**KARADENİZ TEKNİK ÜNİVERSİTESİ**
**FEN BİLİMLERİ ENSTİTÜSÜ**

**ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI**

# OTONOM ARAÇ İÇİN DA MOTOR SÜRÜCÜ SİSTEMİ VE DENETLEYİCİNİN GERÇEKLENMESİ

**YÜKSEK LİSANS TEZİ**

**Elektrik-Elektronik Müh. İbrahim YAŞAR**

**KASIM 2009**
**TRABZON**

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

# OTONOM ARAÇ İÇİN DA MOTOR SÜRÜCÜ SİSTEMİ VE DENETLEYİCİNİN GERÇEKLENMESİ

**Elektrik Elektronik Müh. İbrahim YAŞAR**

**Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünce
"Elektrik Yüksek Mühendisi"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.**

**Tezin Enstitüye Verildiği Tarih : 12.06.2009**
**Tezin Savunma Tarihi : 20.11.2009**

**Tez Danışmanı : Yrd. Doç. Dr. H. İbrahim OKUMUŞ**
**Jüri Üyesi : Prof. Dr. A. Sefa AKPINAR**
**Jüri Üyesi : Yrd. Doç. Dr. Hüseyin PEHLİVAN**

**Enstitü Müdürü : Prof. Dr. Salih TERZİOĞLU**

**Trabzon 2009**

# ÖNSÖZ

Bu tez, Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı, Elektrik Mühendisliği Yüksek Lisans Programı'nda hazırlanmıştır. Tezimde insansız araçlar üzerinde çalışılmıştır. Sesüstü ve kızılötesi algılama sistemleriyle çalışan akıllı bir sistem oluşturulmaya çalışılmıştır. Tez çalışmam süresince bilimsel desteği ve değerli düşünceleriyle bana her aşamada yardımcı olan danışmanım Doç. Dr. Halil İbrahim OKUMUŞ' a, çalışmamda gayret veren, bildiklerini, zamanını benden esirgemeyen değerli hocam Arş. Gör. Emre ÖZKOP'a, yardımlarını benden esirgemeyen çok sevdiğim arkadaşlarım Hakan KAHVECİ ve Mehmet EKİCİ'ye, bilgilerini benden esirgemeyen Ali YALÇIN beye kaynak yapmayı bana öğreten kaynak ustası Murat'a ve Amcam Cevat YAŞAR'a teşekkürü bir borç bilirim.

Tüm eğitim-öğretim hayatım boyunca maddi ve manevi desteklerini esirgemeyen, ellerinden gelen her türlü imkanı sağlayan, hep sevdiğim ve seveceğim aileme ve çalışmam boyunca sabırla, tam destek vererek beni gayrete getiren, kendisine ayırmam gereken vakitlerden feragat eden, eşim Esin'e teşekkür ederim. Bu tezin, bundan sonraki çalışmalara katkı sağlamasını temenni ederim.

İbrahim YAŞAR
Trabzon 2009

# İÇİNDEKİLER

**Sayfa No**

# ÖZET

Otomasyon teknolojisindeki gelişmeler insana olan ihtiyacı azaltmaktadır. Teknolojik makinelerin satın alma maliyetleri oldukça yüksek olmasına rağmen işletme ve bakım maliyetleri düşüktür. Bu tür makinalar sayesinde insandan kaynaklanan hatalar ortadan kalkmaktadır. Bu çalışmada, verilen rotayı takip edecek bir araç için DA motor sürücü sistemi geliştirilmiştir. Bu araç kullanıcı tarafından görev verildikten sonra aynen bir insan gibi harekete başlayarak, yolda önüne çıkan engellere çarpmadan rotasını ve davranışını belirlemektedir.

Araçta sürekli mıknatıslı bir DA motoru kullanılmaktadır. DA motorunun dört bölge denetimi mikrodenetleyici tabanlı olarak yapılmaktadır. Mikro denetleyici tarafından üretilen DGM (PWM) sinyali ile sürülen H-köprü da-da konvertörü, motora uygulanan gerilimin ortalama değeri değiştirmekte ve böylece motorun hızı denetlenmektedir.

**Anahtar Kelimeler:** Otonom araç, Sabit mıknatıslı d.a. motoru, PIC, Mikrokontrolör

## SUMMARY

### Implementation Of DC Motor Drive System And Controller For Autonomous Vehicle

Advances in automation technology reduce the need for human. Although technological machines have relatively high costs but their operation and maintenance cost is low. The errors caused by human are eliminate using this type of machines. In this study, a motor drive system is developed for the vehicle follows the given route. When the command is given the vehicle moves and acts like a human without hitting obstacles on the road.

In the vehicle a permanent magnet dc motor is used. Four quadrant control of the dc motor is performed based on microcontroller. The average voltage value of the H-bridge converter is changed by PWM signal generated by micro-controller so that the motor speed is controlled.



**Key Words:** Autonomous vehicle, Permanent magnet d.c. machine, PIC, Microcontroller

# ŞEKİLLER DİZİNİ

## TABLOLAR DİZİNİ

**Sayfa No**

# 1. GENEL BİLGİLER

## 1.1. PIC Mikro Denetleyici ve Programlama Dilleri

### 1.1.1 Mikro işlemciler

Mikro işlemci herhangi bir sistemde merkezi işlem birimidir ve bulunduğu sistemde aritmetik ve mantıksal işlemleri yürütür. Merkezi İşlem Birimi (Central Processing Unit: CPU), kontrol devresini, bir ALU ( Aritmetik Mantık Birimi ) bazı kaydediciler ve bir adres/program sayıcıyı içerir.

Bir klavyeden gelen verilerin, bir basınç algılayıcısından gelen sayısallaştırılmış çıkışın veya bir başka verinin bilgisayara alınması ve bu bilgilerin çıkış olarak sağlanması mikro işlemci tarafından kontrol edilir. Mikro işlemcinin bu tür işlemleri giriş olarak algılayıp, çıkışa yansıtması için mikro işlemci programlanır. Bir mikro işlemci, belleğinde saklı bulunan programı her bir komutu sıra ile okuyarak yürütür. Her komut önce, onu yürütmek için gerekecek işlemleri belirlemek üzere mikro işlemcinin anlayabileceği makine koduna çevrilir ve sonra gereken işlemler yapılır. Mikro işlemci entegre devresi, yazılan programları meydana getiren makina komutlarını yorumlamak ve yerine getirmek için gerekli olan tüm mantıksal devreleri içerir.

Bir mikro işlemci temel olarak üç kısımdan oluşur. Bunlar merkezi işlem birimi (CPU), giriş çıkış birimi (G/Ç) ve bellektir.

#### 1.1.1.1 Merkezi İşlem Birimi

Bilgisayarın kalbi veya beyni olarak adlandırılan mikro işlemci aynı zamanda merkezi işlem birimi olarak da anılırlar. Merkezi işlem birimi genel olarak aşağıdaki işlemleri yapar:

- Sistemdeki bütün elemanlar ve birimlere zamanlama ve kontrol sinyali sağlar.
- Bellekten komut veya veri alıp getirir ve işler,
- Veriyi giriş/çıkış birimlerine ya da oradan kendisine aktarır,
- Komutların kodunu çözer,
- Komutla birlikte aritmetik ve mantık işlemlerini gerçekleştirir,

• Program işlenirken giriş/çıkış cihazlarından gelen servis isteklerine I bayrağının durumuna göre cevap verirler.

Intel ve Motorola firmalarının üretmiş olduğu mikro işlemcilerde yukarıda bahsedilen işlemleri gerçekleştirmek üzere üç ana bölüm vardır. Bunlar; Kaydediciler, Aritmetik Mantık Birimi (ALU) ve Zamanlama-Kontrol Birimidir.

#### 1.1.1.1.1 Kaydediciler

Mikro işlemcinin içinde işlem yaparken geçici olarak verilerin saklandıkları saklayıcılardır. Mikro işlemci içerisinde değişik işlemleri gerçekleştirmek üzere akümülatör, indis kaydedicileri, stack pointer, program counter gibi kaydediciler mevcuttur. Mikro işlemciler içerisinde işlem yapılırken bu kaydedicilere veri atılabilir, toplama, karşılaştırma ve kaydırma gibi bazı işlemle gerçekleştirilebilir.

#### 1.1.1.1.2 Aritmetik Mantık Birimi (ALU)

ALU, yürütülmekte olan komutta belirtilen iki değer üzerinde aritmetik ve lojik işlemleri yerine getirir. Bu iki değerden biri daima özel bir mikro işlemci kaydedicisi olan ve A kaydedicisi olarak adlandırılan kaydedicinin o anki içeriğidir. Diğer ALU girişi ise; mikro işlemcinin bölümünü oluşturan genel amaçlı kaydedici grubundan bir başka kaydedicidir.

#### 1.1.1.1.3 Zamanlama-Kontrol Birimi

Mikro işlemcinin kontrol birimi, mikro işlemcinin içinde ve dışında olan bütün veri aktarımlarını ve ALU işlemlerini kontrol eder ve çevre birimlerle eş zamanlama için gerekli sinyallemeyi sağlar. Ayrıca, kontrol birimi ALU'da en son yapılan aritmetik ve mantıksal işlemin sonucunu yansıtan bayrakları giriş olarak alır.

#### 1.1.1.2 Giriş-Çıkış Birimi (G/Ç)

Mikro işlemcinin dış dünya ile ilişkisinin sağlandığı ünitedir. Mikro işlemciye verilen bilgiler bu ünite yolu ile işlemci içerisine alınırken, CPU ve diğer birimlerde işlenen veriler yine bu ünite sayesinde dış ortama aktarılırlar.

#### 1.1.1.3 Bellek

Bellek bir mikro bilgisayar sistemi içerisinde bilgileri saklar. Bellek iki bölüme ayrılmıştır. Bir bölümü programı saklar, diğer bölüm ise mikro bilgisayara gerek duyduğu bilgi ya da veriyi tutmak için kullanılır.

Genel amaçlı bir mikro bilgisayar sisteminde program sıklıkla değiştirilir. Çamaşır makinası denetleyicisi gibi bir mikro işlemci tabanlı sistemde program, fabrikada üretilirken yüklenmiştir ve asla değişmez. Bu tür bir uygulama için program ROM adı verilen özel bir bellek türünde saklanır.

ROM (Sadece okunur bellek): Program deyimleri, veri vb. sadece ROM bellekten kopya edilebilir. Yapımcı veya kullanıcı tarafından ROM belleğe kaydedilen hiçbir bilgi değiştirilemez ve bu tür belleklere yeni kayıt yapılamaz. ROM belleklerden başka bir de RAM bellekler de vardır.

RAM (Rastgele erişimi bellek): Kural olarak RAM' lar, üzerinde yazma ve okuma işlemleri yapılabilen bellekler olarak tanımlanmıştır. Bir başka deyişle RAM, değişken veri saklayabilen bir rastgele erişimli bellektir. RAM, bir elektronik devre olduğu için, güce gereksinim duyar. Güç kaynağı kesildiği anda çok anlık olsa bile saklanan bir veri yitirilir. Sistem anahtarı kapatıldığı anda, belleğin içeriği yitirilmiş olur. Bu nedenle bu belleğe kalıcı olmayan saklama ortamı adı verilir.

Başka bellek türleri de vardır. Örneğin, EPROM, silinebilir programlanabilir ROM' dur. EPROM, programcıya bir yonga satın alıp bunu programlama imkanı verir.

