOVLW       H'01'
        MOVWF       TRISA           ;PortA 0.bit giriş
        CLRF        TRISB           ;PortB çıkış
        CLRF        TRISC           ;PortC çıkış
        MOVLW       H'FF'
        MOVWF       TRISD           ;PortD giriş
        BCF         STATUS,5        ;Bank 0'a geç
        RETURN

ANALOG_KUR;==========================================
        MOVLW       B'10000001'     ;A/D çevrimi aktif
        MOVWF       ADCON0          ;Kanal 0 aktif
        BSF         STATUS,5        ;Bank 1'e geç
        MOVLW       B'00001110'     ;Sayıyı sola kaydır
        MOVWF       ADCON1          ;ADRESH'a yükle
        BCF         STATUS,5        ;Bank 0'a geç
        RETURN
```

```
PWM_KUR;=================================================
        CLRF    TMR2            ;TMR2 temizle
        CLRF    CCPR1L          ;CCPR1L temizle
        BSF     STATUS,5        ;Bank 1 e geç
        MOVLW   D'255'          ;PR2 registerine 255 sayısını yükle
        MOVWF   PR2             ;Peryod=1638.4µs
        BCF     STATUS,5        ;Bank 0 a geç
        MOVLW   B'00001100'     ;Pst=1:1 TMR2=ON Pre=1:16
        MOVWF   T2CON           ;T2CON registerini kur
        MOVLW   B'00001100'     ;CCP1XY=0 CCP1M=1100(PWM)
        MOVWF   CCP1CON         ;CCP1CON registerini kur
        RETURN

DUR;=====================================================
        BSF     PORTB,0         ;PortB.0 = "1"
        BSF     PORTB,1         ;PortB.1= "1"
        RETURN

GERI;====================================================
        BCF     PORTB,0         ;PortB.0= "0"
        BSF     PORTB,1         ;PortB.1= "1"
        RETURN

ILERI;===================================================
        BSF     PORTB,0         ;PortB.0="1"
        BCF     PORTB,1         ;PortB.1="0"
        RETURN
        END
```

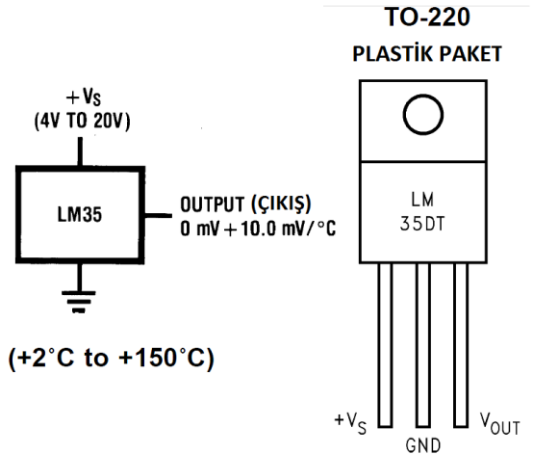
Devrede RV1 potansiyometresi motorun hızını ayarlamaktadır. RV1 analog girişe bağlandığından, potansiyometredeki değişim dijital bilgiye dönüştürülür ve bu değer PWM darbe için CCP1L registerine yüklenir. Bu değer arttıkça PWM darbenin genişliği artar, azaldıkça da azalır. Böylece motorun hızı ayarlanır. Bu sinyal üretildikten sonra motor çalışmaya başlamaz.

Motorun yönü "ILERI", "GERI" ve "DUR" butonları ile kontrol edilir. ILERI butonuna basıldığında PortB'nin 0. biti "lojik 1", 1. biti ise "lojik 0" olur. Bu durumda Q1 ve Q4 transistörleri iletime geçer, motor saat yönünde döner. GERI butonuna basıldığında PortB'nin 0. biti "lojik 0", 1. biti ise "lojik1" olur. Bu durumda Q2 ve Q3 transistörleri iletime geçer, motor saatin tersi yönünde döner. DUR butonuna basıldığında PortB'nin 0. ve 1. biti "lojik 1" olur. Bu durumda motor boşta kalır ve durur.

2.4. Isıtıcı ve Fan Kontrollü Uygulama Devresi

A/D dönüştürücü için gerekli analog giriş için LM 35 ısı sensörü kullanılmıştır. LM 35 çıkışında sıcaklığa bağlı olarak değişen doğrusal bir gerilim üreten sensördür. Bu gerilim 10mV/C olarak değişir. Yani sıcaklığın arttığı her derece için gerilim 10mV artar. LM 35 entegresinin özellikleri aşağıda sıralanmıştır.

- Doğrudan ayarlı santigrat derece
- Her 1 ◦C değişimde 10mV değişim faktörü
- -55 +150 ◦C çalışma aralığı
- Uzaktan kontrol uygulamalar için uygunluk
- 4-30 volt çalışma gerilimi
- 60 mikroamperden az kaynak akımı harcanımı



Şekil 2.11: LM35 entegresinin katolog bilgileri

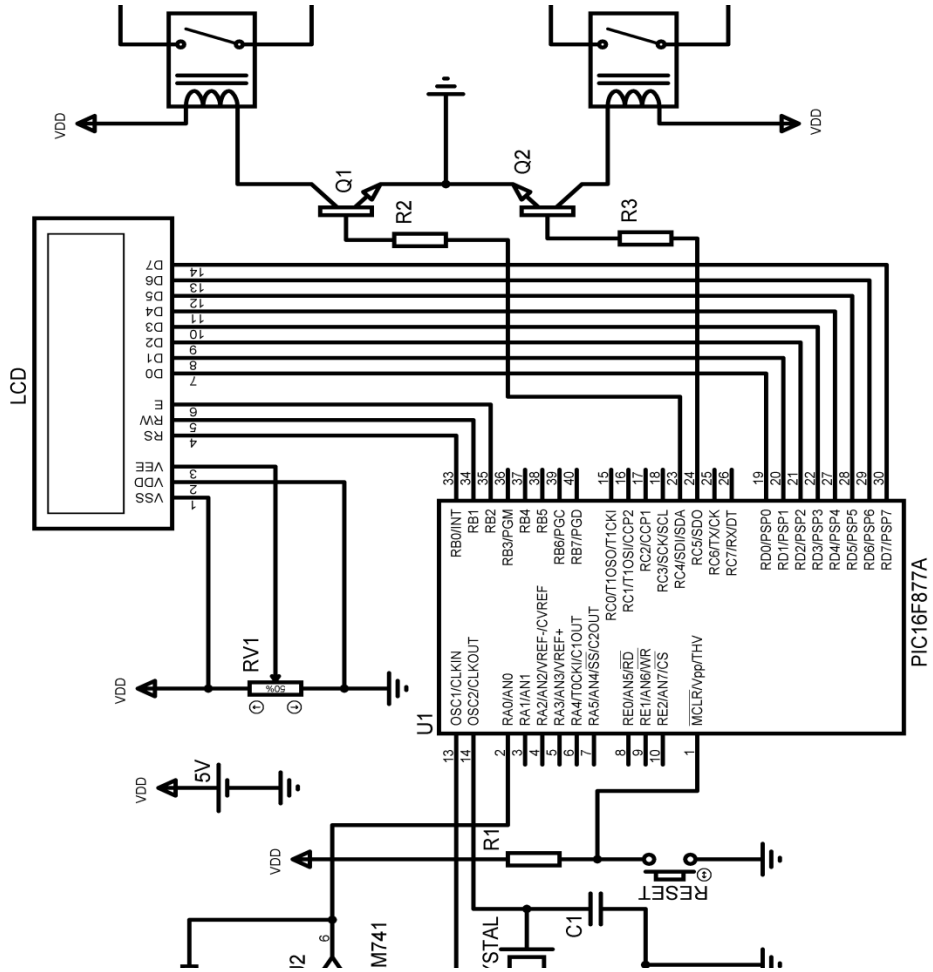
LM 35 ısı sensöründen (Şekil 2.11) gelen analog sinyaller A/D dönüştürücü tarafından dijital sinyallere dönüştürülerek mikrodenetleyici tarafından işlenen bilgi hâline getirilir. İşlenen bu sinyaller çıkış birimi olan LCD'ye sıcaklık değeri olarak yazdırılır. Bu değer daha evvel belirlenmiş olan sıcaklık aralığı (20ºC - 30ºC oda sıcaklığı) ile karşılaştırılır sıcaklık değeri belirlenen sıcaklık değerlerinin altında ise mikrodenetleyicinin Port C'nin 3 numaralı portuna bağlı olan ısıtıcı çalışır. Sıcaklık değeri bu aralık değerinin üstüne çıktığında Port C'nin 4 numaralı portuna bağlı olan fan çalışır. LCD ekranının ilk satırında "OLCULEN SICAKLIK" yazısı bulunmakta ikinci satırında ise LM 35 ile ölçülen sıcaklık değeri yer almaktadır.

2.4.1. Devrenin Malzemeleri

- PIC 16F877 4 Mhz mikrodenetleyici
- X1= 4Mhz kristal

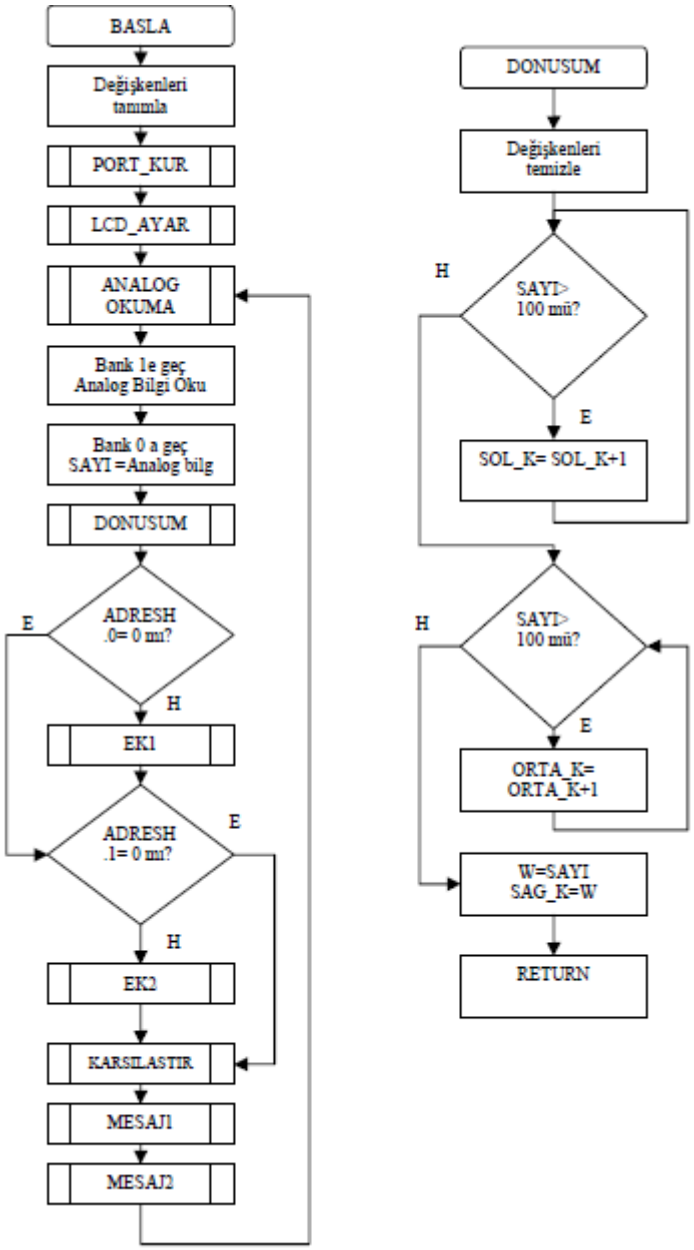
- C1 = C2 = 22pf
- R1= 10K, R2=R3=R5=1K, R4=3.9K
- Fan, Isıtıcı
- RV1= 1K potansiyometre
- Q1=Q2= BC237
- LM 35 Isı Sensörü, 741 OP-AMP
- 2x16 LCD, 5V Röle

2.4.2. Devrenin Şeması



Şekil 2.12: PIC16F877 ile yapılan ısıtıcı ve fan kontrol devresi

2.4.3. Akış Diyagramı



Tablo 2.8: Kontrol devresinin program akış diyagramı

2.4.4. Devrenin ASM Programı

;======ISITICI VE FAN KONTROL DEVRESİ ASM PROGRAMI==========

```
        LIST        P=16F877
        INCLUDE     "P16F877.INC"
#DEFINE RS          PORTB,0
#DEFINE RW          PORTB,1
#DEFINE EN          PORTB,2
SOL_K   EQU         H'10'
ORTA_K  EQU         H'11'
SAG_K   EQU         H'12'
SAYI    EQU         H'13'
TEMP    EQU         H'14'
TEMP1   EQU         H'20'
TEMP2   EQU         H'21'
DEG     EQU         H'22'
SAYAC1  EQU         H'23'
SAYAC2  EQU         H'24'
Y1      EQU         H'25'
Y2      EQU         H'0A'
Y3      EQU         H'0B'
D1      EQU         H'0C'
D2      EQU         H'0D'
D3      EQU         H'0E'

BASLA;=====================================================
        MOVLW       H'03'
        MOVWF       Y1                  ;Y1= 3
        MOVLW       H'00'
        MOVWF       Y2                  ;Y2=0
        MOVLW       H'00'
        MOVWF       Y3                  ;Y3=0
        MOVLW       H'02'
        MOVWF       D1                  ;D1=2
        MOVLW       H'00'
        MOVWF       D2                  ;D2=0
        MOVLW       H'00'
        MOVWF       D3                  ;D3=0
        CLRF        STATUS              ;STATUS temizle
        CLRF        PORTB               ;PORTB temizle
        CLRF        PORTD               ;PORTD temizle
        CLRF        PORTC               ;PORTC temizle
        CALL        PORT_KUR            ;Portları ayarla
        CALL        LCD_AYAR            ;LCD yi ayarla
DONGU
        CALL        ANALOG_OKUMA        ;Analog girişi kontrol et
```

```
        BSF     STATUS,5        ;Bank1 e geç
        MOVF    ADRESL, W       ;Analog bilgiyi oku
        BCF     STATUS,5        ;Bank 0 a geç
        MOVWF   SAYI            ;Analog giriş bilgisini SAYI ya ata
        CALL    DONUSUM         ;Okunan bilgiyi BCD'ye çevir
        BTFSC   ADRESH,0        ;9.bit "0" mı?
        CALL    EK1             ;Hayır, 9.biti ekle
        BTFSC   ADRESH,1        ;10. bit "0" mı?
        CALL    EK2             ;Hayır, 10. biti ekle
        CALL    KARSILASTIR     ;Karşılaştır alt programına git
        CALL    MESAJ1          ;LCD'ye yazı yaz
        CALL    MESAJ2          ;LCD'ye sıcaklığı yaz
        GOTO    DONGU

PORT_KUR;=============================================================
        BSF     STATUS, 5       ;Bank 1
        MOVLW   H'80'           ;Sayı sağa kaydırılır.
        MOVWF   ADCON1
        CLRF    TRISB           ;B portu çıkış
        CLRF    TRISD           ;D portu çıkış
        CLRF    TRISC           ;C portu çıkış
        BCF     STATUS, 5       ;Bank 0
        MOVLW   H'41'           ;AN0 aktif
        MOVWF   ADCON0          ;A/D çevrim aktif
        RETURN

ANALOG_OKUMA;=================================================
        BSF     ADCON0,GO       ;A/D çevrime başla
DON
        BTFSS   PIR1, ADIF      ;A/D çevrim bitti mi?
        GOTO    DON             ;Hayır geri dön
        RETURN

DONUSUM;======================================================
        CLRF    SOL_K           ;Sol karakteri sil
        CLRF    ORTA_K          ;Orta karakteri sil
        CLRF    SAG_K           ;Sağ karakteri sil
SOL
        MOVLW   H'64'           ;W →D'100'
        SUBWF   SAYI,W          ;W = SAYI- W
        BTFSC   STATUS,C        ;Sayı 100' den büyük mü?
        GOTO    SOL2            ;Evet SOL2 ye git
ORTA
```