### 1.1.2 Mikrodenetleyiciler

Bir yazılım olmadan hiçbir işe yaramayan, ancak içerisine yazılan program vasıtasıyla istenilen bir işlemi gerçekleştiren kontrol elemanıdır. Mikrodenetleyici yazılım olması halinde neredeyse sınırsız bir kullanım alanına sahiptir.

Aslında mikrodenetleyici bir bakıma bir bilgisayardır. Her nekadar bir klavyesi, monitörü, kasası ve bunun gibi çevre birimleri olmasa da bir bilgisayarın yaptığı her şeyi yapabilir. Örneğin her bilgisayarın bir merkezi işlem ünitesi (CPU:Central Processing Unit) vardır ve bu ünite makine kodlarını bizim anlayabileceğimiz karakterlere dönüştürür, programları yorumlar, işler, düzenler, bilgisayarın çeşitli birimleriyle irtibat kurar. Ve bu işlemleri yaparken bazı değişkenleri ve geçici olarak elde ettiği bilgileri sakladığı bir rastgele erişimli hafızaya (RAM:Random Access Memory) ihtiyaç duyar. Ayrıca bilgisayarların dış dünyayla bilgi alış ve verişlerinde kullandıkları bazı giriş ve çıkış üniteleri bulunmaktadır. Örnek olarak fare ve klavye giriş yaptığımız elemanlara, monitör ve yazıcı çıkış aldığımız elemanlara birer örnektir. Bilgilerimizi kaydettiğimiz harddiskler ise hem giriş hem de çıkış elemanı olarak çalışmaktadırlar. Aynen bilgisayarda olduğu gibi mikrodenetleyiciye de fare ve klavye gibi çevre elemanlarının işlemlerini nispeten de olsa yerine getirecek elemanlar ekleyerek küçük bir bilgisayar gibi kullanmamız mümkündür.

Bir mikrodenetleyici genel olarak aşağıdaki birimlerden oluşur:

- CPU (Merkezi işlem ünitesi - central processing unit)
- RAM (Rastgele erişimli bellek-Random Access Memory)
- EPROM/PROM/ROM (Silinir, yazılır sadece okunur bellek-Erasable Programmable Read Only Memory)
- I/O (Girdi/çıktı - input/output) - seri ve paralel
- Timers (Zamanlayıcılar)
- Interrupt controller (Kesmeler)

Sadece kullanılacak işe uygun özellikleri bulunan bir mikrodenetleyici seçildiğinde maliyet nispeten düşmektedir.

Mikrodenetleyicilerde işlemler ve komutlar bit bit kontrol edilebildiğinden giriş ve çıkış birimleri ve kesmeler çok etkin bir şekilde kulllanılabilmektedir.

Şu an kullandığımız masaüstü veya dizüstü bilgisayarlar genel amaçlı bilgisayarlardır ve binlerce programı çalıştırıbilirler. Mikrodenetleyiciler ise özel amaçlı bilgisayarlardır ve programlandıkları şeyi en iyi şekilde yaparlar. Bunun dışında;

• Mikrodenetleyiciler sadece bir iş için programlanmışlardır ve bu program içlerindeki ROM'da değişmemek üzere saklı bulunur.
• Mikrodenetleyiciler düşük güçte çalışan çiplerdir. Bir bilgisayar 50W civarı güç harcarken mikrodenetleyiciler sadece 50 miliWatt civarında güç harcarlar.
• Mikrodenetleyicelere sadece girdi yapılmaz aynı zamanda çıktı da alınabilir. LED göstergelerle, sıvı kristal göstergelerle, ikaz sesleriyle vb.. Örneğin televizyonunuzda ve uzaktan kumandasında bulunan mikrodenetleyicilerden bahsedelim. Kumandada bulunan mikrodenetleyici tuşlara bastığımızda girdisini almış olur ve bunu televizyondaki alıcı mikrodenetleyiciye gönderir. Ve o mikrodenetleyici de gelen sinyale göre çıkış vererek ya kanal seçer, ya ses ayarı yapar ya da televizyonla ilgili işlemleri yapar.
• Mikrodenetleyiceler genelde küçük ve düşük fiyatlı çiplerdir. Bir çok parçadan oluşan kompleks birdevreyi kolayca küçük boyutlara ve maliyete indirmenizi sağlar.

Ayrıca mikrodenetleyiceler belleklerine göre de çeşit gösterirler ;

#### 1.1.2.1 EEPROM Bellekli Mikrodenetleyiciler

Elektriksel olarak silinebilen ve yazılabilen belleklerdir.Çoğu mikrodenetleyicilerde sınırlı sayıda bulunan EEPROM lar, bir defadan fazla yazılıp silinebildikleri için oldukça kullanışlıdır.

#### 1.1.2.2 Flash (EPROM) Bellekli Mikrodenetleyiciler

Flash bellekler EEPROM lardan daha hızlı ve daha çok yazma silme işlemine izin vermeleri yönünden üstündürler. Flash belleklerde bilgilerin korunması söz konusu değildir.

#### 1.1.2.3 OTP Bellekli Mikrodenetleyiciler

Bir kez programlanabilen mikrodenetleyicilerdir. OTP bir kez programlanabilen bir ROM'dur. Programınızı bir EPROM programlayıcı ile bir kez yazdıktan sonra silemez veya değiştiremezsiniz. Bu yöntem programınız artık tamamen hazır olduğunda ve bütün hatalarından arındırıldıktan sonra kullanılır.

Mikrodalga fırınlar gibi birçok üründe kullanılan ve maliyeti de oldukça düşük olan bu çiplerin içine düşük güçlü ve düşük maliyetli CPU lar yerleştirildi. Motorola 6811 ve Intel 8051 ailesi mikrodenetleyicileri bu türe iyi birer örnektir. Bunların yanında Microchip firması tarafından üretilen PIC mikrodenetleyicileri de son zamanlarda oldukça popülerdir. Günümüz standartlarında bu çipler inanılmayacak kadar küçük ve düşük fiyatla satın alınabilmektedir. Sıradan bir mikrodenetleyici içinde en azından 1,000 byte kapasiteye sahip bir ROM ve 20 byte kapasiteye sahip bir RAM, 8 adet I/0 pini bulundurabilir.

### 1.1.3 Mikrodenetleyicinin Mikro işlemciye Olan Üstünlükleri

• Mikro işlemcinin kullanımı ve mikro işlemcili sistemin tasarımı mikrodenetleyicili sisteme göre hem daha masraflı hem de daha karmaşıktır.
• Mikrodenetleyicili bir sistemin çalışması için elemanın kendisi ve bir osilasyon kaynağının olması yeterlidir.
• Mikrodenetleyicinin ihtiyaç duyduğu önbellek ve giriş çıkış birimi bir yonga içerisinde bulunmaktadır. Ancak mikro işlemcili bir sistemde önbellek harici olarak bulunur.

## 1.2 PIC Mikrodenetleyici Ailesine Genel Bakış

PIC serisi mikrodenetleyiciler MICROCHIP firması tarafından geliştirilmiştir. Üretim amacı; çok fonksiyonlu mantık uygulamalarının hızlı ve ucuz bir mikrodenetleyici ile yazılım yoluyla karşılanmasıdır.

PIC'in kelime anlamı PERIPHERAL INTERFACE CONTROLLER (Çevresel arabirim denetleyicisi) dir. İlk olarak 1994 yılında 16 bitlik ve 32 bitlik büyük işlemcilerin giriş ve çıkışlarındaki yükü azaltmak ve denetlemek amacıyla çok hızlı ve ucuz bir çözüme ihtiyaç duyulduğu için geliştirilmiştir.

Çok geniş bir ürün ailesinin ilk üyesi olan PIC16C54 bu ihtiyacın ilk ürünüdür. PIC denetleyicileri RISC benzeri işlemciler olarak anılır. PIC16C54 12 Bit komut hafıza genişliği olan 8 bitlik CMOS bir işlemcidir. 18 bacaklı dip kılıfta 13 I/O bacağına sahiptir ve 20 Mhz osilator hızına kadar kullanılabilir. 33 adet komut içermektedir. 512 byte program EPROM'u ve 25 byte RAM`i bulunmaktadır. Bu hafıza kapasitesi CISC işlemciler için düşük gibi görünebilir ancak PIC'in RISC denetleyici olması birçok işin bu kapasitede uygulanmasına olanak vermektedir.

PIC serisi tüm denetleyiciler herhangi bir ek bellek veya giriş/çıkış elemanı gerektirmeden sadece 2 adet kondansatör, 1 adet direnç ve bir kristal ile çalıştırılabilmektedir. Tek bacaktan 40 mA akım çekilebilme ve entegre toplamı olarak 150 mA akım akıtma kapasitesine sahiptir. Entegrenin 4 Mhz osilator frekansında çektiği akım; çalışırken 2 mA, stand-by durumunda ise 20uA kadardır.

PIC 16C54 'un mensup olduğu denetleyici ailesi 12Bit core 16C5X olarak anılır. Bu gruba temel grup adı verilir. Bu ailenin üyesi diğer denetleyiciler PIC16C57, PIC16C58 ve dünyanın en küçük işlemcisi olarak anılan 8 bacaklı PIC12C508 ve PIC 12C509'dur.

Interrupt (Kesme) kapasitesi ilk denetleyici ailesi olan 12Bit Core 16C5X ailesinde bulunmamaktadır. Daha sonra üretilen ve orta sınıf olarak tanınan 14Bit Core- 16CXX ailesi birçok açıdan daha yetenekli bir grup işlemcidir.

Bu ailenin temel özelliği interrupt kapasitesi ve 14 bitlik komut işleme hafızasıdır. Bu özellikler PIC'i gerçek bir denetleyici olmaya ve karmaşık işlemlerde kullanılmaya yatkın hale getirmiştir. PIC16CXX ailesi en geniş ürün yelpazesine sahip ailedir. 16CXX ailesinin en önemli özellikleri seri olarak devre üstünde dahi programlanmasıdır.

PIC 16CXX ailesinin amatör elektronikçiler arasında en çok tanınan ve dünyada üzerinde ençok proje üretilmiş elemanı ise PIC16C84 veya PIC16F84 tur.

PIC 16F84 un bu kadar popüler olması onun çok iyi bir denetleyici olmasından ziyade program belleğinin EEPROM (Elektrikle silinip yazılabilen bellek) olmasından kaynaklanmaktadır. Seri olarak dört adet kabloyla programlanması da diğer önemli avantajıdır. Bugüne kadar amatörce bir işlemciyle uğraşmış herkesin en büyük sıkıntısı EPROM veya EPROM tabanlı denetleyicileri programladıktan sonra UltraViole ışık kaynağı ile silip tekrar programlamaktır. Bu çok zahmetli ve bir amatör için ekipman gerektiren yöntem olmuştur.

Tasarlanan PIC kontrollü çift kanallı dijital termometrede sistemi kontrol etmek için kullanılan PIC 16F877 ise PIC 16F84'ün tüm özelliklerini taşımaktadır.Ayrıca PIC 16F877 'de PIC16F84 'e ek olarak bazı özellikleri yer almaktadır.

### 1.2.1 PIC Çeşitleri

Microchip ürettiği mikrodenetleyicileri 4 gruba ayırarak isimlendirmiştir. Her bir grubu ise bir PIC ailesi olarak adlandırmıştır. PIC ailelerine isim verilirken kelime boyu (word lenght) göz önüne alınmıştır. Bu kısımda kelime boyunun ne anlama geldiğini açıklamakta fayda vardır. Mikroişemciler (CPU) veya mikrodenetleyiciler (MCU) kendi içlerindeki dahili veri saklama alanları olan kayededicileri arasındaki veri alışverişini farklı sayıdaki bitlerle yaparlar. Örneğin 8088 mikro işlemcisi çip içerisindeki veri alışverişini 16 bit ile yaparken, pentium işlemcileri 32 bitlik verilerle iletişim kurarlar. Bir CPU veya MCU'nun dahili veri yolu uzunluğuna *kelime boyu* denir.