```
        MOVLW   H'0A'           ;W →D'10'
        SUBWF   SAYI,W          ;W= SAYI -W
        BTFSC   STATUS,C        ;Sayı 10'dan büyük mü?
        GOTO    ORTA2           ;Evet ORTA2'ye git
        MOVF    SAYI,W          ;W = SAYI
        MOVWF   SAG_K           ;W yi sağ karaktere aktar
        RETURN
SOL2
        INCF    SOL_K,F         ;Sol karakteri bir artır
        MOVWF   SAYI            ;W'yi SAYI değişkenine aktar
        GOTO    SOL             ;SOL etiketine geri dön
ORTA2
        INCF    ORTA_K,F        ;Orta karakteri bir artır
        MOVWF   SAYI            ;W'yi SAYI değişkenine aktar
        GOTO    ORTA            ;ORTA etiketine geri dön

EK1;=====================================================
        MOVLW   H'06'           ;ADRESH 0. biti "1" ise
        ADDWF   SAG_K,F         ;Sayıya 256 eklenir.
        MOVLW   H'0A'           ;Sağ karaktere D'6' ekle
        SUBWF   SAG_K,W         ;Sağ karakter 10'dan büyük mü?
        BTFSS   STATUS,C
        GOTO    EKLE2           ;Hayır EKLE2'ye git
        MOVWF   SAG_K
        INCF    ORTA_K          ;Orta karakteri bir artır
EKLE2
        MOVLW   H'05'           ;Orta karaktere D'5' ekle
        ADDWF   ORTA_K,F        ;Orta karakter 10'dan büyük mü?
        MOVLW   H'0A'
        SUBWF   ORTA_K,W
        BTFSS   STATUS,C
        GOTO    EKLE3
        MOVWF   ORTA_K
        INCF    SOL_K           ;Sol karakteri bir artır
EKLE3
        MOVLW   H'02'           ;Sol karaktere D'2' sayısını ekle
        ADDWF   SOL_K
        RETURN


EK2;=====================================================
        MOVLW   H'02'           ;ADRESH 1. biti "1" ise
        ADDWF   SAG_K,F         ;Sayıya 512 ilave edilir
```

```
        MOVLW   H'0A'           ;Sağ karaktere D'2' eklenir
        SUBWF   SAG_K,W         ;Sayı 10'dan büyük mü?
        BTFSS   STATUS,C
        GOTO    EKLE4           ;Hayır EKLE4 değişkenine git
        MOVWF   SAG_K
        INCF    ORTA_K          ;Orta karakteri bir artır
EKLE4
        MOVLW   H'01'           ;Orta karaktere D'1' sayısını ilave et
        ADDWF   ORTA_K,F
        MOVLW   H'0A'           ;Orta karakter 10'dan büyük mü?
        SUBWF   ORTA_K,W
        BTFSS   STATUS,C
        GOTO    EKLE5           ;Hayır EKLE5 değişkenine git
        MOVWF   ORTA_K
        INCF    SOL_K           ;Sol karakteri bir artır
EKLE5
        MOVLW   H'05'           ;Sol karaktere D'5'sayısını ilave et
        ADDWF   SOL_K
        RETURN

LCD_AYAR;=============================================================
        CLRF    PORTB           ;PortB temizle
        CALL    GECIKME1        ;LCD nin açılması için bekle
        MOVLW   H'38'           ;8 bit , çift satır aktif
        CALL    LCD_YAZ
        CALL    LCD_SIL         ;LCD sil
        MOVL    W H'0C'
        CALL    LCD_YAZ         ;LCD on
        MOVLW   H'06'
        CALL    LCD_YAZ         ;Kursör- 1 artan mod
        RETURN

LCD_YAZ;==============================================================
        MOVWF   TEMP            ;TEMP ←W
        CALL    BF_TEST         ;BF bitini test et
        BCF     RW              ;RW →  0 Yaz
        BCF     RS              ;RS →0 Komut
        BSF     EN              ;E ←1
        MOVF    TEMP,W          ;W ←TEMP
        MOVWF   PORTD           ;LCD'ye gönder
        BCF     EN              ;E ←1
        RETURN
```

```
LCD_SIL;=============================================================
        MOVLW   H'01'
        CALL    LCD_YAZ         ;Ekranı temizle, kursör 1.satır 1.sütunda
        RETURN

BF_TEST;=============================================================
        BSF     STATUS,5        ;BANK 1
        MOVLW   H'FF'           ;W ←'FF'
        MOVWF   TRISD           ;PORT D giriş
        BCF     STATUS,5        ;BANK 0
        BCF     RS              ;RS →0 Komut
        BSF     RW              ;RW →  1 Oku
        BSF     EN              ;E ←1
        MOVF    PORTD,W         ;W←PORT D
        BCF     EN              ;E ←0
        MOVWF   TEMP1           ;TEMP1 ←W
        BTFSC   TEMP1,7         ;BF'yi kontrol et "0" mı?
        GOTO    BF_TEST         ;Hayır tekrar test et
        BCF     RW              ;RW →  0 Yaz
        BSF     STATUS,5        ;BANK 1
        MOVLW   H'00'           ;PORT D çıkış
        MOVWF   TRISD
        BCF     STATUS,5        ;BANK 0
        RETURN

LCD_YAZ 2;===========================================================
        MOVWF   TEMP2           ;TEMP2 ←W
        CALL    BF_TEST         ;BF yi kontrol et
        BCF     RW              ;RW →  0 Yaz
        BSF     RS              ;RS →1 Veri
        BSF     EN              ;E ←1
        MOVF    TEMP2,W         ;W ←TEMP2
        MOVWF   PORTD           ;LCD ye gönder
        BCF     EN              ;E ←0
        RETURN

MESAJ1;==============================================================
        MOVLW   H'80'           ;1. satırın 1.sütun aktif
        CALL    LCD_YAZ
        MOVLW   0
M1
        MOVWF   DEG             ;DEG =0
        CALL    MESAJ_VERI      ;Mesajı al
```

```
        ANDLW   H'FF'
        BTFSC   STATUS,Z        ;Mesajın sonu geldi mi?
        GOTO    MESAJ_SON       ;Evet geri dön
        CALL    LCD_YAZ2        ;Hayır LCD'ye karakter gönder
        MOVF    DEG, W          ;W ←DEG
        ADDLW   1               ;W = W +1
        GOTO    M1              ;M1 etiketine git
MESAJ_SON
        RETURN

MESAJ2;=================================================
        MOVLW   H'C0'           ;2. satırın 1. sütunu aktif
        CALL    LCD_YAZ
        MOVLW   H'0F'
        ANDWF   SOL_K,F
        MOVLW   D'0'            ;W ←0
        SUBWF   SOL_K,W         ;W= SOL_K -W
        BTFSS   STATUS,Z        ;SONUÇ "0 " mı?
        GOTO    SOL_YAZ         ;Hayır SOL_YAZ etiketine git
SOL_BOS
        MOVLW   H'20'           ;W←  20
        MOVWF   SOL_K           ;SOL_K ←W
        GOTO    GIT1            ;GIT1
SOL_YAZ
        MOVLW   H'30'           ;Sol karaktere h'30' ekle
        ADDWF   SOL_K,W         ;LCD'ye gönder
        CALL    LCD_YAZ2
GIT1
        MOVLW   H'30'           ;Orta karaktere h'30' ekle
        ADDWF   ORTA_K,W
        CALL    LCD_YAZ2        ;LCD'ye gönder
        MOVLW   H'2C'
        CALL    LCD_YAZ2        ;LCD'ye "," karakterini gönder
        MOVLW   H'30'           ;Sağ karaktere h'30' ekle
        ADDWF   SAG_K,W
        CALL    LCD_YAZ2        ;LCD'ye gönder
        MOVLW   ' '
        CALL    LCD_YAZ2        ;LCD'ye " º " karakterini gönder
        MOVLW   B'11011111'
        CALL    LCD_YAZ2
        MOVLW   'C'             ;LCD'ye "C " yaz
        CALL    LCD_YAZ2
        RETURN
```

```
GECIKME1;=================================================
        MOVLW   D'60'
        MOVWF   SAYAC1
A1
        MOVLW   D'50'
        MOVWF   SAYAC2
A2
        DECFSZ  SAYAC2,F
        GOTO    A2
        DECFSZ  SAYAC1,F
        GOTO    A1
        RETURN

KARSILASTIR;==============================================
        MOVF    Y1,W            ;Sıcaklığın yüksek sınırını kontrol et
        SUBWF   SOL_K,W         ;Sayının rakamları tek tek kontrol edilir
        BTFSC   STATUS,Z        ;Sol karakter üst sınıra eşit mi?
        GOTO    K1              ;Evet K1'e git
        BTFSS   STATUS,C
        GOTO    KK
        CALL    SOGUT           ;Sıcaklık fazla ise Fanı çalıştır
        RETURN
K1
        MOVF    Y2,W            ;Orta karakter ile refaransı
        SUBWF   ORTA_K,W        ;karşılaştır
        BTFSC   STATUS,Z        ;İki sayı birbirine eşit mi?
        GOTO    K2              ;Evet K2'e git
        BTFSS   STATUS,C
        GOTO    KK
        CALL    SOGUT           ;Sıcaklık fazla ise Fanı çalıştır
        RETURN
K2
        MOVF    Y3,W            ;Orta karakter ile refaransı
        SUBWF   SAG_K,W         ;karşılaştır
        BTFSS   STATUS,C        ;İki sayı birbirine eşit mi?
        GOTO    KK
        CALL    OGUT            ;Sıcaklık fazla ise Fanı çalıştır
        ETURN
KK
        MOVF    D1,W            ;Sıcaklığın düşük sınırını kontrol et
        SUBWF   SOL_K,W         ;Sayının rakamları tek tek kontrol edilir
        BTFSC   STATUS,Z        ;Sol karakter alt sınıra eşit mi?
```

```
        GOTO    K3              ;Evet K3 etiketine git
        BTFSS   STATUS,C
        CALL    ISIT            ;Sıcaklık düşük ise ısıtıcıyı çalıştır
        RETURN
K3
        MOVF    D2,W
        SUBWF   ORTA_K,W
        BTFSC   STATUS,Z
        GOTO    K4
        BTFSS   STATUS,C
        CALL    ISIT            ;Sıcaklık düşük ise ısıtıcıyı çalıştır
        RETURN
K4
        MOVF    D3,W
        SUBWF   SAG_K,W
        BTFSS   STATUS,C
        CALL    ISIT            ;Sıcaklık düşük ise ısıtıcıyı çalıştır
        RETURN

ISIT;=============================================================
        BSF     PORTC,1         ;Isıtıcıyı çalıştır
        RETURN

SOGUT;============================================================
        BSF     PORTC,2         ;Fanı çalıştır
        RETURN

MESAJ_VERI;=======================================================
        ADDWF   PCL ,1          ;PCL = PCL + W ile veriyi adresle
        RETLW   'O'
        RETLW   'L'
        RETLW   'C'
        RETLW   'U'
        RETLW   'L'
        RETLW   'E'
        RETLW   'N'
        RETLW   ' '
        RETLW   'S'
        RETLW   'I'
        RETLW   'C'
        RETLW   'A'
        RETLW 'K'
        RETLW 'L'
```

```
RETLW 'I'
RETLW 'K'
RETLW 0
MOVF        PCL,W                ;İstenilen karakteri W registerine al
RETURN
END
```

UYGULAMA FAALİYETİ

Şekil 2.12 deki uygulama devresini yapınız.

İşlem Basamakları	Öneriler
➢ Kurulacak sistem için ihtiyaçları tespit ediniz.	➢ Kullandığınız devre elemanlarının özelliklerini internetten araştırınız.
➢ İhtiyacı karşılayacak mikrodenetleyiciyi seçiniz.	➢ Mikrodenetleyici olarak PIC 16F877A kullanınız.
➢ Analog veri için gerekli hesaplamaları yapınız.	➢ Matemetiksel işlemleri dikkatli şekilde yapınız.
➢ Sistemin mikrodenetleyici programını yazınız.	➢ Programı yazdıktan sonra MPLAB ile PROTEUS programlarında deneyiniz.
➢ Programı mikrodenetleyiciye yükleyiniz.	➢ Programı mikrodenetleyiciye yüklerken kullanılan pic programlayıcıya uygun yazılım kullanınız.
➢ Çevre elemanları ve analog veri sağlayan elemanlar ile devreyi kurunuz.	➢ Devrenin montajını yapmadan önce breadboard üzerinde kurarak deneyiniz.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet,** kazanamadıklarınız için **Hayır** kutucuklarına (X) işareti koyarak öğrendiklerinizi kontrol ediniz

Değerlendirme Ölçütleri		Evet	Hayır
1.	Kurulacak sistem için ihtiyaçları tespit edebildiniz mi?		
2.	İhtiyacı karşılayacak mikrodenetleyiciyi seçebildiniz mi?		
3.	Analog veri için gerekli hesaplamaları yapabildiniz mi?		
4.	Sistemin mikrodenetleyici programını yazabildiniz mi?		
5.	Programı mikrodenetleyiciye yükleyebildiniz mi?		
6.	Çevre elemanları ve analog veri sağlayan elemanlar ile devreyi kurabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda "Hayır" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "Evet" ise "Ölçme ve Değerlendirme"ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1.( ) PIC 16F84 2 porta, 16F877 ise 5 porta sahiptir.

2.( ) PIC 16F877 8K flash program belleğine, PIC 16F84 1K flash program belleğine sahiptir

3.( ) PIC 16F877 40MHz' e kadar osilatör kullanılır.

4.( ) PIC 16F877 A/D Birimine sahip değildir.

5.( ) Akış diyagramı program yazmak için kullanabileceğimiz temel bir yöntemdir.

6.( ) PWM yöntemi motor hız kontrolünde kullanılmaz.

7.( ) LM35 ısı sensörüdür.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ–3

AMAÇ

Picbasic pro programı ile programlama yapabileceksiniz.

ARAŞTIRMA

- Picbasic ile Pic C arasındaki farkları araştırınız.
- Picbasic programlama dilinin komutlarını araştırınız.
- Picbasic pro derleyicisinin özelliklerini ve kullanımını araştırınız.
- Araştırma işlemleri için internet ortamından yararlanabilirsiniz.

3. PIC BASIC PRO İLE PROGRAMLAMA

Assembly diliyle yazılan program uzadıkça hem karmaşıklaşır hem de zorlaşır. Bundan dolayı alternatif programlama dilleri geliştirilmiştir. Bu programlar basic ve C tabanlı olmak üzere ikiye ayrılır. Basic tabanlı olan program "PicBasic Pro", C tabanlı olan da "Pic C" programlarıdır.

PicBasic PIC mikrodenetleyicileri programlamak için Micro Engineering Labs firması tarafından geliştirilen Qbasic programlama diline benzeyen kod geliştirme aracıdır. PIC mikroassembler ile sayfalarca kod yazmaktansa, üst seviyeli derleyicilerle (Picbasic, pic C) çalışmak, programcıya hem zaman kazandıracak hem de programı sadeleştirecektir. Picbasic Pro ile LCD sürme, A/D çevirimi, PWM üretme gibi zor olan işleri zahmetsizce yürütmeyi sağlamaktadır.

3.1. Programlama Kuralları

- Etiketlerden sonra mutlaka (:) iki nokta üst üste kullanılmalıdır.
- Değişkenler "VAR" sözcüğü ile tanımlanır. Değişkenler "bit", "byte" ve "word" tipinde olabilir. Değişkene istediğimiz ismi verebiliriz.
- Sabitler "CON" sözcüğü ile tanımlanır. Sabite programda başka değer atanmaz.
- Stringler " " tırnak arasına yazılır.
- Sabitler, desimal (100), binary (%101), heksadesimal ($100) olarak kullanılır.
- Açıklama satırı "REM" veya (') tek tırnak ile başlar.
- Birden fazla komutu aynı satırda kullanılırken araya (:) iki nokta üst üste konur.
- Portlar PORTA= %10001111, PortA= $8F şeklinde tanımlanır.
- Pinler, PORTB.1= 1 şeklinde tanımlanır.
- A portunun tüm bacakları çıkış ise → TRISA = %00000000 tanımlanır.

- B portunun tüm bacakları giriş ise → TRISB = %11111111 tanımlanır.
- C portunun tek sayılı bacaklarını giriş, çift sayılı bacaklarını çıkış yapmak için →TRISC = %1010101010 şeklinde tanımlanır.
- A portunun 0 nu'lı bitini çıkış yapmak için →TRISA.0 = 0 şeklinde tanımlanır.
- Ek dosya yüklemek için INCLUDE komutu →INCLUDE "modedefs.bas" şeklinde tanımlanır.

3.1.1. Karşılaştırma Operatörleri

Karşılaştırma operatörleri IF…THEN ifadesiyle beraber kullanılır.

Karşılaştırma Operatörleri	Açıklama
= veya ==	Eşittir
<> veya !=	Eşit değil
<	Küçük
>	Büyük
<=	Küçük eşit
>=	Büyük eşit

Tablo 3.1: Karşılaştırma operatör açıklamaları

Örnek:

```
If sayac > 6 then son ' Sayac 6 sayısından büyükse son etiketine git.
```

3.1.2. Aritmetik Operatörler

Aritmetik Operatör	*Açıklama*
+	Toplama
-	Çıkarma
*	Çarpma
**	Çarpmanın üst 16 bitini elde etme
*/	Çarpmanın ortadaki 16 bitini elde etme
/	Bölme
<<	Sola Kaydırma
>>	Sağa Kaydırma
&	Bit AND'leme
|	Bit OR'lama
~	Bit NOT'lama

Tablo 3.2: Aritmetik operatör açıklamaları

Örnek:

```
X = Y << 2 ' Y değişkenini 2 bit sola kaydır ve X değişkenine ata
```

X = X / 10 ‘ X değişkenini 10’a böl ve X değişkenine ata

ABS

Sayının mutlak değerini alır.
X = ABS Y

SQR

Sayının karekökünü alır.
X = SQR Y

MAX – MIN

İki sayıdan büyük veya küçük olanını bulmak için kullanılır.
X = Y MAX 100 ‘Y değişkeni ile 100 değişkeni arasından büyük olanı X’e ata
X= Y MIN 10 ‘Y değişkeni ile 10 değişkeni arasından küçük olanı X’e ata

COS

Radyan cinsinden verilen bir açının kosinüsünü bulur. Elde edilen sonuç 8 bittir. 0 – 359 º olarak bilinen açıların karşılığı, 0-255 arasında binary sayılarla gösterilir.

Örneğin, 90º = 64, 180º = 128’dir.
Sonuç = (Derece/360)*256 →formülü ile hesaplanır.

SIN

Radyan cinsinden verilen bir açının sinüsünü bulur. Elde edilen sonuç 8 bittir.

DCD

8 veya 16 bitlik binary bir sayı üzerine istenilen bir bitin değerini “1” yapar. Diğer bitlerin değeri “0” olur.

X= DCD 5 ‘ X değişkenine 00001000 sayısını yerleştirir.

DIG

Desimal bir sayının istenilen bir bitini elde etmeye yarar.
X= 593
X1= X DIG 0 ‘ X1 içeriği “5” olur.

NCD

Binary bir sayının içerisindeki en soldaki “1”in sırası kaç ise bu sayıyı elde eder. Eğer hiçbir dijit “1” değilse sonuç “0” dır.
X = NCD %10001001 ‘ X değişkenine 7 sayısı atanır.

REV

Binary bir sayının içerisinde belirtilen bir bitten sağa doğru tüm bitlerin tersini alır. Bitlerin numaralandırılması 1 den başlar.

X = %01100100

Y = X REV 4 ‘ Y = %01101011 olarak atanır.

3.2. Karar Verme ve Döngü İşlemleri

3.2.1. GOTO Komutu

Koşulsuz dallanma komutudur. GOTO komutundan sonra etiket yazılır ve program etiketin bulunduğu konumdan devam eder.

Kullanımı

GOTO devam ‘devam etiketine git.

devam:

3.2.2. IF… THEN Komutu

Koşullu dallanma komutudur. IF… THEN karşılaştırılan ifadelerin doğru ya da yanlış olduğunu değerlendirerek farklı işlemleri gerçekleştirmek için kullanılır. IF komutundan sonra “şart” yazılır, şart doğruysa THEN komutundan sonraki işlem gerçekleştirilir, yanlışsa bir alt satıra geçilir.

Kullanımı:

IF şart THEN Etiket (İşlem)

IF…THEN kullanımının dışında ELSE ve ENDIF komutlarıyla grup hâlinde çalışması da kullanılabilir.

Kullanımı:

```
IF şart THEN
        İşlemler…..
ELSE
        İşlemler……
ENDIF
```

Örnek:

IF….THEN komutunun kullanıldığı bir örnek programdır. PortA.0’ıncı portuna bağlı

olan bir butona basıldığında PortB.1'inci portundaki LED'i yakan programdır.

```
'Program1.bas
'PIC : 16F84

        TRISA.0=1                          'PortA 0.biti giriş
        TRISB = %00000000                  'PortB çıkış
        PORTB = 0                          'LED'i söndür
DON:
        IF PORTA.0 = 0 THEN PORTB.1 = 1    ' Butona basılı ise LED'i yak
        GOTO DON                           'Butona basılı değilse DON etiketine git
        END
```

3.2.3.BRANCH Komutu

Atanan değişkene bağlı olarak belirlenen etiketlerden birine dallanmayı sağlar. Eğer değişken "0" ise program ilk etikete allanır, "1" ise ikinci etikete dallanır. BRANCH komutuyla en fazla 256 etiket yazılabilir.

Kullanımı:

BRANCH değişken , [Etiket0, Etiket1, Etiket2….]

3.2.4. FOR…NEXT Komutu

Döngü düzenleme komutudur. FOR…NEXT arasına yazılan program parçasını istenilen sayıda tekrar etmek için kullanılır.

Kullanımı:

```
FOR sayaç = Başlangıç TO bitiş
.
        Komutlar
.
NEXT
```

Herhangi bir değişiklik yapılmazsa döngü her döndüğünde sayaç "1" artar. Adım miktarını değiştirmek için STEP komutu kullanılır. STEP komutundan sonra belirlenen sayıya göre döngünün dönüş adımı belirlenir.

Kullanımı:

FOR Sayaç = Başlangıç TO bitiş STEP adım

Komutlar

NEXT

3.2.5. WHILE… WEND Komutu

Şart doğru olduğu sürece WHILE…WEND arasındaki işlemler çalıştırılır. Şart yanlış ise WEND komutundan sonraki satırdan program devam eder.

Kullanımı:

WHILE şart

İşlemler…….

WEND

3.3. PBP Komutları

3.3.1. PAUSE Komutu

Programı belirlenen süre miktarınca durdurmak için kullanılır. PAUSE komutundan sonra yazılan değişken kadar "milisaniye" bekler. Maksimum bekleme süresi 16 bitlik bir değişken olduğundan 65535 ms'dir.

Kullanımı:

PAUSE Süre

PAUSE	500	'500 ms'lik gecikme yap.

3.3.2. PAUSEUS Komutu

Programı mikrosaniye süresince durdurur. Maksimum bekleme süresi 65535 μs'dir.

Kullanımı:

PAUSEUS Süre

PAUSEUS	500	'500 μs'lik gecikme yap.

3.3.3. GOSUB…RETURN Komutu

Alt programlara dallanmak için kullanılır. Alt program RETURN komutunu görünceye kadar çalışmasına devam eder. RETURN komutundan sonra program kaldığı yerden devam eder.

Kullanımı:

```
GOSUB Etiket

GOSUB Kesme
      Kesme:
RETURN
```

3.3.4. Örnek Programlar

Yukarıda anlatılan komutları aşağıdaki örnekleri uygulayarak pekiştirebilirsiniz.

3.3.5. LED Flaşör Devresi

PortB nin 3. bitine bağlı olan LED'i 1sn aralıklarla yakıp söndüren uygulama devresidir. Bu devrede hem 16F84 hem de 16F877 entegresini kullanabilirsiniz. Sure CON 1000 'Sure değişkenine 1000 sayısını ata.

```
        TRISB.3 = 0                 ' PortB nin 3. bitini çıkış yap
DON:
        PORTB.3 = 1                 'LED'i yak
        PAUSE       Sure            '1sn bekle
        PORTB.3 = 0                 'LED'i söndür
        PAUSE       Sure            '1sn bekle
        GOTO        DON             'DON etiketine git
        END
```

3.3.6. Sayıcı Uygulama Devresi

16F877 entegresinin D portuna bağlı olan 8 LED ile 0–255 arası binary sayıcı uygulama programıdır.

```
        SAYAC  VAR  BYTE            'SAYAC değişkenini tanımla
        TRISD = %00000000           'PORTD tüm portları çıkış
        PORTD = %00000000           'PORTD'yi temizle
DON:
        FOR   SAYAC = 0   TO   255  ' 0 dan 255'e kadar dön
        PORTD = SAYAC               ' SAYAC değişkenini çıkışa aktar
        PAUSE       500             '500ms bekle
```

```
NEXT                                ‘ SAYAC 255 değilse geri don
GOTO        DON                     ‘DON etiketine git
END
```

3.3.7. Karaşimşek Uygulama Devresi

16F877’nin B portuna bağlı olan tüm LED’leri sırasıyla sağa ve sola kaydıran programdır.

```
        SAYAC VAR BYTE              ‘SAYAC değişkenini tanımla
        TRISB = %00000000           ‘PORTB çıkış
        PORTB = %00000000           ‘PORTB tüm LED’leri sönük
        SAYAC = %00000001           ‘ SAYAC değişkenine “1” sayısını yükle
SAG:
        PORTB = SAYAC
        PAUSE       500
        SAYAC = SAYAC << 1
        IF  SAYAC.7 = 1  THEN  SOL
        GOTO        SAG
SOL:
        PORTB = SAYAC
        PAUSE       500
        SAYAC = SAYAC >> 1
        IF  SAYAC.0 = 1  THEN  SAG
        GOTO        SOL
        END
```

3.3.8. LCD Uygulama Devresi

LCD’ye veri göndermek için LCDOUT komutu kullanılır.

- “# “ işareti kullanılarak LCD’ye veri gönderilirse bu sayının ASCII karşılığı LCD’ye yazdırılır.
- Bir LCD’ye veri göndermeden önce en az 0.5 sn beklemek gerekir.
- LCD’ye komut göndermek için $FE kullanılır.
- LCD’ye yazı yazmak için “ “ kullanılır.

Komut	Görevi
$FE, 1	Ekranı siler.
$FE, 2	Kursörü satır başına alır.
$FE, $0C	Kursörü görünmez hâle getirir.
$FE, $0E	Kursörü altçizgi biçimine alır.
$FE, $0F	Kursörü yanıp/söner yapar.
$FE, $10	Kursörü bir karakter sola alır.

$FE, $14	Kursörü bir karakter sağa alır.
$FE, $C0	Kursörü ikinci satırın başına alır.

Tablo 3.3: LCD komutları

LCD'ler 8 bitlik ve 4 bitlik veri yolu ile mikrodenetleyiciye bağlanabilir. (Ayrıntılı bilgi için Mikrodenetleyiciler ile Dijital Uygulamalar modülünü inceleyiniz.) LCDOUT komutu yalnızca LCD'ye veri yazmak için kullanılacaksa R/W bacağı toprağa bağlanır. LCD'nin bağlantı ayarlarını değiştirmek için DEFINE komutu kullanılır. Bu tanımlama programın başında yapılır. Aşağıdaki tanımlamalarda koyu renkli kısımlar değiştirilemez, açık renkli olan kısımlar ise programcı tarafından değiştirilebilir.

- **DEFINE LCD_DREG** PORTB 'LCD data portunu belirler.
- **DEFINE LCD_DBIT** 4 '4 bitlik veri yolu kullanıldığında data 'bitlerin başlangıcını belirler.(0 veya 4)
- **DEFINE LCD_RSREG** PORTB 'RS'nin bağlanacağı portu belirler.
- **DEFINE LCD_RSBIT** 1 'RS'nin bağlanacağı portun bitini belirler
- **DEFINE LCD_EREG** PORTB 'E'nin bağlancağı portu belirler.
- **DEFINE LCD_EBIT** 0 'E'nin bağlanacağı portun bitini belirler.
- **DEFINE LCD_RWREG** PORTB 'RW'nin bağlanacağı portu belirler.
- **DEFINE LCD_RWBIT** 2'RW'nin bağlanacağı portun bitini belirler.
- **DEFINE LCD_BITS** 4 'LCD veri yolu 4 bit veya 8 bit.
- **DEFINE LCD_LINES** 2 'LCD satır sayısını belirler.
- **DEFINE LCD_COMMANDUS** 2000 'Komut gecikme süresi µs cinsinden.
- **DEFINE LCD_DATAUS** 50 'Veri gecikme süresi µs cinsinden.

LCD ekranının ilk satırına "ELEKTRONIK", ikinci satırına da "ERKAN AVCI EML" yazan programı yazınız.

```
    DEFINE LCD_DREG      PORTD     'LCD data portunu = ERKAN AVCI
EML" yazan program aşağıda verilmiştir.
    DEFINE LCD_DBIT      4         '4 bitlik veri yolu kullanıldığında data
    DEFINE LCD_RSREG     PORTB     'RS = PORTB
    DEFINE LCD_RSBIT     1         'RS = PORTB.1
    DEFINE LCD_EREG      PORTB     'E = PORTB
    DEFINE LCD_EBIT      0         'E = PORTB.0
    DEFINE LCD_BITS      8         'LCD veri yolu 8 bit.
    DEFINE LCD_LINES     2         'LCD satır sayısı = 2
    TRISD = %00000000
    TRISB = %00000000