Microchip PIC'leri 12/14/16 bitlik kelime boylarında üretilmektedir ve buna göre aşağıdaki aile isimleri mevcuttur.

• PIC 16C5XX ailesi 12 bit kelime boyu
• PIC 16CXXX ailesi 14 bit kelime boyu
• PIC 17CXXX ailesi 16 bit kelime boyu
• PIC 12CXXX ailesi 12 bit/14 bit kelime boyu

Bir MCU çip dışındaki harici ünitelerle veri alışverişini kaç bitle yapıyorsa buna *veri yolu* bit sayısı denir. PIC'ler farklı kelime boylarında üretilmelerine rağmen harici veri yolu

tüm PIC ailelerinde 8 bittir. Yani bir PIC, G/Ç portu aracılığı ile çevresel ünitelerle veri alışverişi yaparken 8 bitlik veri yolu kullanır.

PIC programcıları program kodlarını yazarken bir komutun kaç bitlik kelime boyundan oluştuğu ile pek fazla ilgilenmezler. Seçilen bir çipi programlarken uyulması gereken kuralları ve o çiple ilgili özelliklerin bilinmesi yeterlidir. Bu özellikler PIC'in bellek miktarı, G/Ç portu sayısı, A/D dönüştürücüye sahip olup olmadığı, kesme (interrupt) fonksiyonlarının bulunup bulunmadığı, bellek tipinin ne olduğu (Flash, EPROM, EEPROM vb) gibi bilgilerdir.

### 1.2.2 PIC Mikrodenetleyicilerinin Tercih Sebepleri

a-) Fiyatının ucuz olması;
b-) Mantıksal işlemlerde performansının yüksek olması;
c-) Verilere ve belleğe hızlı bir şekilde erişimin sağlanması;
d-) 8 bitlik bir mikrodenetleyici olması ;
e-) Veri ve bellek için yolların (bus'ların) ayrılmış olması;
f-) Yüksek frekanslarda çalışabilme özelliği;
g-) Uyku modunda (Sleep mode) 1μA gibi küçük bir akım çekmesi;
h-) 14 bitlik komut işleme hafızası;
i-) Yalnızca 2 kondansatör ve bir direnç ile çalışabilme özelliği;
j-) RISC mimarisine sahip olması;
k-)Temin etme kolaylığı.

### 1.2.3 PIC'in Özellikleri

Güvenirlik**:**PIC komutları bellekte çok az yer kaplarlar. Dolayısıyla bu komutlar 12 veya 14 bitlik bir program bellek sözcüğüne sığarlar. Harward mimarisi kullanılmayan mikrodenetleyicilerde yazılım programının veri kısmına atlama yaparak bu verilerin komut gibi çalışmasını sağlamaktadır . Bu ise büyük hatalara yol açmaktadır. PIC' lerde bu durum engellenmiştir.

Hız: PIC oldukça hızlı bir mikrodenetleyicidir. Her bir komut satırı 1µsn'lik bir zaman diliminde işlenir. Örneğin 5 milyon komutluk bir programın 20Mhz' lik bir kristalle işletilmesi yalnız 1sn sürer. Bu süre kabaca 386 diye tanımladığımız sayısal bilgisayarın hızının yaklaşık 2 katıdır. Ayrıca PIC'lerin RISC mimarisine sahip olmasının hıza etkisi oldukça büyüktür.

Komut Takımı: PIC'te bir işlem gerçekleştirmek için kullanılacak komut sayısı oldukça azdır. Örneğin PIC16F8XX ailesinde 33 komutu kullanarak sınırsız sayıda işlem yapabilmek mümkündür.

Statik işlem: PIC mikrodenetleyici tamamıyla statik bir işlemcidir. Bu da demek oluyor ki işlemciye pals sağlayan osilasyon kaynağı durdurulsa bile işlenen veriler muhafaza edilmektedir.

Sürme özelliği: PIC'ler yüksek bir sürme kapasitesine sahiptir. Çıkış olarak tanımlanan pinlerin yalnız birinin aktif olması halinde 40mA çekilebilmektedir. Entegre elemanın tamamı düşünüldüğünde ise 150 mA'e kadar akım çekilebilmektedir.

Güvenlik: PIC üretim özelliği itibariyle bir protect yani koruma bitine sahiptir. Bu bitin programlanması yolu ile PIC içerisine yazılan programın başkaları tarafından okunması ve kopyalanmasına engel olunmuş olunur.

Flash olma özelliği: Bu özellik PIC'in yeniden programlanabilir olması durumunu ifade etmektedir. Yani PIC üzerine yazılan program geliştirme amacı ile silinebilir ve yeni bir program yüklenebilir.

### 1.2.4 Bir PIC'in İşlem Yapabilmesi İçin Gerekli Bileşenler

Giriş-Çıkış (I/O): Mikrodenetleyicinin dış dünya ile ilişkisini sağlayan, girdi ve çıktı şeklinde ayarlanabilen bir bağlantı pinidir.

Yazılım: Mikrodenetleyicinin çalışmasını ve işletilmesini sağlayan bilgidir. Başarılı bir uygulama için yazılım hatasız olmalıdır. Yazılım C, Pascal veya Assembler gibi çeşitli dillerde veya ikilik(binary) olarak yazılabilir

Donanım: Mikrodenetleyiciyi, bellek, arabirim bileşenleri, güç kaynakları, sinyal düzenleyici devreler ve bunları çalıştırmak ve arabirim görevini üstlenmek için bu cihazlara bağlanan tüm bileşenlerdir

Simülatör: PC üzerinde çalışan ve mikrodenetleyicinin içindeki işlemleri simüle eden MPSIM gibi bir yazılım paketidir. Hangi olayların ne zaman meydana geldiği biliniyorsa bir simülatör kullanmak tasarımları test etmek için kolay bir yol olacaktır. Öte yandan simülatör, programları tümüyle veya adım adım izleyerek hatalardan arındırma fırsatı sunar. Şu anda en gelişmiş simülatör programı Microchip firmasının geliştirdiği MPLAB programıdır.

ICE: PIC MASTER olarak da adlandırılır. (In- Circuit Emulator / İç devre takipçisi) PC ve Mikrodenetleyicinin yer alacağı soket arasına bağlanmış yararlı bir gereçtir. Bu gereç yazılım, PC de çalışırken devre kartı üzerinde bir mikrodenetleyici gibi davranır. ICE, bir programa girilmesini, mikro içinde neler olduğunu ve dış dünyayla nasıl iletişim kurulduğunun izlenilmesini mümkün kılar.

Programlayıcı**:** Yazılımın mikrodenetleyicinin belleğinde programlamasını ve böylece ICE' nin yardımı olmadan çalışmasını sağlayan bir birimdir. Çoğunlukla seri port 'a (örneğin ICSTART, PROMASTER) bağlanan bu birimler çok çeşitli biçim, ebat ve fiyatlara sahiptir.

Kaynak Dosyası: Hem asembler' in hem de tasarımcının anlayabileceği dilde yazılmış bir programdır. Kaynak dosya mikrodenetleyicinin anlayabilmesi için önceden assemble edilmiş olmalıdır.

Assembler: Kaynak dosyayı bir nesne dosyaya dönüştüren yazılım paketidir. Hata araştırma bu paketin yerleşik bir özelliğidir. Bu özellik assemble edilme sürecinde hatalar çıktıkça programı hatalardan arındırırken kullanılır. MPASM, tüm PIC ailesini elinde tutan Microchip' in son assemble edicisidir.

Nesne dosyası (object file): Assembler tarafından üretilen bu dosya; programcı, similatör veya ICE' nin anlayabilecekleri ve böylelikle dosyanın işlevlerinin çalışmasını sağlayabilecekleri bir dosyadır. Dosya uzantısı assemble edicinin emirlerine bağlı olarak, .OBJ veya .HEX olur.

### 1.3. PIC 16F877

#### 1.3.1 Genel Tanımlama

PIC 16F877 yüksek performanslı, CMOS, full-statik, 8 bit mikrodenetleyicidir. Tüm PIC 16/17 mikrodenetleyicileri gibi PIC 16F877 de RISC mimarisini kullanmaktadır. PIC16F87X mikroları birçok esas özelliklere sahiptir. 14 seviyeli, derin küme ve çoklu iç ve dış kesme kaynaklarına sahiptir. 2 aşamalı komut hattı tüm komutların tek bir saykıl' la (çevrimle) işlenmesini sağlamaktadır. Yalnızca bazı özel komutlar 2 saykıl çekerler. Bu komutlar dallanma komutlarıdır. PIC16F87X ailesi dış elemanları azaltacak spesifik özelliklere sahiptir ve böylece maliyet minimuma inmekte, sistemin güvenirliği artmakta, enerji sarfiyatı azalmaktadır. Bunun yanı sıra tüm PIC'lerde 4 adet osilatör seçeneği mevcuttur. Bunlarda tek pinli RC osilatör, düşük maliyet (4 MHZ) , LP osilatör (Kristal veya seramik rezonatör) , enerji sarfiyatını minimize etmekte (asgari akım) (40 KHZ), XT kristal veya seramik rezonatör osilatörü standart hızlı ve HS kristal veya seramik rezonatörlü osilatör çok yüksek hıza sahiptir (20 MHZ). PIC mikrodenetleyicilerinin en büyük özelliği sleep modu özelliğidir. Bu mod sayesinde işlem yapılmadığı durumlarda PIC uyuma moduna geçerek çok düşük akım çeker. Kullanıcı bir kaç iç ve dış kesmelerle PIC' i uyuma modundan çıkarabilmektedir. Yüksek güvenilirlikli Watchdog Timer kendi bünyesindeki çip üstü RC osilatörü ile yazılımı kilitlemeye karşı korumaktadır. PIC16F877 EEPROM program belleği , aynı aygıt paketinin orjinali ve üretimi için kullanılmasına olanak vermektedir. Yeniden programlanabilirliği mikroyu uygulamanın sonundan kaldırmadan kodu güncelleştirmeye izin vermektedir. Bu aygıtın kolayca erişilemediği, fakat prototipinin kod güncelleştirmesi gerekli olduğu durumlarda, bir çok uygulamanın geliştirilmesinde yararlıdır. Bunun yanı sıra bu kodun güncelleştirilmesi diğer ayrı uygulamalarda da yararlıdır.