DON:
    LCDOUT     $FE,1               'LCD ekranini temizle
```

```
PAUSE       500                 '500ms bekle
LCDOUT      "ELEKTRONIK"        '1. satıra ELEKTRONIK yaz
PAUSE       500                 '500ms bekle
LCDOUT      $FE, $C0            'Kursörü 2. satırın başına al
LCDOUT      "ERKAN AVCI EML"    '2. satıra veriyi yaz
PAUSE       100                 '100ms bekle
GOTO        DON                 'DON etiketine git
END
```

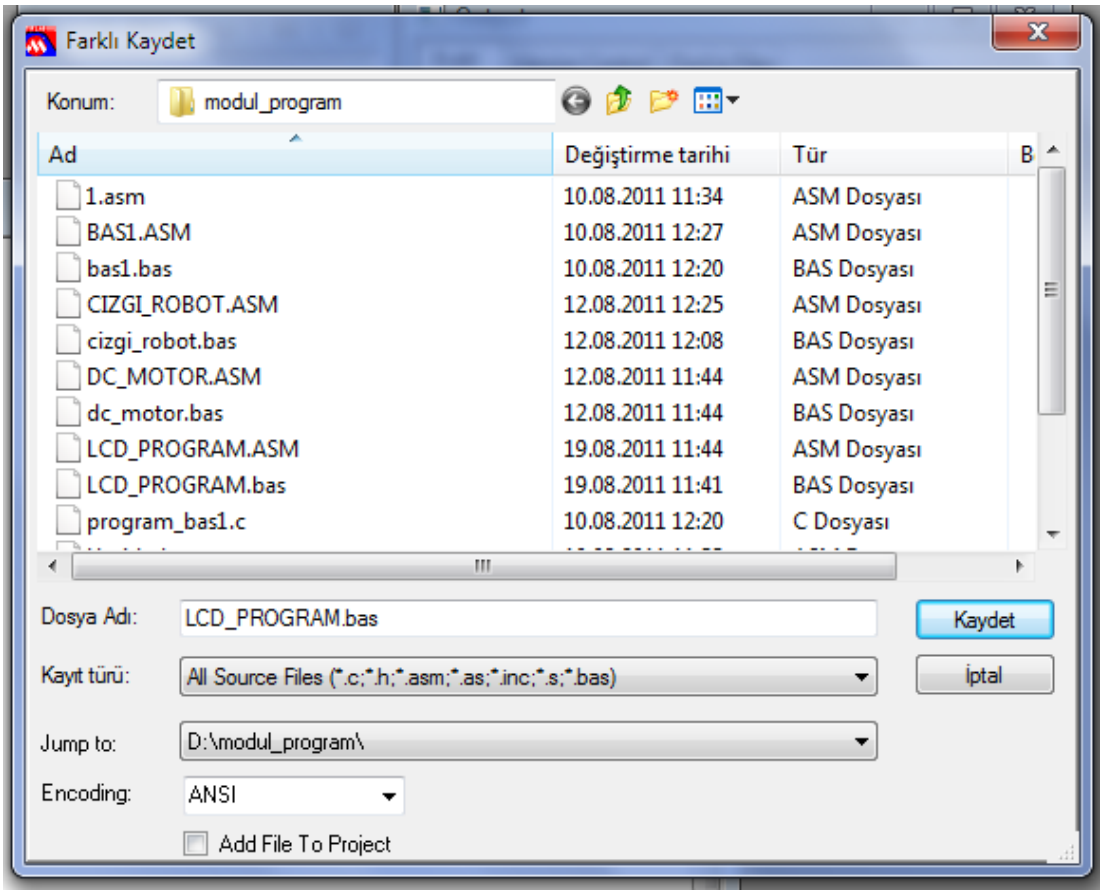
3.4. Pıc Basıc Pro Programının Kullanımı

PBP uygulaması üç aşamada gerçekleştirilir.

- BAS uzantılı program dosyası oluşturulur.
- BAS dosyası PicBasic Pro ile derlenerek HEX dosyasına dönüştürülür.
- HEX dosyası PIC'e yüklenir. (Mikrodenetleyiciler ile Dijital İşlemler modülünü inceleyiniz.)

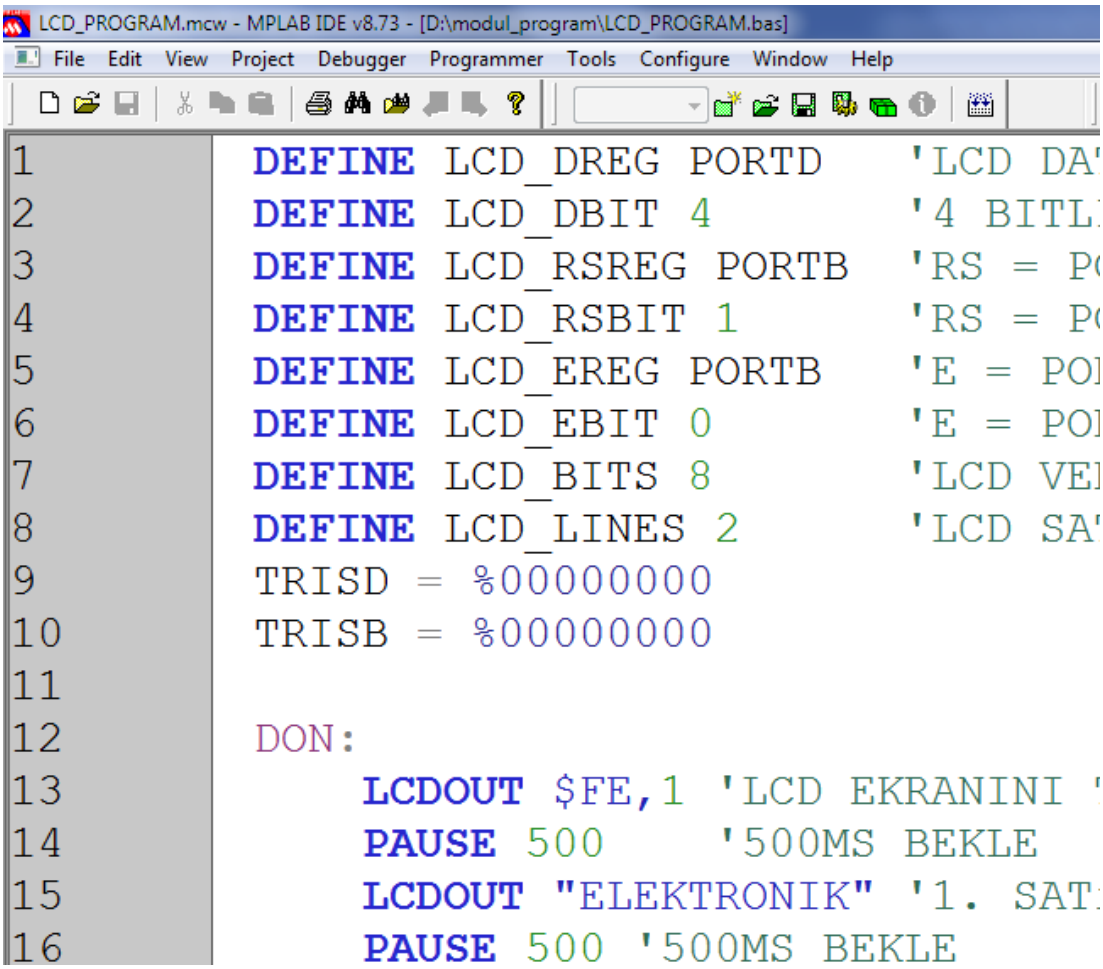
3.4.1. BAS Dosyasının Oluşturulması

PicBasic Pro derleyici komutlarını yazıp ".BAS" uzantılı bir dosya oluşturmak için bir metin editörü kullanılır. Metin editörü olarak NOTEPAD veya MPLAB programları kullanılabilir. Bu modüldeki uygulamalarda NOTEPAD programı kullanılmıştır. Kodlar yazıldıktan sonra DOSYA_ADI.BAS olarak kaydedilir. Burada dikkat edilmesi gereken konu dosyanın PBP klasörüne kaydedilmesidir. Şekil 3.1, 3.2 ve 3.3' de MPLAB ile yazılma işlemleri görülmektedir.



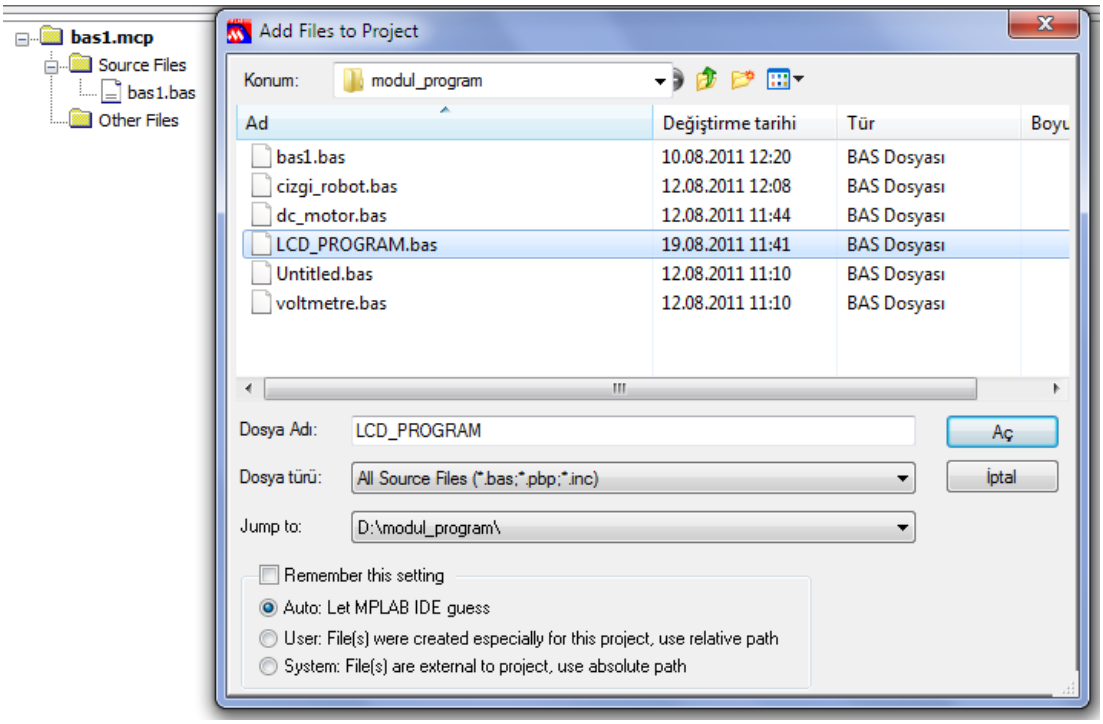
Şekil 3.1: Programın BAS uzantılı olarak kaydedilmesi

MPLAB' da çalışma yapılırken açılan çalışma sayfası (WORK SPACE) farklı kaydet (Save As) seçilerek bas uzantılı (LCD_PROGRAM.bas) olarak kaydedilir.



Şekil 3.2: Programın yazılması

Açılan pencerede program picbasic pro komutlarına göre yazılır. Yazıldıktan sonra kaydedilir.

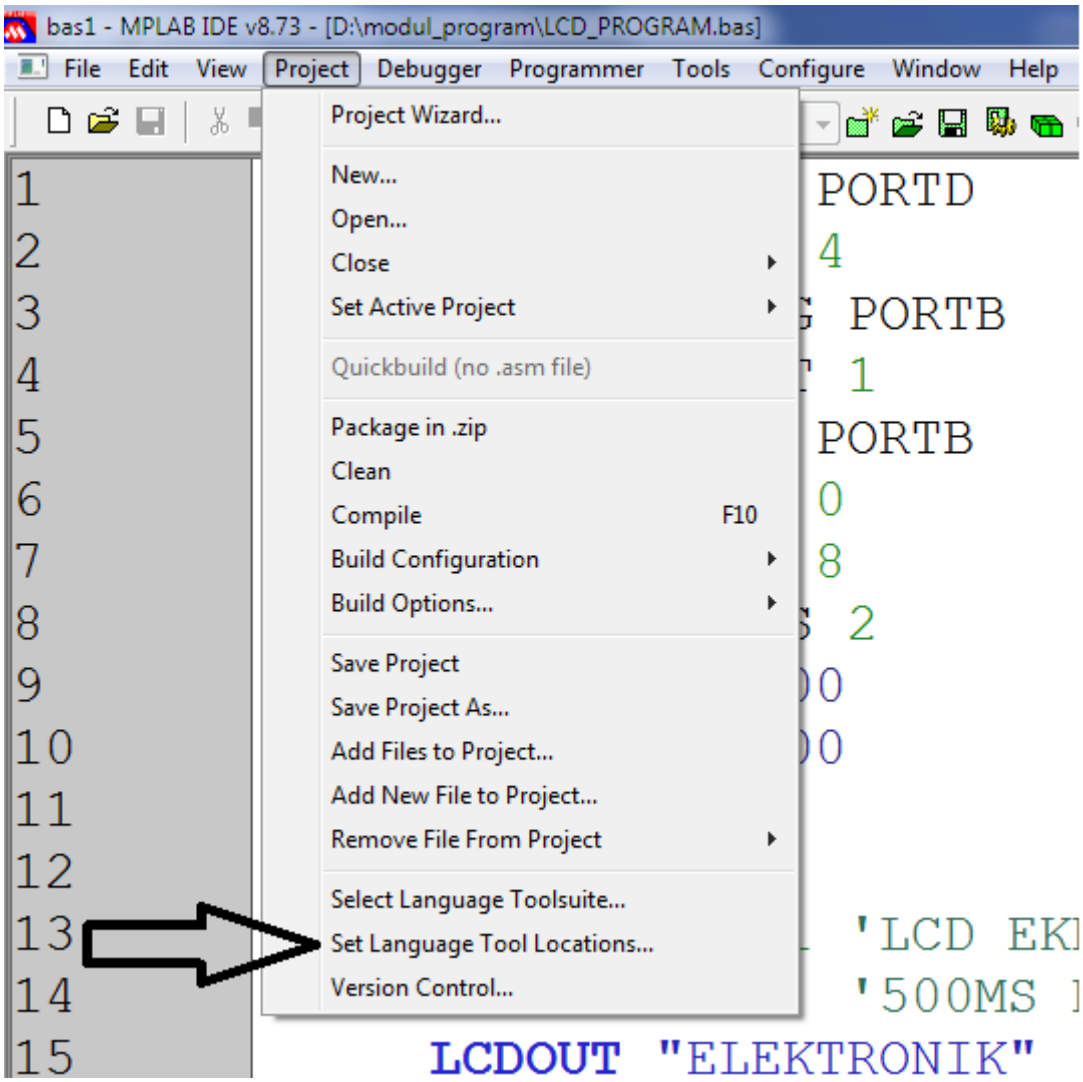


Şekil 3.3: Yazılan program oluşturalan projeye eklenmesi

Yazılan program, oluşturulmuş olan projenin Source Files kısmına eklenir.

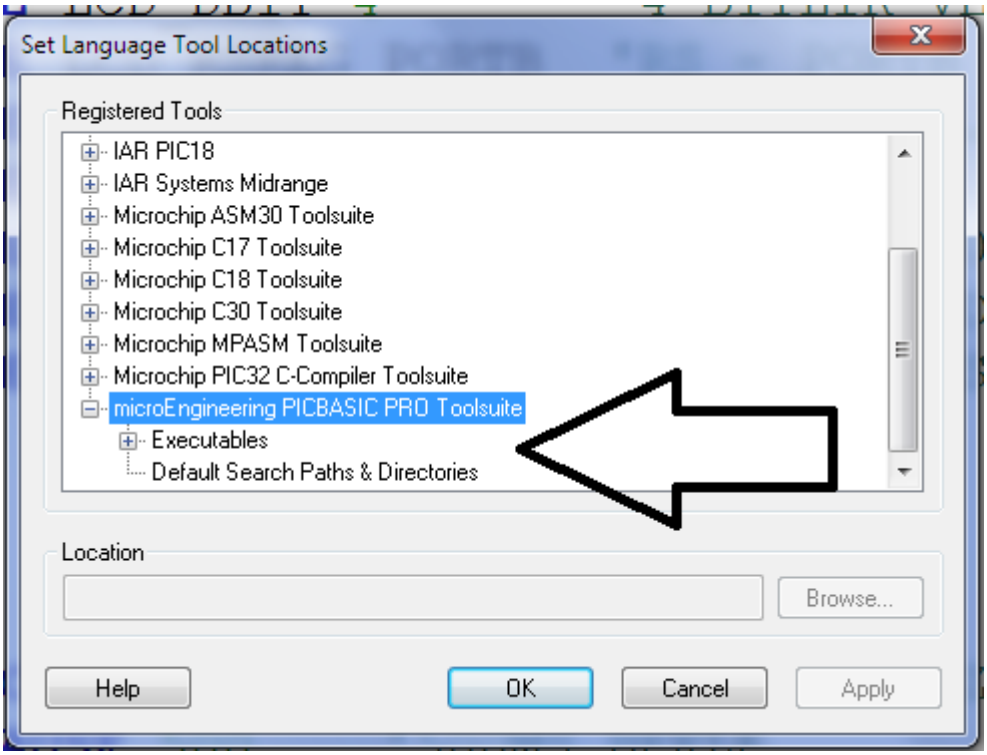
3.4.2. BAS Dosyasının Derlenmesi

Derleme (Compile) işlemi Basic dilinde yazılan programın makine diline çevrilmesi (hex uzantılı dosyalar) işlemidir. Derleme sonucunda asm ve hex uzantılı dosyalar oluşur. Bu işemin yapılabilmesı için proje dosyası oluştulmalıdır.



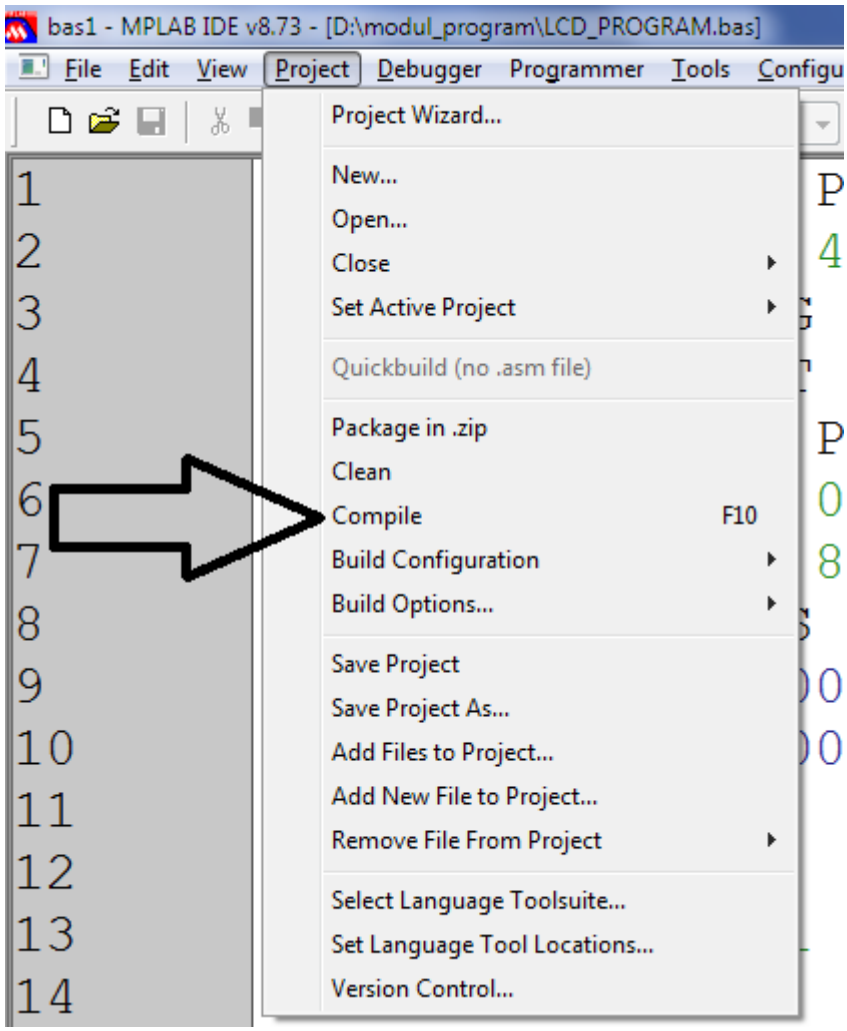
Şekil 3.4: Derleyici programın seçilmesi

Project menüsü seçilir. Set Language Tool Locations seçeneğinden derleyici olarak önceden kurulmuş olan PicBasic Pro pragramı seçilir.



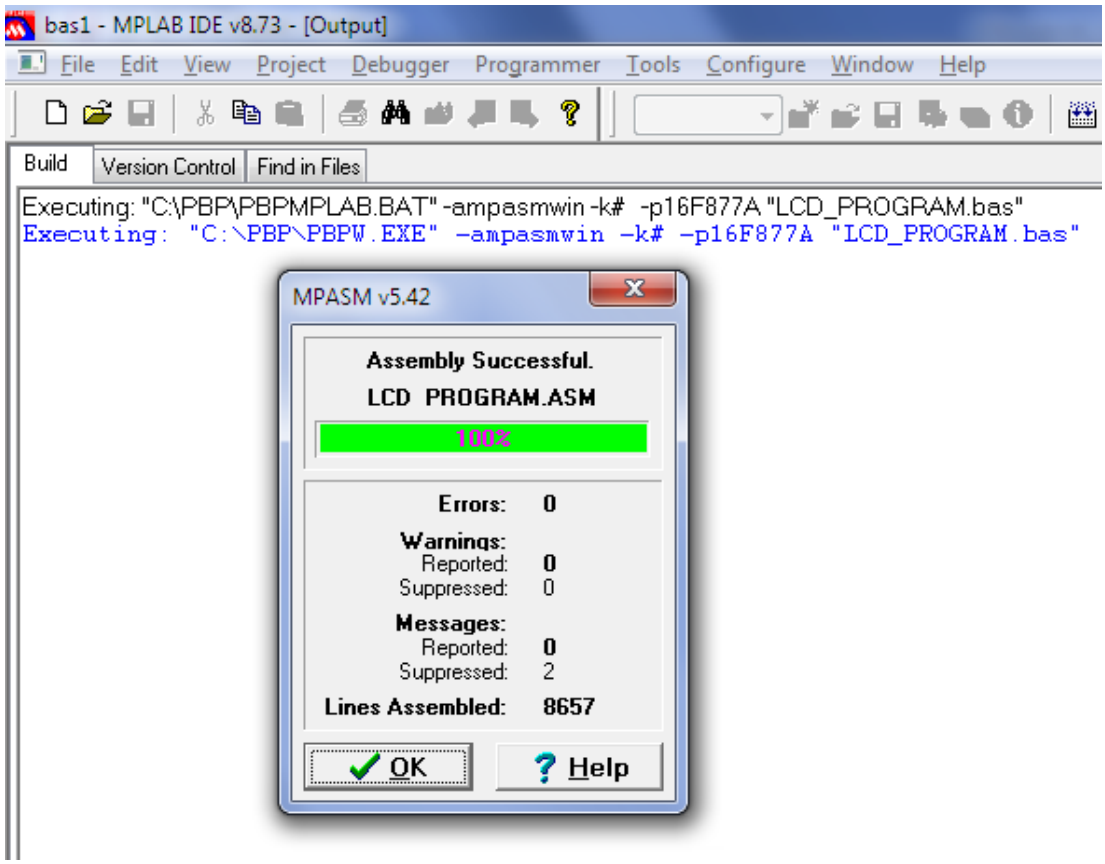
Şekil 3.5: Derleyici programı

Bu işlem gerçekleştirildikten sonra programın derlenmesi Project menüsünden Compile seçeneği seçilerek ya da F10 kısayol tuşuna basılarak yapılır.



Şekil 3.6: Programın derlenmesi

Derleme işlemi sonucu başarılı ise yani herhangi bir yazım hatası yoksa Şekil 3.7'deki ekran çıktısı alınır.



Şekil 3.7: Programın başarılı derlenmesi

UYGULAMA FAALİYETİ

3.3.5 - 3.3.6 - 3.3.7 - 3.3.8 başlıkları altındaki örnekleri uygulayınız.

İşlem Basamakları	Öneriler
➢ Kurulacak sistem için ihtiyaçları tespit ediniz.	➢ Kullandığınız devre elemanlarının özelliklerini internetten araştırınız.
➢ İhtiyacı karşılayacak mikrodenetleyiciyi seçiniz.	➢ Mikrodenetleyici olarak PIC 16F877A kullanınız.
➢ Sistemin mikrodenetleyici programını PIC Basic Pro ile yazınız.	➢ Programı yazdıktan sonra MPLAB ile PROTEUS programlarında deneyiniz.
➢ Programı mikrodenetleyiciye yükleyiniz.	➢ Programı mikrodenetleyiciye yüklerken kullanılan pic programlayıcıya uygun yazılım kullanınız.
➢ Çevre elemanları ve analog veri sağlayan elemanlar ile devreyi kurunuz.	➢ Devrenin montajını yapmadan önce breadboard üzerinde kurarak deneyiniz.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet,** kazanamadıklarınız için **Hayır** kutucuklarına (X) işareti koyarak öğrendiklerinizi kontrol ediniz.

	Değerlendirme Ölçütleri	Evet	Hayır
1.	Kurulacak sistem için ihtiyaçları tespit edebildiniz mi?		
2.	İhtiyacı karşılayacak mikrodenetleyiciyi seçebildiniz mi?		
3.	Analog veri için gerekli hesaplamaları yapabildiniz mi?		
4.	Sistemin mikrodenetleyici programın yazabildiniz mi?		
5.	Programı mikrodenetleyiciye yükleyebildiniz mi?		
6.	Çevre elemanları ve analog veri sağlayan elemanlar ile devreyi kurabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda "Hayır" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "Evet" ise "Ölçme ve Değerlendirme"ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1.( ) Assebly dilinde yazılan programlar uzadıkça karmaşıklaştığından dolayı PicBasic Pro gibi yazılım dilleri kullanılır.

2.( ) PicBasic dilinde karşılaştırma komutları yoktur.

3.( ) PicBasic dilinde binari ifadeler " % " işareti ile kullanılır.

4.( ) PicBasic dilinde açıklama yapılacak ise " karakteri kullanılır.

5.( ) GOTO komutu hem PicBasic hem Assebly dilinde aynı işlemi yapar.

6.( ) PicBasic dili LCD' ye bilgi yazmada bize kolaylık sağlar.

7.( ) MPLAB programında PicBasic dili ile yazılan program derlenemez.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ–4

AMAÇ

Picbasic pro programı ile uygulama devreleri yapabileceksiniz.

ARAŞTIRMA

- PBP ile yapılan termometre ve nemölçer uygulama devrelerini araştırınız.
- Araştırma işlemleri için internet ortamından yararlanabilirsiniz.

4. PIC BASIC İLE UYGULAMA DEVRELERİ

4.1. Voltmetre Uygulama Devresi

Öğrenme Faaliyeti 2' de anlattığımız gibi PIC 16F877 entegresi 10 bitlik 8 adet analog girişe sahiptir. PicBasic ile analog girişleri kontrol etmek için ADCIN komutu kullanılır.

Kullanımı:

ADCIN kanal, değişken

ADCIN komutu belirlenen kanaldan okuma yapar ve sonucu belirtilen değişkene aktarır.

ADCIN komutunun kullanılabilmesi için aşağıdaki işlemler yapılır.

- İstenilen bacakların giriş yapılması gerekir. Bunun için TRIS registeri kullanılır.
- İstenilen bacakların analog giriş için kullanılması için ADCON1 registeri kullanılır.
- ADCON1'in 7. biti ile sonucun sağa ya da sola hizalı olması belirlenir.
- 8 bitlik bir A/D çevrimde sonuç sola hizalı olmalı ve ADCON1.7 = 0 yapılmalıdır.
- 10 bitlik bir A/D çevrimde sonuç sağa hizalı olmalı ve ADCON1.7 = 1 yapılmalıdır.
- ADCIN ile yaplan okumada sayısal veriye dönüştürme işlemlerinde DEFINE ile tanımlamalar yapılmalıdır.
- **DEFINE ADC_BITS** 8'A/D çevrim çözünürlüğü (8 /10 bit)
- **DEFINE ADC_CLOCK** 3'Saat kaynağını ayarlar. (RC = 3)
- **DEFINE ADC_SAMPLEUS** **50** 'Örnekleme sayısıdır. Birimi µs dir.

4.1.1. Devrenin Malzemeleri

- PIC 16F877 4 Mhz mikrodenetleyici
- C1 = C2 = 22pf , CRYSTAL= 4Mhz kristal
- R1= 10K, RV1= RV2 = 1K potansiyometre
- 2x16 LCD, Buton

Analog giriş olarak AN0 kanalı kullanılmıştır. Okunan analog bilgi sayı değişkenine aktarılmaktadır. Sayı değişkeni word tipinde tanımlandığından 16 bittir. Sayı değişkenin yüksek bitlerine ADRESH değişkeni aktarılır, düşük bitlerine ise ADRESL değişkeni aktarılır. Programda sağa kaymalı okuma yapıldığından okunan analog bilginin yüksek bitleri ADRESH'da, analog bilginin düşük bitleri ADRESL'de kaydedilir.

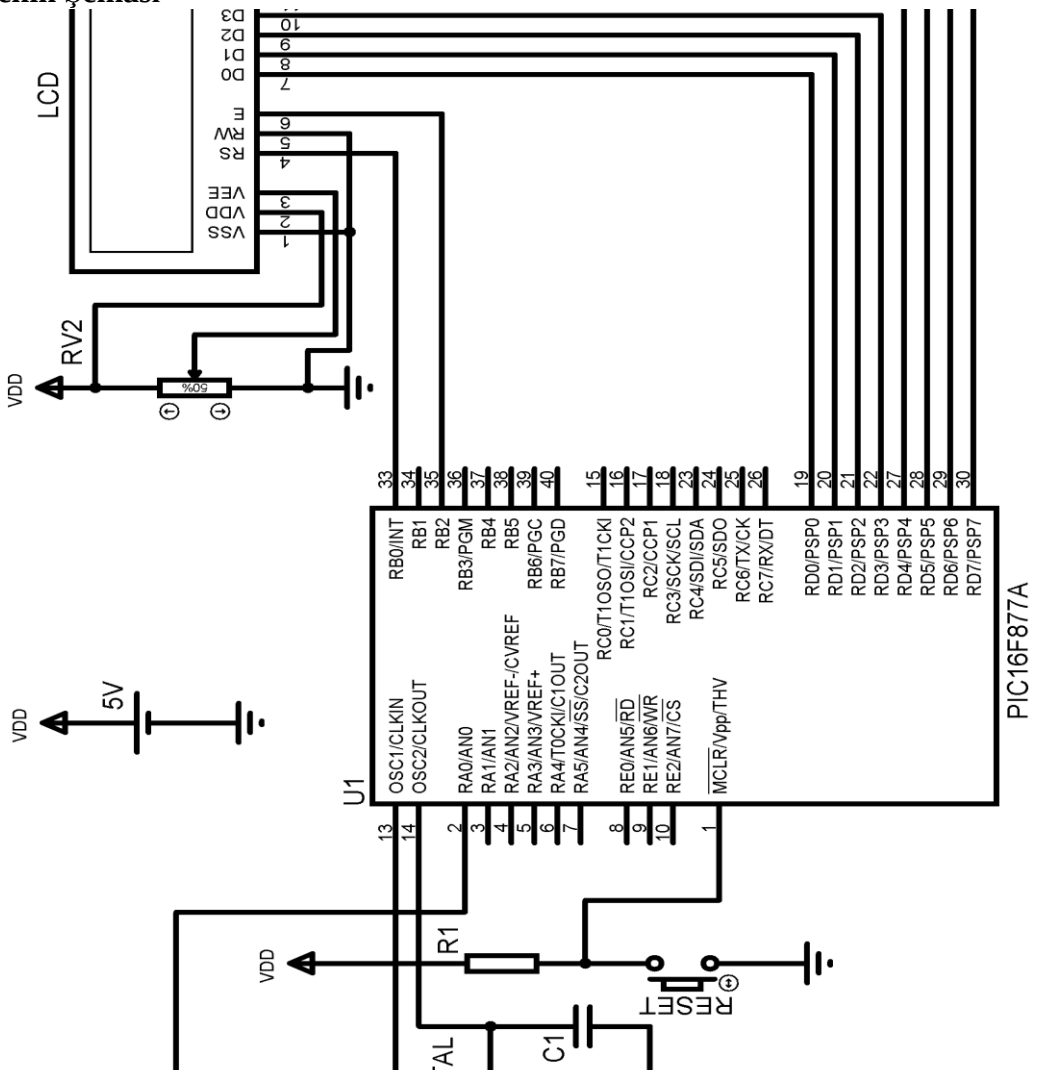
Analog bilgiden dijital bilgiye dönüştürülen bilgi SAYI değişkeninde saklanır. 10 bitlik okuma yapıldığından 5V (Ölçülen maks. gerilimdir.) maksimum değeri dijital olarak 1024 dijital bilgisine karşılık gelir. Örneğin;

2.5V için dijital bilgi 2.5*1024/5 = 512 sayısına denk gelir.

Okunan dijital bilgi direkt olarak LCD'ye gönderilmez. İlk olarak gerilime dönüştürülür. Bunun için dijital bilgi 5/1024 sayısı ile çarpılır. (5 gerilimin maks. değeridir.) PIC Basic'te, bölme işleminden sonra virgülden sonraki sayılar doğrudan atılır. Yani floating point ya da yuvarlama olayı yoktur. Yaptığımız analog-dijital çeviricide virgülden sonraki rakam değerlerini kaybetmemek, hassasiyeti artırmak için okunan dijital bilgi 100 değeri ile çarpılır.

Elde edilen gerilim SAYI değişkeninde saklanır. Bu bilgi LCD'ye direk gönderilemediğinden sayının her dijiti ayrı bir karaktere aktarılır.

4.1.2. Devrenin Şeması



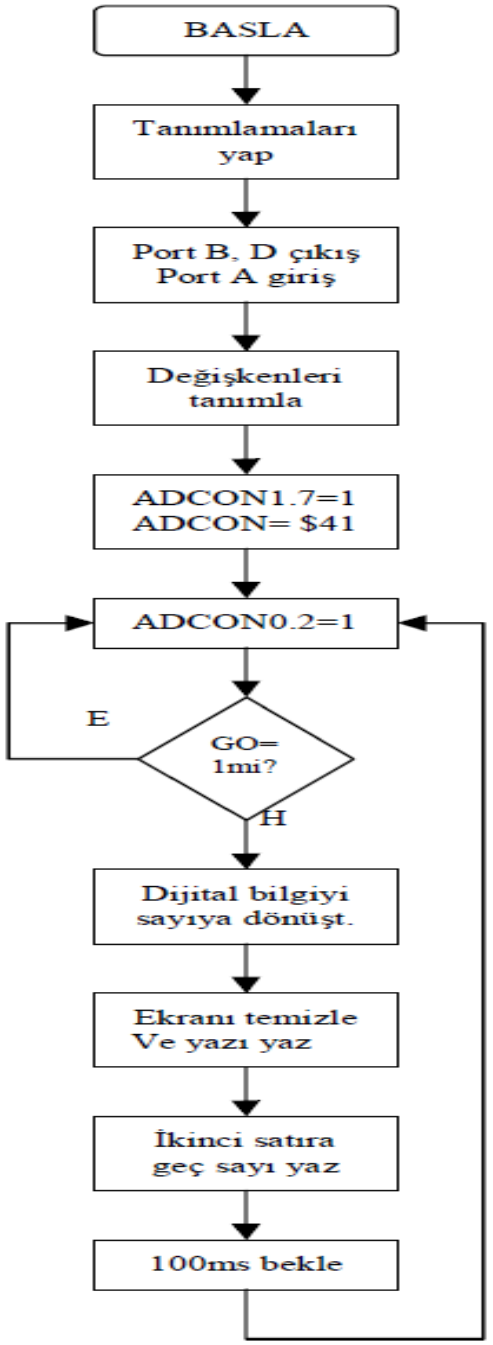
Şekil 4.1: Voltmetre uygulama devresi

4.1.3. Devrenin Asm Programı