### 1.3.2 PIC 16F877'nin Genel Özellikleri

• Yüksek hızlı RISC işlemciye sahiptir;
• 35 adet komut mevcuttur;
• Tüm komutlar 1 saykıl çeker, (Dallanma komutları 2 saykıl çeker.);
• 20 Mhz'ye kadar işlem hızına sahiptir;
• 8Kx14 word'lük flash program belleği mevcuttur;
• 368x8 bayt'lık data belleği;
• 256x8 byte'lık EEPROM data belleği;
• PIC16C73B/74B/76/77 ile uyumlu pin yapısı;
• Doğrudan ve dolaylı adresleme;
• Power-on Reset (POR), Power-up Timer (PWRT) , üzerinde bulunan RC osilatör ile çalışan Watchdog Timer (WDT);
• Programlanabilen kod koruma;
• Enerji tasarrufu için uyku (SLEEP) modu;
• Düşük güçlü yüksek hızlı CMOSFLASH/EEPROM teknolojisi;
• Tamamen statik dizayn;
• Devre üzerinde seri programlama;
• 5 V'luk kaynak ile çalışma;
• 2 V ile 5.5 V arasında işlem yapabilme özelliği;
• Düşükj güç harcaması;
- < 2 mA tyPICal @ 5V, 4 MHz
-20 mA tyPICal @ 3V, 32 kHz
-< 1 mA tyPICal standby

### 1.3.3 PIC 16F877'nin Belirleyici Özellikleri

• Timer0: 8 bit pescaler'e sahip 8bit zamanlayıcı/sayıcı,

• Timer1:Sleep modunda artış gösterebilen ve harici saat darbesiyle artırılabilen Prescaler' li 16 bit zamanlayıcı/sayıcı,

• Timer2:8bit peryot kaydedicili,prescaler ve postscalerli 16bit zamanlayıcı/sayıcı,

• İki adet tutma,karşılaştırma, PWM modülü

• 200ns çözünürlükte 16 bitlik karşılaştırma

• 10 bit çok kanallı Analog-Dijital çevirici

• 9 bit adres saptamaya sahip USART/SCI

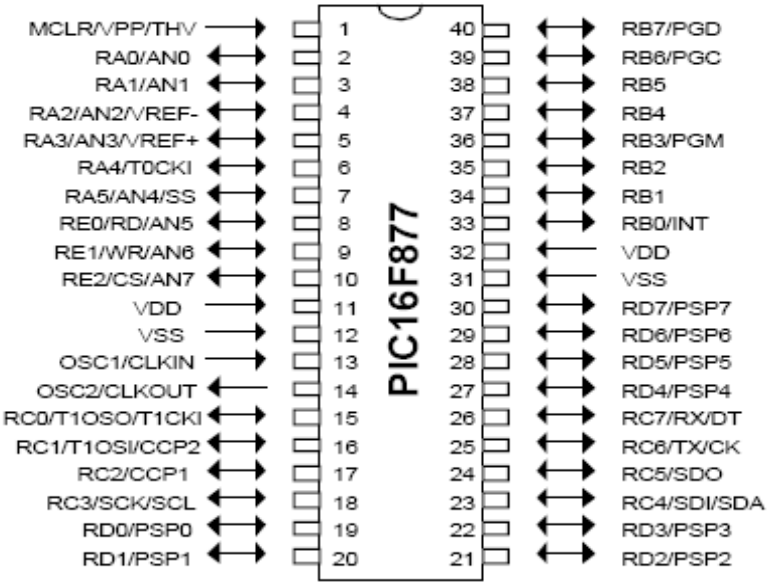
• 8 bit genişliğinde paralel slave port

Aşağıdaki tabloda PIC16F877'nin PIC 16F84 ile karşılaştırılması yer almaktadır.

Tablo 1.1: PIC 16F877 ile PIC 16F84'ün karşılaştırılması

| **ÖZELLİKLER** | **PIC 16F877** | **PIC 16F84** |
|---|---|---|
| Çalışma hızı | DC-20 Mhz | DC-10 Mhz |
| Program belleği | 8Kx14 word Flash ROM | 1Kx14 word Flash ROM |
| Eeprom belleği | 256 byte | 64 byte |
| Kullanıcı RAM | 368x8 byte | 68 x 8 byte |
| Giriş/Çıkış port sayısı | 33 | 13 |
| Timer | Timer0,Timer1,Timer2 | Timer0 |
| A/D Çevirici | 8 kanal 10 bit | YOK |
| Capture/Comp/PWM | 16 bit Capture 16 bit compare<br>10 bit PWM çözünürlük | YOK |
| Seri çevresel arayüz | SPI ve $I^2C$ modunda SPI portu<br>(senkron seri port) | YOK |
| Parelel slave port | 8 bit, harici RD,WR ve CS kontrollu | YOK |
| USART/SCI | 9 bit adresli | YOK |

### 1.3.4 PIC 16F877'nin Fiziksel Yapısının İncelenmesi



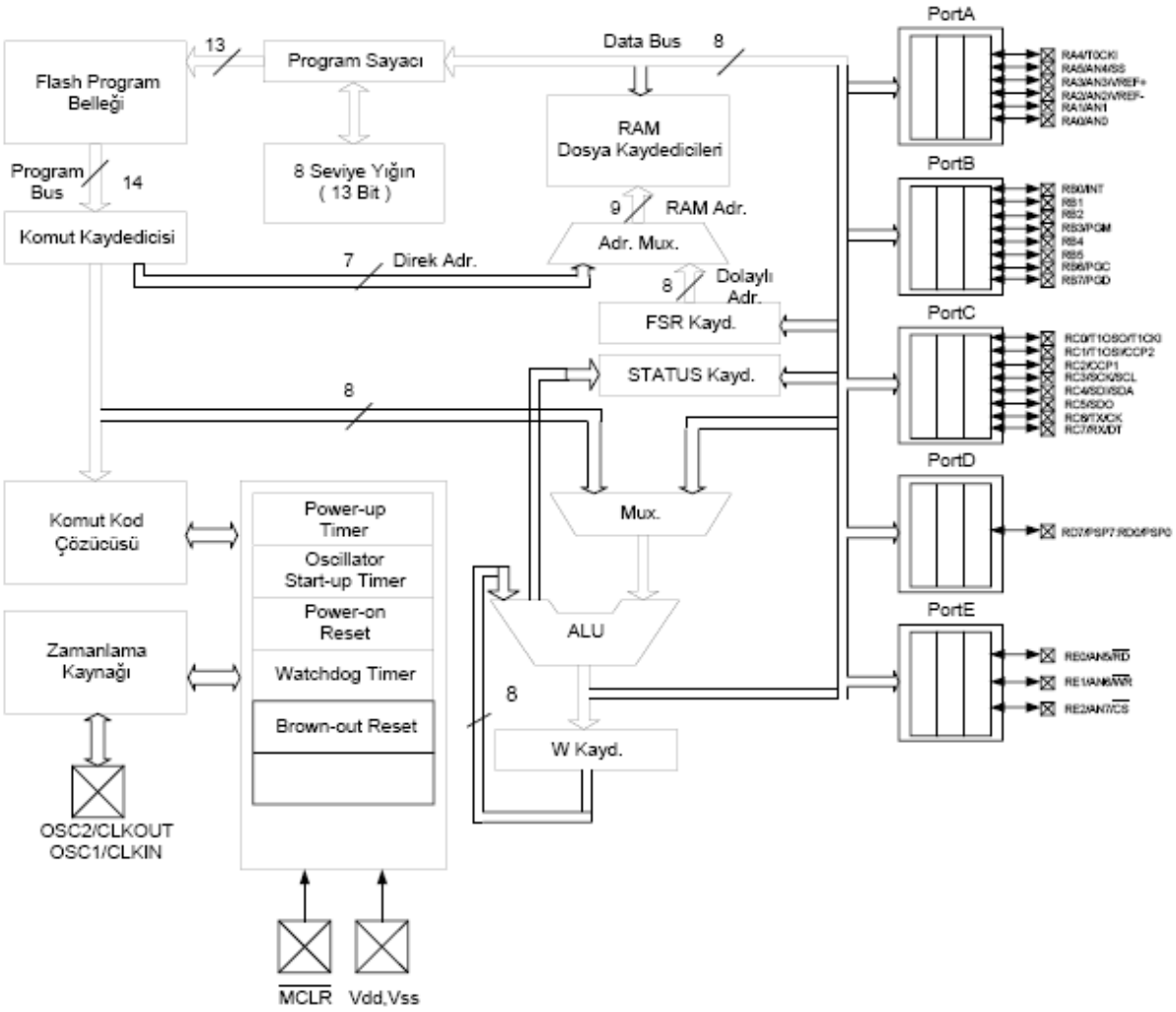

Şekil 1.1. PIC 16F877'nin bacak yapısı

### 1.3.5 PIC 16F877 Pin Tanımlamaları

Tablo 1.2. IC 16F877'de Pin Tanımlamaları (P:Power, G/Ç:Giriş/Çıkış, ST: Schmitt Trigger giriş, G:Giriş, Ç:Çıkış, TTL:Transistor-Transistor lojik giriş)

| | | | | |
|---|---|---|---|---|
| OSC1/CLKIN | 13 | G | ST/CMOS | Kristal osilatör girişi/Harici osilatör kaynağı girişi |
| OSC2/CLKOUT | 14 | Ç | — | Kristal osilatör çıkışı.RC osilatör modunda 1/4 f değerinde frekans çıkışı |
| MCLR/VPP/THV | 1 | G/P | ST | Mikrodenetleyici için reset ucu. Normal çalışmada 1 seviyesinde tutulur |
| RA0/AN0 | 2 | G/Ç | TTL | PORTA:G/Ç olarak yönlendirilebilir port.Pinler G/Ç görevi dışında;<br>RA0: 0. Analog giriş görevi yapar.<br>RA1: 1. Analog giriş görevi yapar.<br>RA2: 2. Analog giriş veya negatif referans gerilimi girişim görevi yapar.<br>RA3: 3. Analog giriş veya pozitif referans gerilimi girişim görevi yapar.<br>RA4: TIMER0 için clock girişi görevi yapar. Açık drain çıkışa sahiptir.<br>RA5: 4. Analog giriş veya SSP için slave seçimi görevi yapar. |
| RA1/AN1 | 3 | G/Ç | TTL | |
| RA2/AN2/VREF- | 4 | G/Ç | TTL | |
| RA3/AN3/VREF+ | 5 | G/Ç | TTL | |
| RA4/T0CKI | 6 | G/Ç | ST | |
| RA5/SS/AN4 | 7 | G/Ç | TTL | |
| RB0/INT | 33 | G/Ç | TTL/ST | PORTB: G/Ç olarak yönlendirilebilir port.Tüm girişleinde yazılımla programlanabilir düşük değerli pull-uplar vardır.Pinler G/Ç görevi dışında;<br>RB0:Harici kesme ucu görevi yapar.<br>RB3:Düşük seviye programlama girişi görevi yapar.<br>RB6:Seri programlama girişi görevi yapar<br>RB7:Seri programlamada data girişi görevi yapar. |
| RB1 | 34 | G/Ç | TTL | |
| RB2 | 35 | G/Ç | TTL | |
| RB3/PGM | 36 | G/Ç | TTL | |
| RB4 | 37 | G/Ç | TTL | |
| RB5 | 38 | G/Ç | TTL | |
| RB6/PGC | 39 | G/Ç | TTL/ST | |
| RB7/PGD | 40 | G/Ç | TTL/ST | |
| RC0/T1OSO/T1CKI | 15 | G/Ç | ST | PORTC: G/Ç olarak yönlendirilebilen port .Pinler G/Ç görevi dışında;<br>RC0:TIMER1 osilatör çıkışı veya TIMER1 clock çıkışı görevi de yapar.<br>RC1:TIMER1 osilatör girişi veya Capture2-G/Compare2-O/PWM2-Ç görevi yapar.<br>RC2:Capture1-G/Compare1-Ç/PWM1-Ç görevi de yapar.<br>RC3: SPI ve I2C modunda senkron seri clock G/Ç görevi yapar.<br>RC4. SPI modunda SPI data giriş, I2C modunda data G/Ç görevi yapar.<br>RC5: SPI modunda SPI data çıkış görevi yapar.<br>RC6:USART asenkron gönderme veya senkron clock görevi yapar.<br>RC7: USART asenkron alma ve senkron data görevi yapar. |
| RC1/T1OSI/CCP2 | 16 | G/Ç | ST | |
| RC2/CCP1 | 17 | G/Ç | ST | |
| RC3/SCK/SCL | 18 | G/Ç | ST | |
| RC4/SDI/SDA | 23 | G/Ç | ST | |
| RC5/SDO | 24 | G/Ç | ST | |
| RC6/TX/CK | 25 | G/Ç | ST | |
| RC7/RX/DT | 26 | G/Ç | ST | |
| RD0/PSP0 | 19 | G/Ç | ST/TTL | PORTD:G/Ç olarak yönlendirilebilir port veya mikroişlemci hattında arabirim olarak kullanıldığında paralel slave port. |
| RD1/PSP1 | 20 | G/Ç | ST/TTL | |
| RD2/PSP2 | 21 | G/Ç | ST/TTL | |
| RD3/PSP3 | 22 | G/Ç | ST/TTL | |
| RD4/PSP4 | 27 | G/Ç | ST/TTL | |
| RD5/PSP5 | 28 | G/Ç | ST/TTL | |
| RD6/PSP6 | 29 | G/Ç | ST/TTL | |
| RD7/PSP7 | 30 | G/Ç | ST/TTL | |
| RE0/RD/AN5 | 8 | G/Ç | ST/TTL | PORTE: G/Ç olarak yönlendirilebilir port . Pinler G/Ç görevi dışında;<br>RE0:Paralel Slave porttan okuma kontrolu veya 5. analog giriş görevi yapar.<br>RE1:Paralel Slave porttan yazma kontrolu veya 6. analog giriş görevi yapar.<br>RE2:Paralel Slave porttan seçim kontrolu veya 7. analog giriş görevi yapar. |
| RE1/WR/AN6 | 9 | G/Ç | ST/TTL | |
| RE2/CS/AN7 | 10 | G/Ç | ST/TTL | |
| VSS | 12,31 | P | — | Mikrodenetleyici için toprak seviyesini oluşturur. |
| VDD | 11,32 | P | — | Mikrodenetleyici için pozitif kaynak gerilimini oluşturur. |