```
'================================================================
DEFINE LCD_DREG PORTD                   'LCD data portu = PortD
DEFINE LCD_DBIT 0                       'data bitlerin başlangıcı = 0
DEFINE LCD_RSREG PORTB                  'RS = PortB
DEFINE LCD_RSBIT 0                      'RS = PortB.0
DEFINE LCD_EREG PORTB                   'E = PortB
DEFINE LCD_EBIT 2                       'E = PortB.2
```

```
DEFINE LCD_LINES 2                  '2 satır aktif
DEFINE LCD_BITS 8                   '8 bitlik veri yolu
DEFINE ADC_BITS 10                  'A/D çevrim 10 bit
DEFINE ADC_CLOCK 3                  'RC = 3
DEFINE ADC_SAMPLEUS 50              'Örnekleme sayısı = 50µs

TRISA = %11111111                   'PortA giriş
TRISD = %00000000                   'PortD çıkış
TRISB = %00000000                   'PortB çıkış

SAYI VAR WORD                       'SAYI degiskenini Word olarak tanımla
SOL_K VAR BYTE                      'SOL_K degiskenini tanimla
ORTA_K VAR BYTE                     'ORTA_K değişkenini tanımla
SAG_K VAR BYTE                      'SAG_K değişkenini tanımla
LSB VAR BYTE                        'LSB değişkenini tanımla

        LSB = 5000/1024
        ADCON1.7 = 1                'Sağa kaydırmalı olarak A/D yap
        ADCON0 = $41                'A/D çevrim aktif, Kanal 0 seçili
        PAUSE 500                   '500ms bekle

ADC_OKU:
        ADCON0.2=1                  'GO = 1 yap yani A/D çevrime başla
DON:
        PAUSE 5                     '5msn bekle
        IF ADCON0.2=1 THEN GOTO DON 'A/D çevrim bitmediyse DON'e git
        SAYI.HIGHBYTE = ADRESH      'SAYI'nın yüksek 8 biti = ADRESH
        SAYI.LOWBYTE=ADRESL         'SAYI'nın düşük 8 biti = ADRESL
        SAYI = SAYI*LSB/10          'Okunan dijital bilgi Volt olarak
        LCDOUT $FE, 1               'Ekranı temizle
        SOL_K= SAYI DIG 2           'Sayının 2. dijitini sol karaktere aktar
        ORTA_K= SAYI DIG 1          'Sayının 2. dijitini orta karaktere aktar
        SAG_K= SAYI DIG 0           'Sayının 2. dijitini sağ karaktere aktar
        LCDOUT $FE,2                'Kursörü 1. satırın başına al
        LCDOUT "OLCULEN GERILIM"    'Ekrana tırnak içindekini yaz
        LCDOUT $FE,$C3              'Kursörü 2.satırın 3. sütununa al
        LCDOUT #SOL_K, ".",#ORTA_K, #SAG_K, "V"
'Ekrana sol, orta ve sağ karakteri yaz
        PAUSE 100                   '100 ms bekle
        GOTO ADC_OKU                'Geri dön

        END
```

4.1.4. Akış Diyagramı



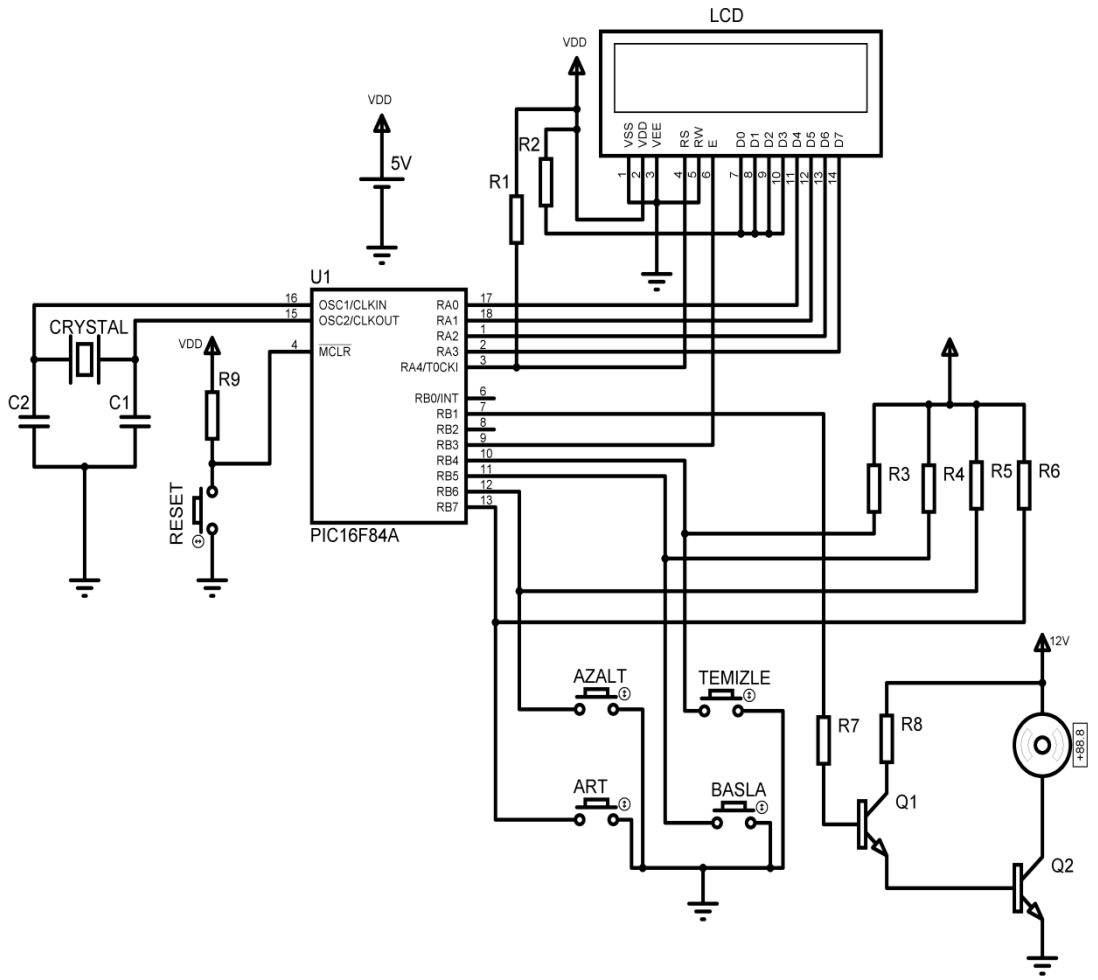
Tablo 4.1: Voltmetre devresi programı akış diyagramı

4.2. DC Motor Devir Ayar Uygulama Devresi

4.2.1. Devrenin Malzemeleri

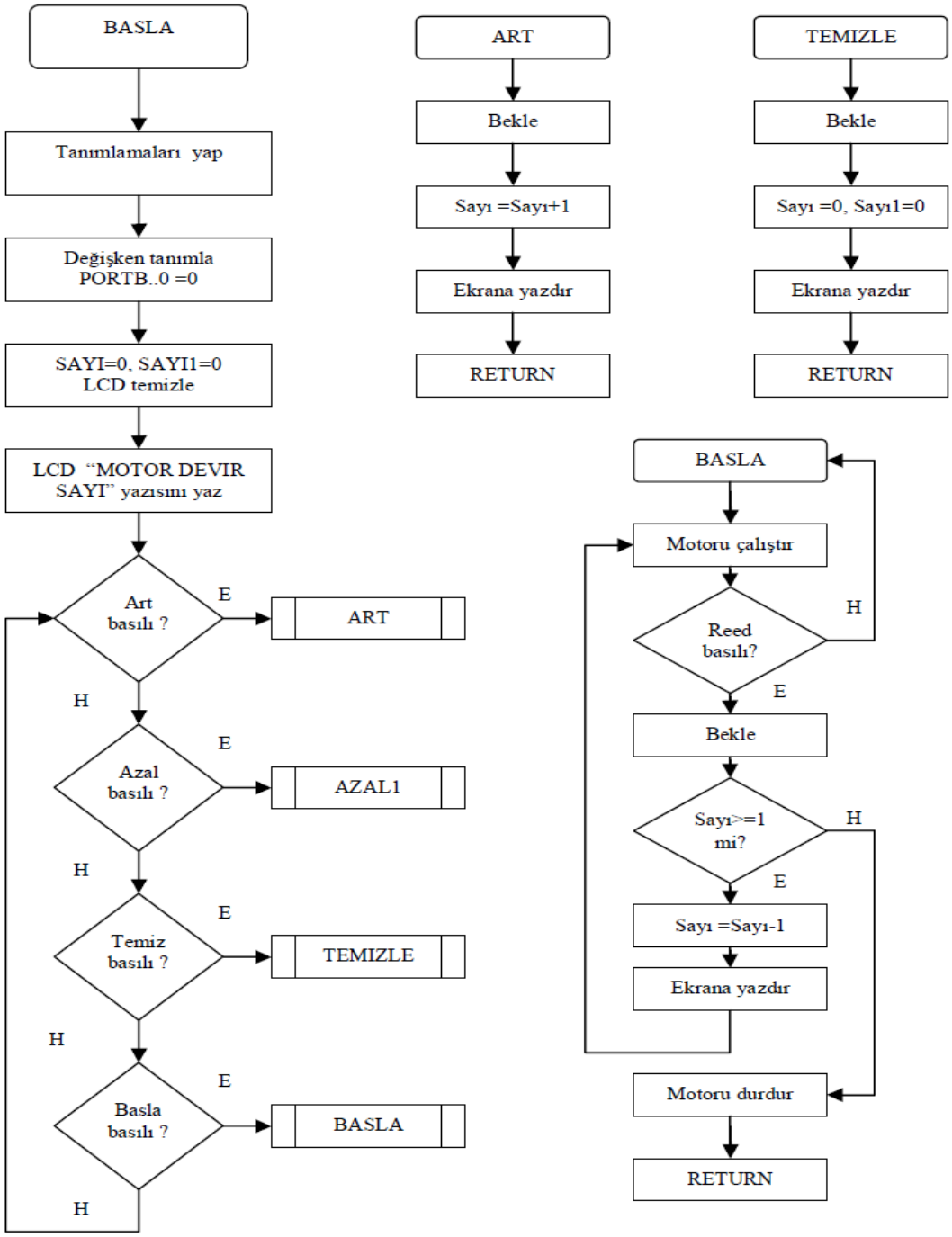
- PIC 16F84A 4 Mhz mikrodenetleyici
- C1 = C2 = 22pf, CRYSTAL= 4Mhz kristal
- R2= R3 = R4 = R5 = R6 = R7 = R8 = 1K
- R1=R9 = 10K, Q1 = BC237, Q2 = BD135, 12V DC Motor
- 2x16 LCD, Buton

4.2.1. Devrenin Şeması



Şekil 4.2: DC motor devir ayar devresi

4.2.3. Akış Diyagramı



Tablo 4.2: DC motor devir ayarlama devresi programı akış diyagramı

4.2.3. Devrenin Asm Programı

```
'=============================================================
DEFINE LCD_DREG PORTA                          'LCD data portu = PortA
DEFINE LCD_DBIT 0                              'data bitlerin başlangıcı = 0
DEFINE LCD_RSREG PORTA                         'RS = PortA
DEFINE LCD_RSBIT 0                             'RS = PortA.0
DEFINE LCD_EREG PORTB                          'E = PortB
DEFINE LCD_EBIT 3                              'E = PortB.3
DEFINE LCD_LINES 2                             '2 satır aktif
DEFINE LCD_BITS 4                              '4 bitlik veri yolu
SAYI VAR BYTE                                  'SAYI değişkenini tanımla
SAYI1 VAR BYTE                                 'SAYI1 değişkenini tanımla

        PAUSE 100               '100 ms'lik gecikme LCD'nin açılmasını bekle
        SAYI=0
        SAYI1=0
        LCDOUT $FE, 1                          'LCD'nin ekranını sil
        LCDOUT "MOTOR SARIM SAYI:",#SAYI1
        LCDOUT $FE, $C0, #SAYI

BAS:
        IF PORTB.7=0 THEN GOSUB ART            'ART butonu basılı mı?
        IF PORTB.6=0 THEN GOSUB AZAL1          'AZAL butonu basılı mı?
        IF PORTB.5=0 THEN GOSUB BASLA          'BASLA butonu basılı mı?
        IF PORTB.4=0 THEN GOSUB TEMIZLE        'TEMIZLE basılı mı?
        GOTO BAS                'Hiçbir butona basılı değil BAS etiketine geri dön

ART: '======================================================
        PAUSE 50                               '50 ms bekle
        SAYI = SAYI+1                          'SAYI değişkenini bir arttır.
        SAYI1 = SAYI                           'SAYI1'e aktar
        LCDOUT $FE,1                           'LCD'ye yazdır
        LCDOUT "MOTOR SARIM SAYI:", #SAYI1
        LCDOUT $FE, $C0, #SAYI
        RETURN

AZAL1: '=====================================================
        PAUSE 50
        IF SAYI<=0 THEN GOTO CIK1              'Sayı 0'a eşitse geri dön
        SAYI= SAYI-1                           'değilse sayıyı azalt
        SAYI1= SAYI
        LCDOUT $FE, 1                          'Ekrana yazdır
```

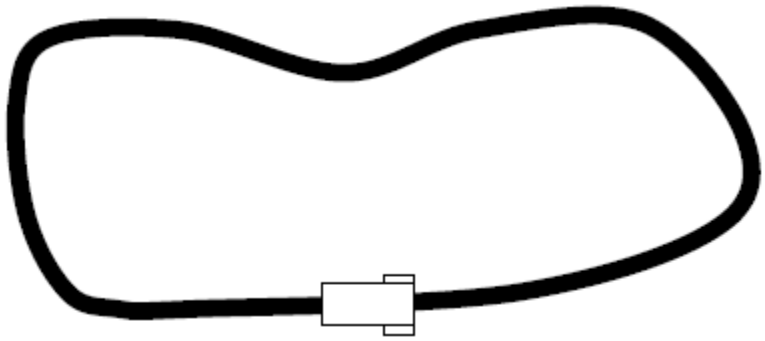
```
        LCDOUT "MOTOR SARIM SAYI:", #SAYI1
        LCDOUT $FE, $C0, #SAYI