### 1.3.6 PIC 16F877'nin Basitleştirilmiş İç Yapısı



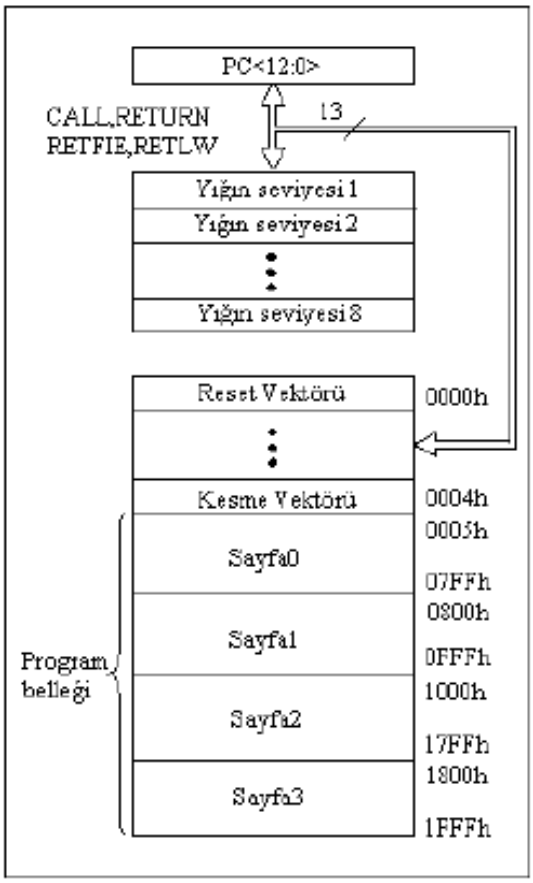

Şekil 1.2. PIC 16F877 basitleştirilmiş iç yapısı

### 1.3.7 Bellek Organizasyonu

Her PIC mikrodenetleyicisinde 3 bellek bloğu bulunmaktadır. Bunlar program belleği,veri belleği ve bunları ayıran veri hattıdır. Her bir bellek kendi taşıyıcısına sahiptir; böylece her bir bloğa erişim aynı osilatör süreci boyunca meydana gelebilmektedir. Bunun ötesinde, veri belleği genel amaçlı RAM ve özel fonksiyon kayıtları (SFR) olmak üzere ikiye bölünür. SFR`ler her bir bireysel özelleşmiş modülü ele alan bölümde açıklanan özel modülleri kontrol etmek için kullanılmaktadır. Veri belleği EEPROM veri belleğini de içermektedir. Bu bellek, direkt veri belleğine planlanmamış, fakat indirekt olarak planlanmıştır; ve indirekt adres göstergeleri okumak/yazmak için EEPROM belleğinin adresini belirlemektedir.

### 1.3.7.1 Program Bellek Organizasyonu

PIC 16F877 13 bit program sayacına ve 8Kx14 adresleme kapasitesine sahiptir. PIC16F877 denetleyicisi 8Kx14 FLASH program belleğine sahiptir. Reset vektörü 0000h ve kesme vektörü 0004h adresindedir. Şekil 1.3'te PIC16F877 program bellek haritası görülmektedir.




Şekil 1.3. PIC16F877 program bellek haritası

### 1.3.7.2 Veri Bellek Organizasyonu

Veri belleği genel amaçlı yazmaçlar ve özel işlev yazmaçları (SFR) olmak üzere ikiye ayrılır. RP0 ve RP1 bitleri küme seçimi için ayrılmış bitlerdir. Her bir bank (küme) 7Fh' ye kadar (128 bayt) uzanır. Her bank'ın alt kısımları özel işlev yazmaçları için ayrılır. Üstteki özel işlev yazmaçları ise statik RAM olarak kullanılan yazmaçlardır. Bütün banklarda özel işlev yazmaçları vardır. Özel işlev yazmaçlarındaki yüksek kullanım bir banktan kod indirgemesi ve hızlı erişim için başka bankta gösterilebilir.

| RP1 | RP0 |
|---|---|

= 00 Bank0

= 01 Bank1

= 10 Bank2

= 11 Bank3

### 1.3.8 Özel Fonksiyon Kaydedicileri

Özel fonksiyon kaydedicileri gerçek bellek birimleri olarak gözükmeseler de, PIC içerisinde veri belleği adreslerinde tanımlanmış sıradan bellek hücreleridir. Bu kaydediciler programlama esnasında bir nevi kayıt tutma görevi üstlenirler. Tablo 1.3'te özel fonksiyon kaydedicilerinin isimleri ve adresleri verilmiştir.

Tablo 1.3. PIC 16F877'de Özel Fonksiyon Kaydedicileri ve Adresleri

| BANK 0 | | BANK 1 | | BANK 2 | |
|---|---|---|---|---|---|
| 00 | INDF | 80 | INDF | 100 | INDF |
| 01 | TMR0 | 81 | OPTION_REG | 101 | TMR0 |
| 02 | PCL | 82 | PCL | 102 | PCL |
| 03 | STATUS | 83 | STATUS | 103 | STATUS |
| 04 | FSR | 84 | FSR | 104 | FSR |
| 05 | PORTA | 85 | TRISA | 105 | — |
| 06 | PORTB | 86 | TRISB | 106 | PORTB |
| 07 | PORTC | 87 | TRISC | 107 | — |
| 08 | PORTD | 88 | TRISD | 108 | — |
| 09 | PORTE | 89 | TRISE | 109 | — |
| 0A | PCLATH | 8A | PCLATH | 10A | PCLATH |

Tablo 1.3’ ün devamı

| | | | | | |
|---|---|---|---|---|---|
| 0B | INTCON | 8B | INTCON | 10B | INTCON |
| 0C | PIR1 | 8C | PIE1 | 10C | EEDATA |
| 0D | PIR2 | 8D | PIE2 | 10D | EEADR |
| 0E | TMR1L | 8E | PCON | 10E | EEDATH |
| 0F | TMR1H | 8F | — | 10F | EEADRH |
| | | | | BANK 3 | |
| 10 | T1CON | 90 | — | 180 | INDF |
| 11 | TMR2 | 91 | SSPCON2 | 181 | OPTION_REG |
| 12 | T2CON | 92 | PR2 | 182 | PCL |
| 13 | SSPBUF | 93 | SSPADD | 183 | STATUS |
| 14 | SSPCON | 94 | SSPSTAT | 184 | FSR |
| 15 | CCPR1L | 95 | — | 185 | — |
| 16 | CCPR1H | 96 | — | 186 | TRISB |
| 17 | CCP1CON | 97 | — | 187 | — |
| 18 | RCSTA | 98 | TXSTA | 188 | — |
| 19 | TXREG | 99 | SPBRG | 189 | — |
| 1A | RCREG | 9A | — | 18A | PCLATH |
| 1B | CCPR2L | 9B | — | 18B | INTCON |
| 1C | CCPR2H | 9C | — | 18C | EECON1 |
| 1D | CCP2CON | 9D | — | 18D | EECON2 |
| 1E | ADRESH | 9E | ADRESL | 18E | — |
| 1F | ADCON0 | 9F | ADCON1 | 18F | — |

### 1.3.8.1 STATUS Kaydedicisi

Aritmetiksel işlemlerin yürütüldüğü, sıfırlama işlemlerinin gerçekleştirildiği ve küme kurma işlemlerinin gerçekleştirildiği kaydedicidir.

| R/W-0 | R/W-0 | R/W-0 | R-1 | R-1 | R/W-x | R/W-x | R/W-x |
|---|---|---|---|---|---|---|---|
| IRP | RP1 | RP0 | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |

Şekil 1.4. Status kaydedicisinin bit yerleşimi

Status,0 (C): (Carry Bit)Aritmetiksel işlemler sonucunda taşmaların kaydedildiği bittir. Bu bit bir işlem sonucunda 1 ise taşmanın olduğu, 0 ise taşmanın olmadığı anlaşılır.

Status,1(DC): (Digit Carry Bit)Bu bit; işlem sonucunda ilk 4 bitin dışına taşma olup olmadığının kaydını tutar. Eğer bu bit 1 ise 5. bite taşma olduğu, 0 ise 5. bite taşma olmadığı anlaşılır.

Status,2(Z): (Zero Bit)İşlem yapılan hücrede sıfırlamanın olup olmadığının kontrolünü yapar. 1 ise kontrolün yapıldığı hücrenin sayısal değeri 0 olmuş demektir, 0 ise hücre sıfırdan farklı bir sayısal değere sahiptir.

Status,3(PD): (Power Down Bit)Sleep modu ve watchdog timer ile ilgili işlemlerde kullanılır. Watchdog timer sıfırlandıktan sonra (CLRWDT) bu bit 1 yapılmalıdır. Mikrodenetleyici sleep moduna sokulur iken de bu bit 0 yapılarak işleme başlanır.

Status,4(TO): (Time Out Bit)PD biti gibi bu bit de sleep modu ve watchdog timer ile ilgili işlemlerde kullanılır. Watchdog timerin sıfırlanması işlemi sonunda bu bit 0 yapılırken, mikrodenetleyici sleep modundan çıkarılırken bu bit 1 yapılır.