CIK1:
        RETURN

TEMIZLE: '=============================================================
        PAUSE 50
        SAYI1= 0                                    'Değişkenleri sıfırla
        SAYI= 0
        LCDOUT $FE, 1                               'Ekrana yazdır
        LCDOUT "MOTOR SARIM SAYI:", #SAYI1
        LCDOUT $FE, $C0, #SAYI
        RETURN

BASLA: '=============================================================
        PORTB.1=1                                   'Motoru çalıştır
        IF PORTB.0=1 THEN GOTO BASLA                'Motor bir tur attı mı?
        PAUSE 50                                    'Evet 50ms bekle
        IF SAYI>=1 THEN GOTO DEVAM                  'Sayı 0 dan büyük mü?
        PORTB.1=0                                   'Hayır motoru durdur
        RETURN                                      'Alt programdan çık

DEVAM:                                              'Evet sayı 0 dan büyük
        SAYI= SAYI-1                                'Sayıyı azalt
        LCDOUT $FE,1                                'Ekranı temizle
        LCDOUT "MOTOR SARIM SAYI:",#SAYI1           'Ekrana yazdır
        LCDOUT $FE,$C0,#SAYI
        GOTO BASLA                                  'BASLA etiketine geri dön

        END
'=============================================================
```

Devrede dört tane buton kullanılmıştır. ART butonu devir sayısını artırmak için kullanılır. Devre ilk çalıştırıldığında LCD'nin ilk satırında "MOTOR DEVIR SAYISI" ve ikinci satırında da devir sayısı bulunur.

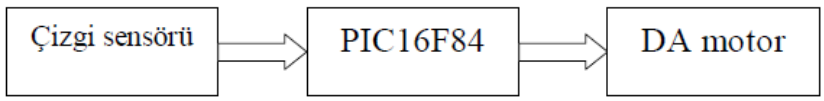
AZALT butonu devir sayısı azaltılır. TEMIZLE butonu ile de devir sayısı resetlenir. Devir sayısı ayarlandıktan sonra BASLA butonuna basıldığında motor çalışmaya başlar. Motorun attığı her tur reed röle ile belirlenir. Motorun her turunda reed röle kapanır ve motorun devir sayısı 1 azalır. Bu LCD ekranında da gözlenebilir. Devir sayısı 0'a indiğinde motor durmaktadır. Uygulama devresinde reed rölenin yerine bir buton kullanılmıştır.

4.3. Çizgi Takip Eden Robot Uygulama Devresi

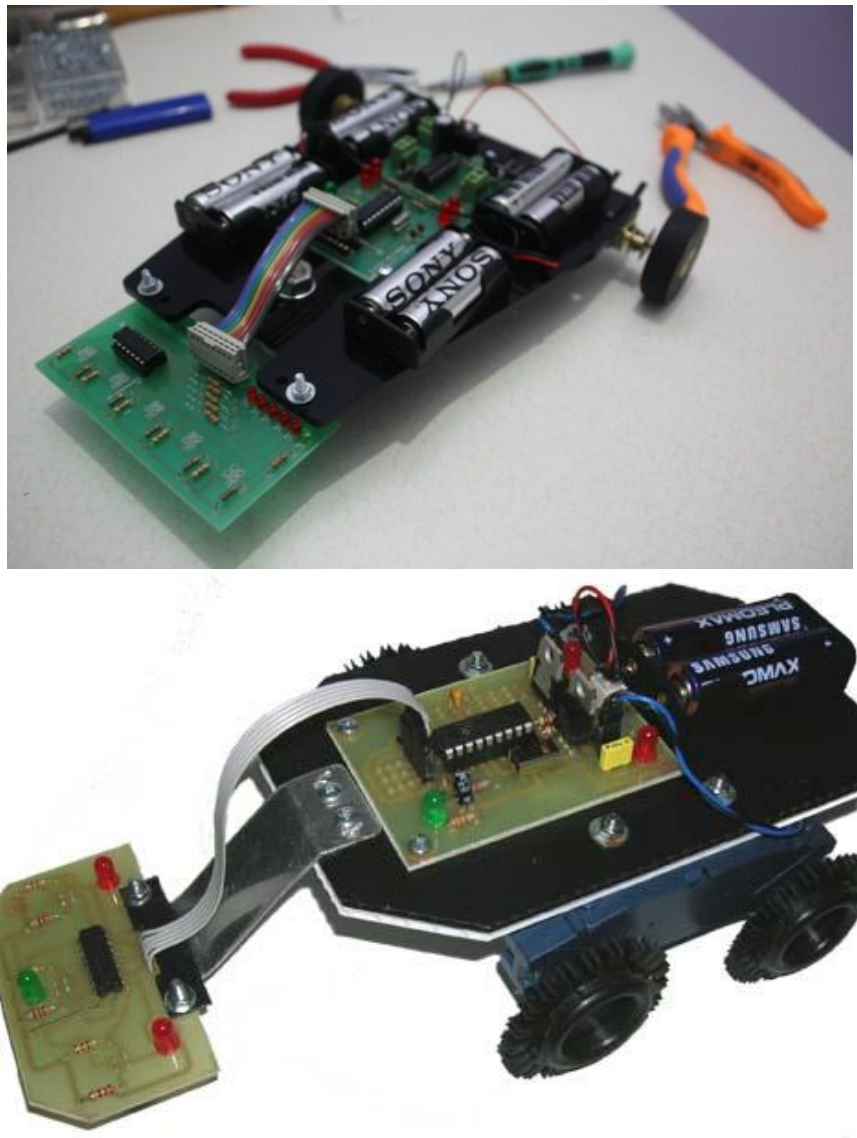
Hızla gelişen günümüz teknolojisi ile beraber birçok alanda insan gücüne olan ihtiyaç azalmıştır. Akıllı cihazların geliştirilmesiyle insan gücünün yerini robotlar almıştır. Robotlar sanayinin tüm alanlarına ve hatta günlük yaşantımıza girmiştir. Çizgi takip eden robot şekil 4.3' te gösterildiği gibi siyah çizgiyi takip etmektedir. Robot sistemi, algılama, karşılaştırma, kontrol ve hareket sisteminden oluşur (Şekil 4.4).



Şekil 4.3: Çizgi takip eden robot hattı



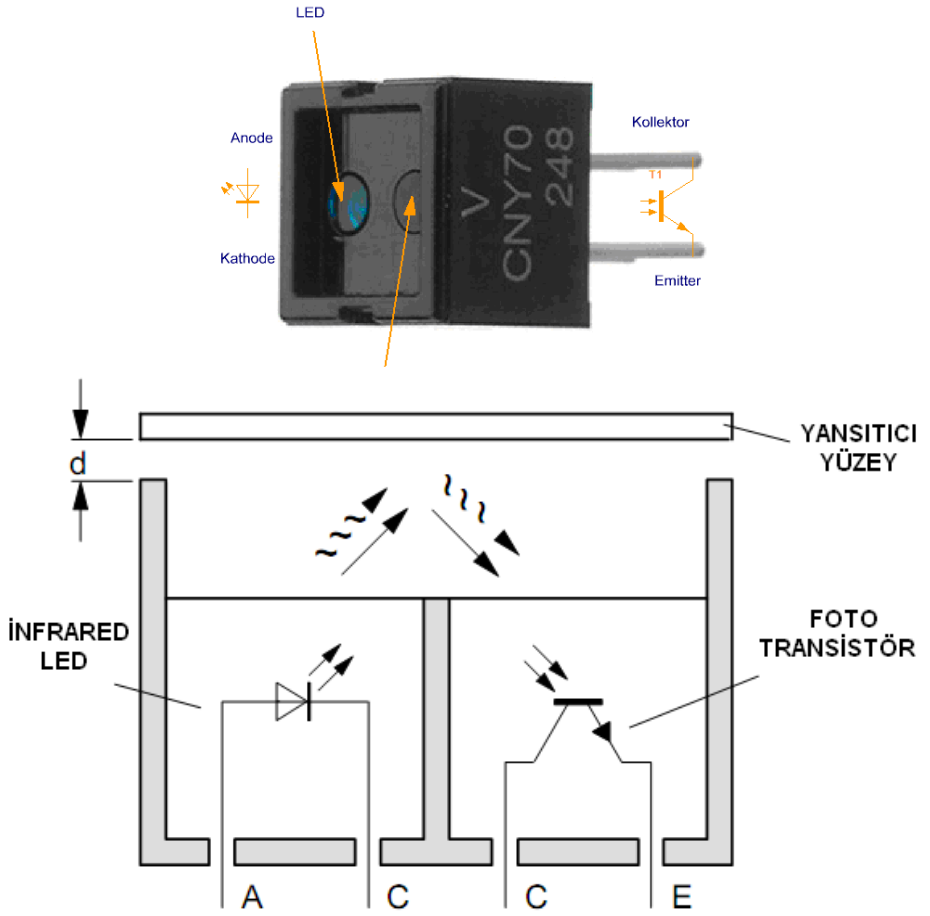
Şekil 4.4: Robot sisteminin blok diyagramı



Resim 4.1: Çizgi izleyen robot resimleri

4.3.1. Algılama Sistemi

Algılama sistemi çizgi sensöründen oluşur. Çizgi sensörü bir kızıl ötesi LED ve bir foto transistörden oluşur. Çizgi sensörü olarak CNY 70 entegresi de kullanılabilir. CNY 70 entegresi Resim 4.2'de gösterildiği gibi bir fototransistör ve kızıl ötesi LED'den oluşur.



Resim 4.2: CNY70 optik sensörü

Algılama sisteminde kızıl ötesi LED, zemine sürekli ışık tutar. Işık yerden fototransistöre yansır.

Eğer ışık beyaz zeminden yansımış ise fototransistör üzerine fazla ışık düşer. Yansıma siyah zeminde gerçekleşirse siyah renk ışık emdiği için daha az ışık yansır.

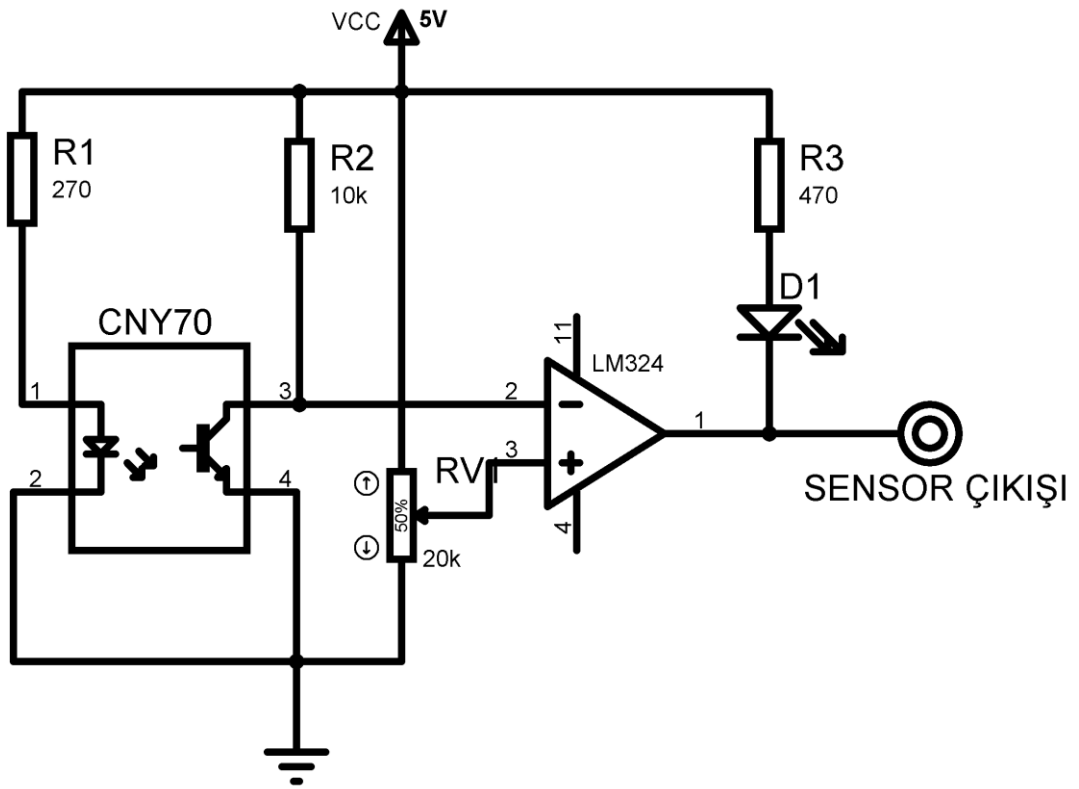
Fototransistörler BJT'lerden farklı olarak beyz girişine sahip değildir. Bu transistörler beyze ışık geldiğinde iletime geçer. Siyah zemindeki zayıf yansımadan dolayı transistör ya iletime geçmez ya da çok küçük sızıntı akımı akıtır. Beyaz zeminde ise transistör güçlü yansımadan dolayı iletime geçer.

4.3.2. Karşılaştırma Sistemi

Karşılaştırıcı olarak devrede OP-AMP kullanılmıştır. Algılama sisteminden elde ettiğimiz beyaz ve siyah renkler arasındaki gerilim farkını PIC mikrodenetleyicinin anlayacağı lojik ifadelere dönüştürmek gerekir. Şekil 4.5'te gösterildiği gibi algılama devresine karşılaştırıcı devresi ilave edilir. OP-AMP olarak LM324 entegresi kullanılmıştır ve OP-AMP'ın beslemesi +5V, 0V olarak yapılmalıdır yani simetrik besleme yapılmaz.

Siyah çizgide, yansıma az olduğundan fototransistör kesimde olur ve OP-AMP'ın (-) girişine uygulanan gerilim (Bu gerilim 5V'a eşittir.), referans geriliminden daha büyük olduğundan çıkış gerilim 0V'tur. PIC'in portuna uygulanan gerilim lojik 0 olur.

Beyaz çizgide, yansıma fazla olduğundan fototransistör iletime geçer ve OP-AMP'ın (-) girişine uygulanan gerilim (Bu gerilim 0V'a eşittir.), referans geriliminden küçük olduğundan çıkış gerilim +5V'tur. PIC'in portuna uygulanan gerilim lojik 1 olur.



Şekil 4.5: Çizgi takip eden robot devresi programı akış diyagramı karşılaştırma devresi

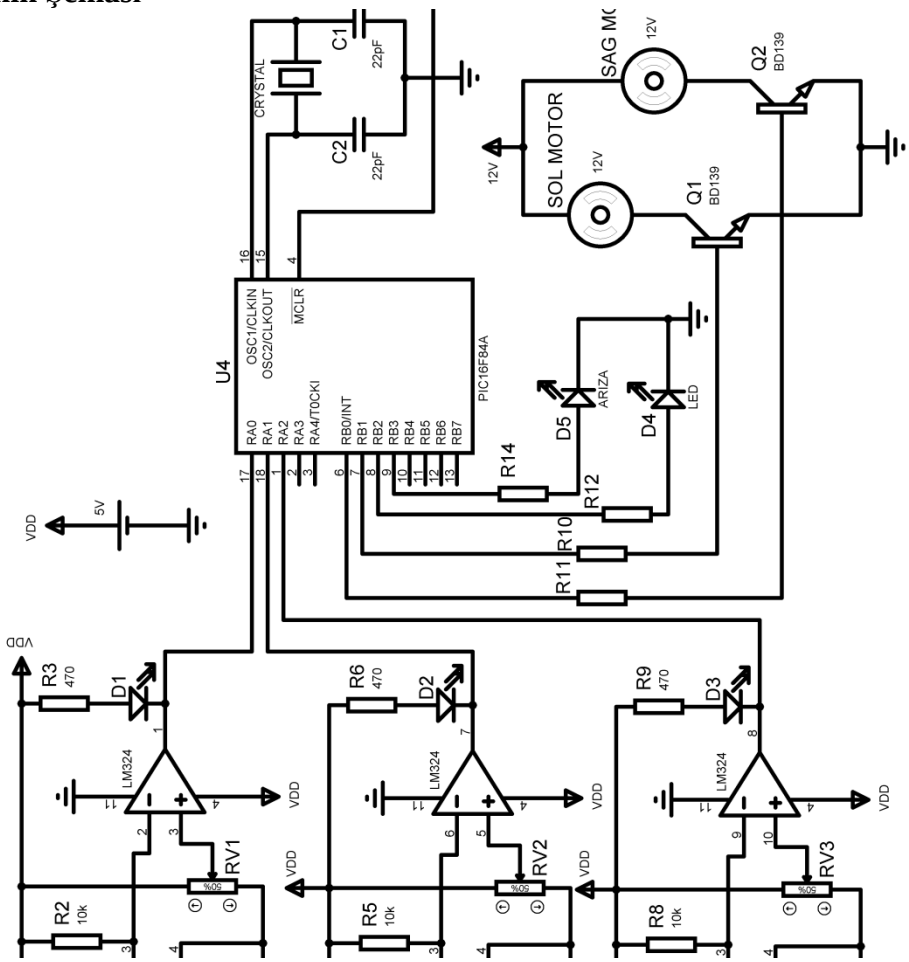
Karar verme sisteminde, PIC 16F84 mikrodenetleyicisi kullanılmıştır. Sensörlerin çizgide olma durumu aşağıda açıklanmıştır. Sistemde üç adet sensör kullanılmıştır.

- Sensör2 siyah çizgi üzerinde, sensör1 ve sensör2 beyaz zeminde ise çizgi robotun tam ortasından geçer. Bu durumda her iki motorda ileri yönde hareket eder.
- Sensör1 siyah çizgi üzerinde ise soldaki motor duracak, sağdaki motor çalışacaktır.
- Sensör3 siyah çizgi üzerinde ise sağdaki motor duracak, soldaki motor çalışacaktır. Hangi sensör siyah çizgi üzerinde ise o yöndeki motor durmalıdır veya yavaşlamalıdır.

4.3.3. Kullanılan Malzemeler

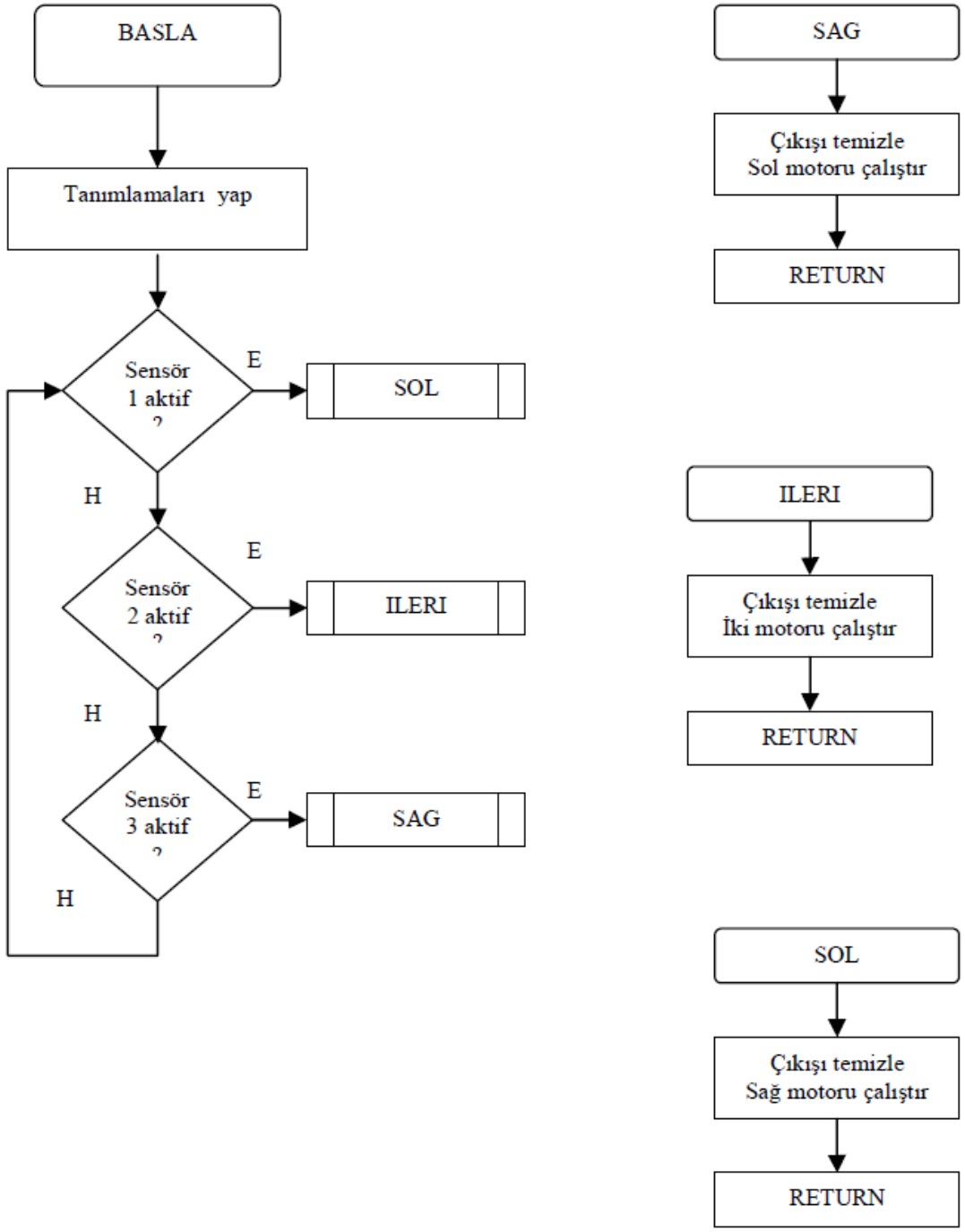
- PIC 16F84A 4 Mhz mikrodenetleyici
- C1 = C2 = 22pf , X1= 4Mhz kristal
- R1= R4 = R7 = 270Ω
- R2= R5= R8= R13= 10K
- R3= R6= R9= R10= R11= R12= 470Ω
- Q1 = Q2= BD139, 12V DC motor
- RV1= 20K potansiyometre

4.3.4. Devrenin Şeması



Şekil 4.6: Çizgi takip eden robot devresi

4.3.5. Akış Diyagramı



Tablo 4.3: Çizgi takip eden robot devresi programı akış diyagramı

4.3.6. Asm Programı