Status,5,6(RP0-RP1): (Register Bank Select Bits) Küme seçimi ile ilgili işlemlerde kullanılırlar. İki bit bir arada düşünüldüğünde 4 konum mevcuttur ve bu dört konumun her birisinde de ayrı bir küme kurulmaktadır.(00=Bank0, 01=Bank1, 10=Bank2, 11=Bank3). Direkt adresleme modunda kullanılan bir bittir.

Status,7 (IRP): Status5,6 gibi bu bit de küme seçimi işlemlerinde kullanılır. Ancak bu bit dolaylı adresleme durumlarında kullanılır.( 1 = Bank 2, 3 0 = Bank 0, 1 )

### 1.3.8.2 OPTION Kaydedicisi

Bu kaydedici üzerinde portB, TMR0 ve dış kesmeleri düzenleyici bitler bulunmaktadır.

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{RBPU}}$ | INTEDG | TOCS | TOSE | PSA | PS2 | PS1 | PS0 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |

Şekil 1.5. Option Kaydedicinin bit yerleşimi

Option,0,1,2(PS0,PS1,PS2):(Prescaler Rate Select Bits)Timer0(TMR0) ve Watchdog Timer (WDT)'ın süreç oranlarını belirleyen bitlrdir. Üç bit bir arada düşünüldüğü zaman 8 farklı durum ortaya çıkmaktadır. 8 durum için TMR0 ve WDT'ın aldığı durumlar Tablo 3.4 teverilmiştir.

Tablo 1.4. PS2,PS1,PS0'a göre TMR0,WDT oranları

| PS2,PS1,PS0 | TMRO Oranı | WDT Otranı |
|---|---|---|
| 000 | 1:2 | 1:1 |
| 001 | 1:4 | 1:2 |
| 010 | 1:8 | 1:4 |
| 011 | 1:16 | 1:8 |
| 100 | 1:32 | 1:16 |
| 101 | 1:64 | 1:32 |
| 110 | 1:128 | 1:64 |
| 111 | 1:256 | 1:128 |

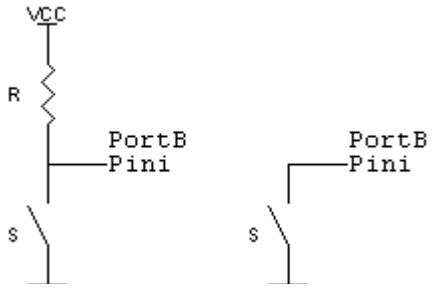
Option,3(PSA):(Prescaler Assignment Bit) Aldığı duruma göre TMR0 veya WDT'a tahsis edilir. Eğer 0 ise TMR0 üzerinde yapılan işlemlerde kullanılır, 1 ise WDT üzerinde yapılan işlemlerde kullanılır.

Option,4(TOSE):(Timer0 Source Edge select bit) Devrede harici clock kaynağı kullanılması durumunda palslerin düşen kenar mı yoksa yükselen kenar mı olduğunun tayininde işlem yapar. Eğer 1 ise RA4/TOCKI pininde kullanılan palslerin düşen kenar, 0 ise RA4/TOCKI pinindeki palslerin yükselen kenar olması gerekir.

Option,5(TOCS):(Timer0 Source select) PIC ile yapılan devrede hangi pals kaynağının kullanılacağına karar vermek için kullanılır. Eğer 1 ise palsler RA4/TOCKI pininden, 0 ise dahili pals kaynağından (CLKOUT) sağlanır.

Option,6(INTEDG):(Interrupt edge select )RBO/INT pininde kesme sinyali olması halinde yükselen kenar veya düşen kenar tetikleme gönderilmesi durumlarını ayarlar. 1 ise RBO/INT pininden yükselen kenar, 0 ise RBO/INT pininden düşen kenar bir tetikleme uygulanmalıdır.

Option,7(RBPU):(PortB Pull-up Control ): Bu bit uygulamalarda alan daralmasını sağlayan bir bittir. Bu bitin 1 yapılması ile gerilimin pozitif ucu ile portB pini arasına bir direnç konulur. Bu direnç sayesinde PortB'nin bu pini lojik 1'de tutulmuş ulur. Bu pini lojik 0 yapmak için ise pin ile gerilimin negatif ucu arasını bir push buton ile kısa devre yapmak yeterli olacaktır. Bu bitin 0 yapılması halinde ise pinledeki pull-up dirençleri kaldırılır. Bu biti kullanırken yalnız portB için pull-up direnci konulduğunu unutmamak gerekir.




Şekil 1.6. RBPU'nun analog düzen karşılığı

Şekil 1.6.a'da RBPU'nun kullanılmadığı durumda kararlılık için B portuna takılması gereken harici direncin bağlantısı görülmektedir. Şekil 1-6-b'de ise RBPU'nun kullanıldığı durum verilmiştir. Bu durumda harici bir dirence ihtiyaç duyulmamıştır.

### 1.3.8.3 INTCON Kaydedicisi

Bu kaydedici tüm kesmelerin kontrolü için bazı okunabilir ve yazılabilir bitler sağlayabilmektedir.

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-x |
|---|---|---|---|---|---|---|---|
| GIE | PEIE | TOIE | INTE | RBIE | TOIF | INTF | RBIF |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |

Şekil 1.7. Intcon kaydedicisinin bit yerleşimi

Intcon,0 (RBIF): B portuna ait kesme bayrak bitinde değişiklikler yapmaya yarar. Program başında RB7:RB4 bitlerinin sıfırlandığı varsayılır ise RB7:RB4 pinlerinden herhangibirisinin durum değiştirmesi halinde RBIF 1 olur. (0'dan 1'e) 1'den sıfıra düşme durumunda ise RBIF lojik olarak konum değiştirir ve 0 olur.

Intcon,1(INTF): RB0/INT pini ile ilişkilendirilmiştir. Bu bitin 0 olması durumunda RB0/INT aktif değildir, 1 olması durumunda ise aktiftir.

Intcon,2 (TOIF): TMRO da meydana gelen taşmaya göre işlem yapar. Eğerki TOIF 1 ise TMRO'da taşma vardır. Aksi durumda ise taşma yoktur.

Intcon,3 (RBIE): İşlevi RBIF'inkine benzer. PortB kesme yetkilendirme bitidir. 1 olması durumunda PortB kesme yetkilendirmesi aktif, diğer durumda ise aktif değildir.

Intcon,4 (INTE): RB0/INT kesme yetkilendirme bitidir. Bu bitin 1 olması halinde RB0/INT pini kesme yetkilendirmesi verilmiştir. 0 olması halinde RB0/INT pininde kesme yetkilendirmesi yoktur.

Intcon,5 (TOIE): TMR0 kesmesini yetkilendirme bitidir. TOIE=1 ise kesme yetkilendirmesi aktif durumdadır. 0 olması halinde ise TMR0 'daki kesme ketkilendirmesi kalkmış durumdadır.

Intcon,6 (PEIE):Çevresel Kesmeler için yetkilendirme bitidir.PEIE=1 olması durumunda çevresel kesmeler etkindir,PEIE= 0 olması durumunda çevresel kesmeler etkin değildir.

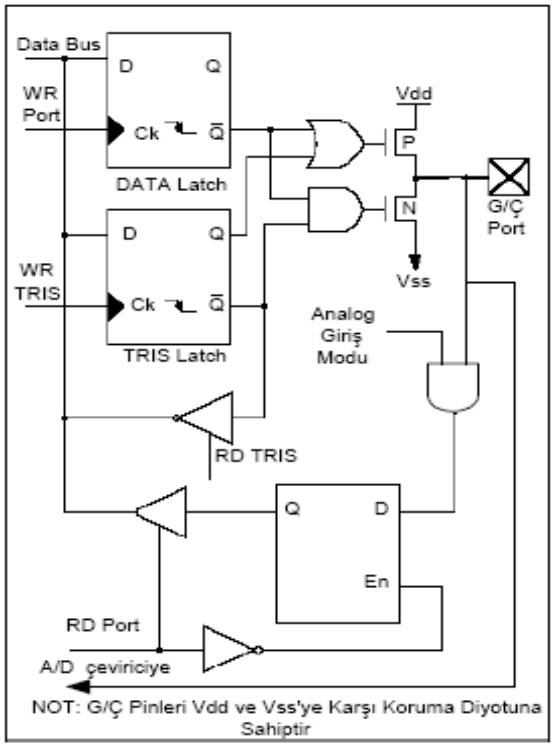
Intcon,7 (GIE):Geniş kapsamlı bir yetkilendirme bitidir. Bu bitin 1 yapılması ile program içerisinde tüm yetkilendirmeler aktif hale getirilebilir. 0 olma durumunda ise verilen tüm yetkilendirmeler kaldırılabilir.

### 1.3.9 Giriş/Çıkış Portları (G/Ç Portları)

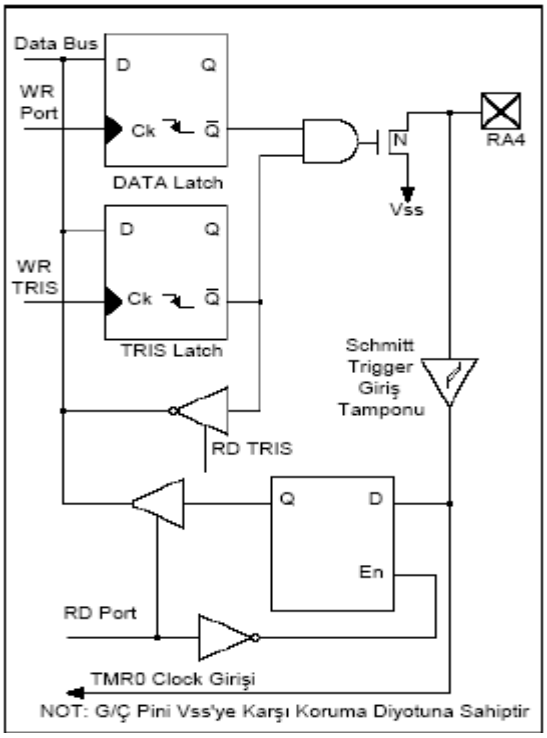
G/Ç portları giriş/çıkış vazifelerinin dışında bazı çevresel işlemleri de yapacak özelliklere sahiptirler. Çevre birimleri kullanıldığında genel amaçlı giriş/çıkış pini kullanılmaz.

#### 1.3.9.1 PORTA ve TRISA Kaydedicisi

PORTA 6 bit giriş/çıkış olarak yönlendirilebilir porttur. Bu portu yönlendiren yazmaç ise TRISA yazmacıdır. TRISA kaydındaki herhangi bir bit 1 ise buna uygun çıkış sürücüsü yüksek direnç moduna getirilecektir. TRISA kaydındaki herhangi bir bitin 0 olması durumunda ise çıkış mandalı seçilen pinin üzerine getirilir. Analog giriş kullanıldığında TRISA yazmacı RA pininin yönünü kontrol eder. Şekil 3.12.a ve şekil 3.12.b A portunun iç yapısını göstermektedir.