```
            SAY VAR BYTE

;***GIRISLERI TANIMLADIK************************************
            INPUT PORTA.1                   'soldaki cny70
            INPUT PORTA.2                   'ortadaki cny70
            INPUT PORTA.3                   'sağdaki cny70

;***ÇIKISLARI TANIMLADIK************************************
            OUTPUT PORTB.1                  'sol motor ileri
            OUTPUT PORTB.2                  'sol motor geri
            OUTPUT PORTB.3                  'led çıkışı
            OUTPUT PORTB.4                  'ikaz için

;**ISLEM KOLAYLIGI ICIN PORT ISIMLERINI SEMBOLLERE DONUSTURDUK**
            SYMBOL SOLCNY =PORTA.0
            SYMBOL ORTACNY =PORTA.1
            SYMBOL SAGCNY =PORTA.2
            SYMBOL SOLILERI=PORTB.0
            SYMBOL SAGILERI=PORTB.1
            SYMBOL LED=PORTB.2
            SYMBOL IKAZ=PORTB.4

;***ILK HAREKET**********************************************************
            BASLANGIC:
            GOTO DUZGIT

;***ANA DONGU************************************************************
ANA:
            IF ORTACNY=0 THEN SIFIRLA_ORTA 'eğer orta beyazsa yalpalı gıt
            IF SOLCNY=0 THEN SIFIRLA_SOL
            IF SAGCNY=0 THEN SIFIRLA_SAG
ALT:
            IF ORTACNY=1 THEN A1            'eğer orta siyahsa A1'e git
A1:
            IF SOLCNY=1 THEN A2             'eğer sol siyahsa A2'ye git
A2:
            IF SAGCNY=1 THEN A3             'eğer sağ siyahsa A3'e git
GOTO ANA

;***ALT RUTINLER*********************************************************
SIFIRLA_ORTA:                               'eğer orta beyazsa iki motoru calıstır
```

```
            SAY=0
            GOTO YALPALA

SIFIRLA_SOL:                        'eger sol beyazsa sol motoru calıstır
            SAY=0
            GOTO SOLA_DON

SIFIRLA_SAG:                        'eger sağ beyazsa sağ motoru calıstır
            SAY=0
            GOTO SAGA_DON

SOLA_DON:
            IF SOLCNY=SAGCNY THEN DUZGIT
            HIGH SAGILERI           'yalnız sağ motoru çalıstıtırsa
            LOW SOLILERI
            GOTO ANA                'ana program bloğuna geri doner

SAGA_DON:
            IF SAGCNY=SOLCNY THEN DUZGIT
            HIGH SOLILERI           'yalnız sol motoru çalıştırır
            LOW SAGILERI
            GOTO ANA                'ana program bloğuna gerı doner

DUZGIT:
            HIGH SOLILERI           'her ıkı motoru da çalıstıtırsa duz der
            HIGH SAGILERI
            GOTO ANA                'ana program bloğuna gerı doner

A3:
            IF SAY=20 THEN DURDUR   'eger 50 kez arayıp çizgi bulamazsa
            SAY=SAY+1
            HIGH SOLILERI
            HIGH SAGILERI
            GOTO ARAMA

YALPALA:
            IF ORTACNY=SOLCNY THEN SOLA_DON
            IF ORTACNY=SAGCNY THEN SAGA_DON
            HIGH SOLILERI
            LOW SAGILERI
            PAUSE 100
            HIGH SAGILERI
            LOW SOLILERI
```

```
        PAUSE 100
        GOTO ANA

ARAMA:
        HIGH IKAZ
        PAUSE 50
        LOW IKAZ
        PAUSE 50
        GOTO ANA

DURDUR:
        HIGH LED
        PAUSE 50
        LOW LED
        PAUSE 100
        HIGH LED
        PAUSE 50
        LOW LED
        PAUSE 100
        HIGH LED
        PAUSE 50
        LOW LED
        PAUSE 100

        LOW SOLILERI
        LOW SAGILERI
        END
```

UYGULAMA FAALİYETİ

- Voltmetre uygulama devresi,
- DC motor devir ayar uygulama devresi,
- Çizgi takip eden robot uygulama devresi yapınız.

İşlem Basamakları	Öneriler
➢ Kurulacak sistem için ihtiyaçları tespit ediniz.	➢ Kullandığınız devre elemanlarının özelliklerini internetten araştırınız.
➢ İhtiyacı karşılayacak mikrodenetleyiciyi seçiniz.	➢ Mikrodenetleyici olarak PIC 16F877A veya PIC 16F84A kullanınız.
➢ Analog veri için gerekli hesaplamaları yapınız.	➢ Matemetiksel işlemleri dikkatli şekilde yapınız.
➢ Sistemin mikrodenetleyici programın yazınız.	➢ Programı yazdıktan sonra MPLAB ile PROTEUS programlarında deneyiniz.
➢ Programı mikrodenetleyiciye yükleyiniz.	➢ Programı mikrodenetleyiciye yüklerken kullanılan pic programlayıcıya uygun yazılım kullanınız.
➢ Çevre elemanları ve analog veri sağlayan elemanlar ile devreyi kurunuz.	➢ Devrenin montajını yapmadan önce breadboard üzerinde kurarak deneyiniz.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet,** kazanamadıklarınız için **Hayır** kutucuklarına (X) işareti koyarak öğrendiklerinizi kontrol ediniz.

	Değerlendirme Ölçütleri	Evet	Hayır
1.	Kurulacak sistem için ihtiyaçları tespit edebildiniz mi?		
2.	İhtiyacı karşılayacak mikrodenetleyiciyi seçebildiniz mi?		
3.	Analog veri için gerekli hesaplamaları yapabildiniz mi?		
4.	Sistemin mikrodenetleyici programını yazabildiniz mi?		
5.	Programı mikrodenetleyiciye yükleyebildiniz mi?		
6.	Çevre elemanları ve analog veri sağlayan elemanlar ile devreyi kurabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda "Hayır" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "Evet" ise "Ölçme ve Değerlendirme"ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1.( ) PicBasic programlama dilinde ADCIN komutu ile analog girişler kontrol edilir.

2.( ) Voltmetre devresinde potansiyometre üzerinde gerilim ölçülür.

3.( ) Kullanılan LCD' ler üç satırlıktır.

4.( ) Motor deviri ayarlanırken butonlar kullanılır.

5.( ) Çizgi izleyen robot devresinde beyaz çizgi kullanılmıştır.

6.( ) CNY70 ısı sensörüdür.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise "Modül Değerlendirme"ye geçiniz.

MODÜL DEĞERLENDİRME

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

1. Fiziksel büyüklüklere benzeyen sinyallere ne denir?
A) Dijital sinyal
B) Analog sinyal
C) Testere dişi sinyal
D) Kare sinyal

2. Aşağıdakilerden hangisi DAC işlemlerinde kullanılan yöntemlerden değildir?
A)Ağırlık dirençli
B)R-2R merdiven tipi
C)Karşılaştırcı
D)PWM

3. CALL GECIKME komutunun işlevi aşağıdakilerden hangisidir?
A)Gecikme alt programına gider.
B) Gecikme alt programını siler.
C) Gecikme alt programını taşır.
D) Gecikme alt programından geri döner.

4. 10V'luk tepe değerine sahip bir PWM sinyalinin darbe genişliği %50 ise ortlama gerilim nedir?
A)10V
B)5V
C)20V
D)2.5V

5. PIC16F877 entegresinin maksimum osilatör frekansı nedir?
A) 10MHz
B) 4MHz
C) 8MHz
D) 20MHz

6. CLRF PORTB komutunun işlevi aşağıdakilerden hangisidir?
A) PORTB' yi taşı.
B) PORTB' yi birle.
C) PORTB' yi sıfırla.
D) PORTB' yi kapat.

7. PWM sinyalin hangi özelliğini değiştirir?
A) Frekans
B) Dalga genişliği
C) Periyot
D) Genlik

8. LM35 sensörü her sıcaklık değişiminde hangi büyüklüğü değiştirir?
A) Akım
B) Direnç
C) Gerilim
D) Güç

9. PAUSE komutunun işlevi aşağıdakilerden hangisidir?
A) msn cinsinden zaman gecikmesi sağlar.
B) Programı hızlandırır.
C) Programı durdurur.
D) sn cinsinden gecikme sağlar.

10. **10.** Çizgi izleyen robot devresinde kaç adet optik sensör kullanılır?
A)1
B) 2
C) 3
D) 4

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki modüle geçmek için öğretmeninize başvurunuz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ-1'İN CEVAP ANAHTARI

1	Doğru
2	Doğru
3	Yanlış
4	Doğru
5	Doğru

ÖĞRENME FAALİYETİ-2'NİN CEVAP ANAHTARI

1	Doğru
2	Doğru
3	Yanlış
4	Yanlış
5	Doğru
6	Yanlış
7	Doğru

ÖĞRENME FAALİYETİ-3'ÜN CEVAP ANAHTARI

1	Doğru
2	Yanlış
3	Doğru
4	Yanlış
5	Doğru
6	Doğru
7	Yanlış

ÖĞRENME FAALİYETİ-4'ÜN CEVAP ANAHTARI

1	Doğru
2	Doğru
3	Yanlış
4	Doğru
5	Yanlış
6	Yanlış

MODÜL DEĞERLENDİRME CEVAP ANAHTARI

1	**B**
2	**C**
3	**A**
4	**B**
5	**D**
6	**C**
7	**B**
8	**C**
9	**A**
10	**C**

KAYNAKÇA

- MEGEP, Mikrodenetleyici ile Analog İşlemler, Ankara, 2007.