Şekil 1.8.a. RA0:RA3 ve RA5 İç yapısı




Şekil 1.8.b. R A4/TOCKI İç Yapısı

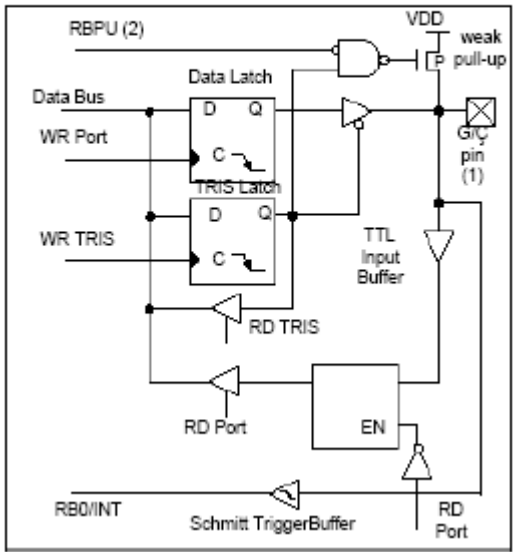
Örnek 3.1. A portunun G/Ç olarak kurulması:

```
bcf status, rp0          ;
bsf status, rp0          ; Bank 1 seçildi.
movlw b'00000101'  ; PortA nın giriş ve çıkış olacak pinleri belirleniyor
movwf trisa              ; Giriş için RA0,RA2
                         ; Çıkış için RA1,RA3,RA4,RA5
```

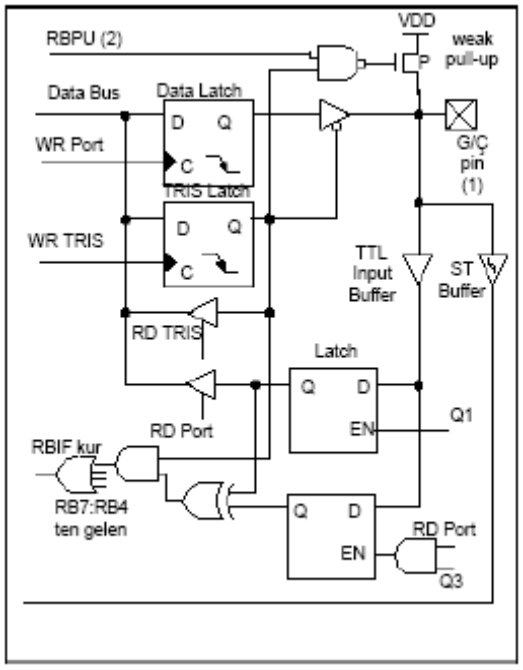
### 1.3.9.2 PORTB ve TRISB Kaydedicisi

PORTB 8 bit giriş/çıkış olarak yönlendirilebilir porttur. Bu portu yönlendiren yazmaç ise TRISB yazmacıdır. TRISB kaydındaki herhangi bir bit 1 ise buna uygun çıkış sürücüsü yüksek direnç moduna getirilecektir. TRISB kaydındaki herhangi bir bitin 0 olması durumunda ise çıkış mandalı seçilen pinin üzerine getirilir. Analog giriş kullanıldığında TRISB yazmacı RB pininin yönünü kontrol eder. (1)

Her bir PORTB pini iç direnç düşürücü engellere sahiptir. RBPU(OPTION – REG<7) bitinin silinmesiyle aktif yapılır. Düşürücü engeller, port pini çıkış olarak konfigüre edildiği zaman otomatik olarak kapanmaktadır. Ayrıca dört PORTB pini, RB7: RB4 değişim özelliklerinde kesmelere sahiptir. Yalnızca giriş olarak konfigüre edilen pinlerkesmenin meydana gelmesine sebep olabilirler. (yani, herhangi bir çıkış olarak şekillendirilen RB7:RB4 pini değişim ilişkisi üzerindeki kesmeden hariç tutulmuştur). Giriş modundaki pinlerin değeri PORTB` nin önceki okunmasındaki eski değeri ile karşılaştırılır. Pinlerin "uyuşmayan" kısımları RB port değişim kesmesini üretmek için birlikte OR'lanır. Şekil 1.9.a ve şekil 1.9.b portunun iç yapısını göstermektedir.




Şekil 1.9.a. RB3:RB0 Pinleri İç Yapısı




Şekil 1.9.b. R7:RB4 Pinleri İç Yapısı

Örnek 3.2. B Portunun Kurulması:

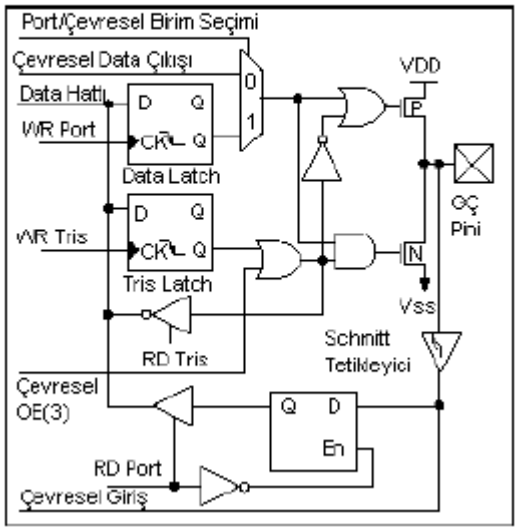
```
bcf status, rp0          ;
bsf status, rp0          ; Bank 1 seçildi.
movlw b'10111111'  ; PortB nın G/Ç olacak pinleri belirleniyor
movwf trisb              ; Giriş için RB<3:0>
                         ; Çıkış için RB<5:4>
```

**1.3.9.3 PORTC ve TRISC Kaydedicisi**

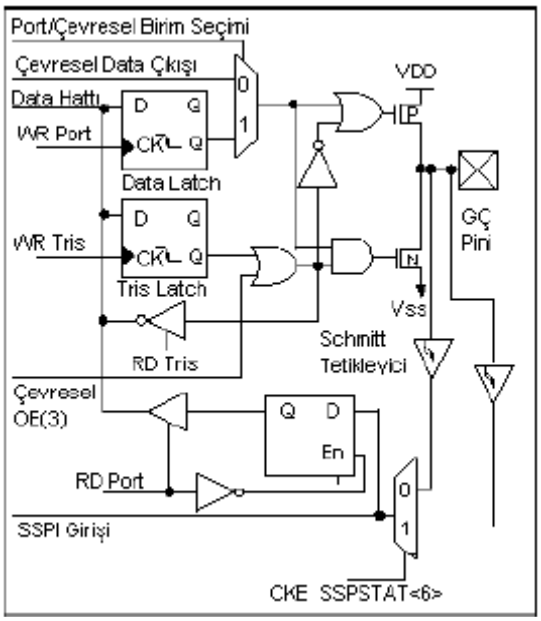
PORTC 8 bit G/Ç olarak yönlendirilebilir porttur. Bu portu yönlendiren yazmaç ise TRISC yazmacıdır. TRISC Kaydındaki herhangi bir bit 1 ise buna uygun çıkış sürücüsü yüksek direnç moduna getirilecektir. TRISC Kaydındaki herhangi bir bitin 0 olması durumunda ise çıkış mandalı seçilen pinin üzerine getirilir. Analog giriş kullanıldığında TRISC yazmacı RC pininin yönünü kontrol eder. Şekil 1.10.a ve şekil 1.10.b C portunun iç yapısını göstermektedir.

Örnek 3.3. C Portunun Kurulması;

```
bcf status, rp0          ;
bsf status, rp0          ; Bank 1 seçildi.
movlw b'10111111'  ; Veri yönlendirmek
                         ; kullanılan değer
movwf trisc              ; Giriş için RC<3:0>
                         ; Çıkış için RC<5:4>
```



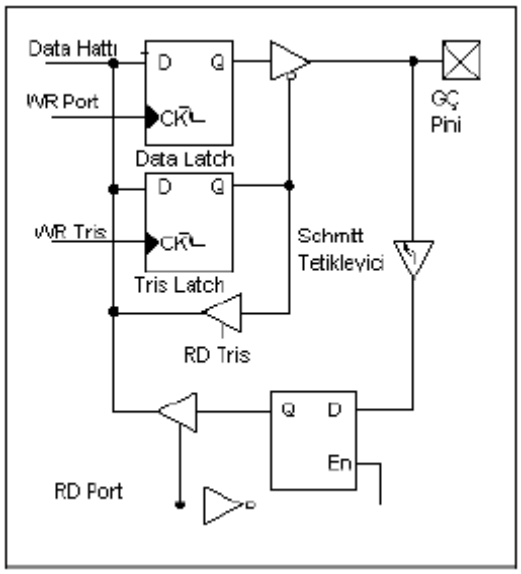

Şekil 1.10.a. RC0:2, RC5:7 Pinlerinin Yapısı




Şekil 1.10.b. RC3:4 Pinlerinin İç Yapısı

### 1.3.9.4 PORTD ve TRISD Kaydedicisi

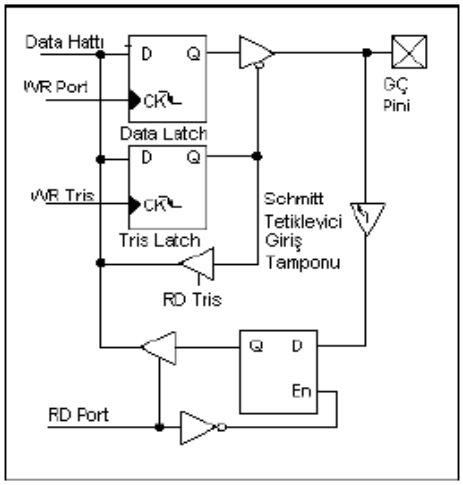
PORTD 8 bit Schmitt trigger tampon girişli porttur. Bu port PSPMODE (TRISD) denetim biti tarafından kurulur. Giriş tamponu TTL tampondur. Şekil 1.11 de D portunun blok diyagramı görülmektedir.




Şekil 1.11. D Portu I/O Blok Diyagramı

### 1.3.9.5 PORTE ve TRISE Kaydedicisi

Bu port RE0/RD/AN5, RE1/WR/AN6 ve RE2/CS/AN7 olmak üzere 3 adet pine sahiptir. Giriş veya çıkış portu olarak ayarlanabilir. PORTE kontrolü TRISE tarafından yapılır. PSPMODE(TRISE<4>) denetim biti tarafından ayarlanır. Şekil 1.12 da E portu blok diyagramı görülmektedir.




Şekil 1.12. E Portu I/O Blok Diyagramı

### 1.3.10. Zamanlama0 (TIMER0) Modülü ve TMR0 Kaydı

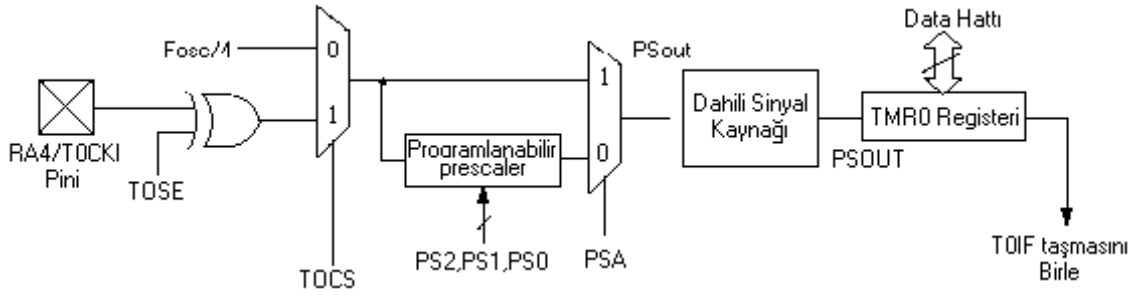
• Timer0 modül, timer/sayaç aşağıdaki özelliklere sahiptir.
• 8 bitlik timer/sayaç ,
• Okunabilir ve yazılabilir ,
• 8 bitlik programlanabilir prescaler.,
• İçten veya dıştan saat ayarı ,
• FFh` tan 00h` ye taşma üzeri kesme ,
• Dış saatin sınır seçimi ,

Timer modu TOCS bitinin (OPTION<5>) temizlenmesiyle seçilir. Timer modunda Timer0 modülü her bir komut sürecini uzatır. Eğer TMR0 kaydı yazılıysa, uzama takip eden 2 süreci engeller. Kullanıcı ayarlanan değeri TMR0 kaydına yazarak, bunun etrafından çalışabilir. Sayaç modu TOCS bitinin (OPTION<5>) ayarlanmasıyla seçilir. Bu modda, TMR0, RA4/TOCK1 pininin sınırlarının herbir artışında ya da düşüşünde artacaktır. Genişleyen sınır, TO kaynak sınır seçim biti tarafından, TOSE (OPTION<4>) tarafından belirlenmektedir. TOSE bitinin temizlenmesi artan sınırları seçecektir.

Prescaler, Timer0 modülü ile Watchdog Timer arasında paylaşılmaktadır. Prescaler ataması, yazılımda PSA biti kontrolü tarafından denetlenmektedir. (OPTION<3>) PSA bitinin temizlenmesi, prescaler' ı Timer0 modülüne atayacaktır. Prescaler okunabilir veya yazılabilir

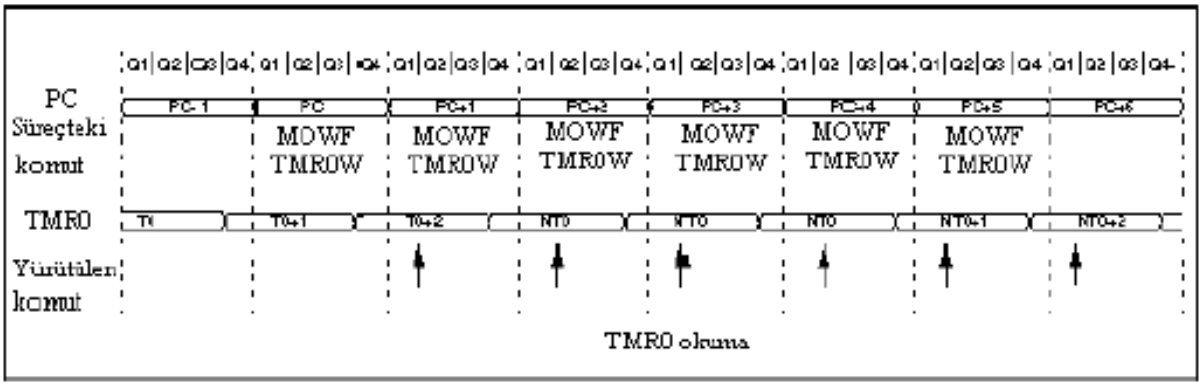
değildir. Prescaler, Timer0 zamanlama modülüne atandığında prescaler değeri (1:2,1:4,1:8,1:16,1:32,1:64,1:128,1:256 olmak üzere) yazılım tarafından seçilebilirdir. (1)

### 1.3.10.1 TMR0 Kesmesi




Şekil 1.13. TMR0 blok diyagramı

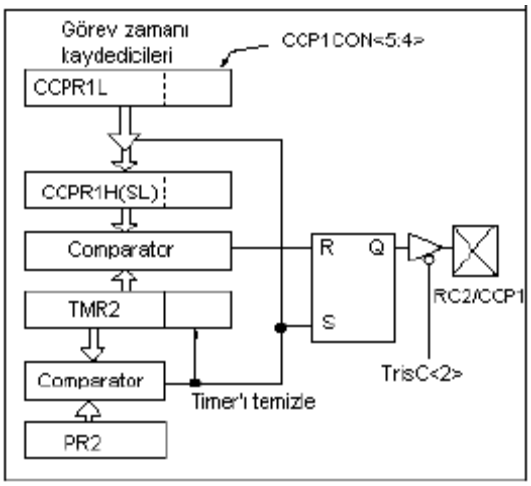
TMR0 kesmesi, TMR0 kaydı FFH`dan 00h`ye akışında üretilmektedir. Bu fazla akım TOIF bitini ( INTCON<2>) kurar (ayarlar). Kesme, aktif TOIE bitinin (INTCON<5>) temizlenmesi ile gizlenebilir. (INTCON<5>) TOIF biti, Timer0 modülü tarafından, bu kesmenin yeniden aktifleştirilmesinden önce yazılımdan silinmelidir. TMR0 kesmesi (Şekil 1.14) denetleyiciyi SLEEP` ten çıkaramaz, çünkü, SLEEP boyunca timer kapalıdır.



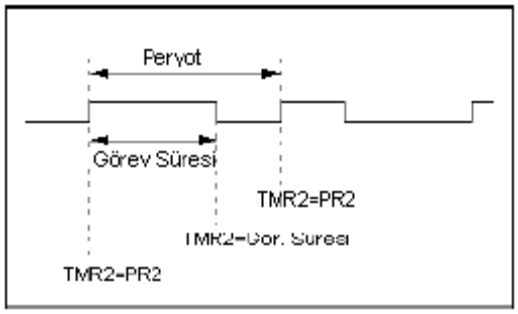

Şekil 1.14. TMR0 Zamanlaması: Prescaler' siz İçten Clock

### 1.3.11.1 PWM (Pals Genişlik Modülasyonu) Modu

Bu modda CCP1 pininden 10 bit çözünürlükte PWM çıkış alınır. PWM modunda işlem yapabilmek için TRISC<2> biti çıkış konumunda tutulmalıdır. Şekil 1.15'de PWM modunda gerçekleştirilen işlemlerin basit blok diyagramı verilmektedir. Şekil 1.16 da ise PIC 16F877'den elde edilecek bir PWM çıkışın şekli verilmiştir.




Şekil 1.15. PWM Moduna Ait Basitleştirlmiş Blok Diyagram




Şekil 3.21. Pwm Çıkış Şekli

#### 1.3.11.1.1 PWM Süresi

Bir PWM işlemi ile verinin aktarılması PR2 kaydedicisinin değerine bağlıdır. (PR2=TMR2 prescaler değeri). PWM süresi formül 3.1 den faydalanılarak hesaplanabilir.

*PWM süresi = [(PR2) + 1] . 4 . TOSC*

PWM frekansı ise 1/PWM süresinden bulunabilir. TMR2'nin PR2'ye eşit olması durumunda ise, sonraki peryotta şu üç durum meydana gelir.

• TMR2 temizlenir.

• CCP1 pini yüksek seviyeye (H) çekilir.

• PWM görev saykılı CCPR1L den CCPR2H içerisine atılır.

#### 1.3.11.1.2 PWM Görev Saykılı

PWM görev saykılı CCPR1L ve kaydedicisi ve CCP1CON<4:5> bitlerine bağlıdır. PWM'de ki 10 bitlik çözünürlük; CCPR1L kaydedicisinin 8 bitlik kısmının MSB, ve CCP1CON'un 4 ve 5. Bitlerinin LSB olarak kabul edilmesiyle elde edilir. Bu 10 bit çözünürlük CCPR1L:CCP1CON<5:4> şeklinde gösterilmektedir. PWM görev saykılının hesaplanmasında ise formül 3.2 kullanılabilir.

*(CCPR1L:CCP1CON<5:4>) •Tosc • (TMR2 prescaler değeri)*

#### 1.3.11.1.3 PIC 16F877'nin PWM İşlemi İçin Kurulması

• PR2 kaydedicisi yazılarak PWM süreci başlatılır,

• CCPR1L kaydedicisine ve CCP1CON<5:4> bitlerine PWM görev saykıl'ı yazılır,

• TRISC<2> biti temizlenerek CCP1 pini çıkış olarak ayarlanır,

• TMR2 prescaler değeri ayarlanır ve T2CONOn biti vasıtasıyla Timer2 yetkilendirilir,

• CCP1 modülü PWM işlemi için düzenlenir.

### 1.3.12 ADC (Analog Dijital Konvertör) Modülü

PIC 16 F877'de 8 kanallı 10 bit'e kadar çevirme işlemi yapabilen bir analog-dijital çevirici (ADC) modülü bulunmaktadır.

PIC 16F877 üzerindeki ADC modülün çalışması şu şekildedir. Analog giriş örnekle ve tut kondansatörünü şarj eder. Örnekle ve tut kondansatörünün çıkışı dönüştürücünün girişine uygulanır. Dönüştürücü, ardışık yaklaştırma yoluyla bu analog düzeyin sayısal sonucunu üretir. Bu A/D dönüşümde, analog giriş sinyali 10 bitlik sayı karşılaştırma ile sonuçlanır. ADC eşsiz bir özelliğe sahiptir. İşlem yapmazken uyuma moduna geçer. Uyuma modunda ADC'nin saatinde bir iç RC osilatörü üretilmelidir. ADC Modül dört (4) kaydediciye sahiptir.

Bunlar;

1- A/D Yüksek sonuç kaydedicisi (ADRESH)

2- A/D Düşük sonuç kaydedicisi (ADRESL)

3- A/D Kontrol kaydedici 0 (ADCON0)

4- A/D Kontrol kaydedici 1 (ADCON1)

A/D çeviricinin kontrolünü ADCON0 ve ADCON1 kaydedicileri sağlamaktadır.

#### 1.3.12.1 ADCON0 Kaydedicisi

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | U-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADCS1 | ADCS0 | CHS2 | CHS1 | CHS0 | GO/DONE | — | ADON |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |

Şekil 1.17. ADCON0 Kaydedicisinin Bit Yerleşimi

Adcon,0(ADON): A/D çeviriciyi yetkilendirme bitidir. ADON=1 ise A/D çevirici açıktır ve işlem yapılabilir. durumdadır. ADON=0 ise A/D çevirici kapalıdır.

Adcon,2(GO/DONE): Eğer ADCON biti de 1 ise A/D çevirici statü biti görevini üstlenir. GO/DONE=1 ise A/D çevirici işlem yapıyor demektir. GO/DONE=0 ise A/D çevirici üzerinde herhangi bir işlem yapılmıyordur.

Adcon,3:5(CHS0:CHS2): A/D çevirici için kanal seçim bitlerini oluşturur. Bu bitlere verilecek değerlerle A/D çevirme için hangi kanalın seçileceği belirlenir. Daha önceden de belirtildiği gibi PIC 16F877'de A/D çevirici için 8 kanal mevcuttur.

000 = kanal 0, (RA0/AN0)

001 = kanal 1, (RA1/AN1)

010 = kanal 2, (RA2/AN2)

011 = kanal 3, (RA3/AN3)

100 = kanal 4, (RA5/AN4)

101 = kanal 5, (RE0/AN5)

110 = kanal 6, (RE1/AN6)

111 = kanal 7, (RE2/AN7)

Adcon,6:7(ADCS0:ADCS1): A/D çevirici için clock frekansı seçim bitleridir. Bu bitlere verilecek değerler ile A/D çevirme işlemi esnasında kullanılacak frekans değeri bize sunulan değerler içerisinden seçilir.

00 = FOSC/2

01 = FOSC/8

10 = FOSC/32

11 = FRC (Harici bir RC osilasyon kaynağından gelen clock darbeleri kullanılır.)

### 1.3.12.2 ADCON1 Kaydedicisi

| U-0 | U-0 | R/W-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADFM | — | — | — | PCFG3 | PCFG2 | PCFG1 | PCFG0 |
| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |

Şekil 1.18. ADCON1 Yazmacının Bit Yerleşimi

Adcon1,0:3(PCFG0:PCFG3): A/D çevirici portunun biçimini düzenlemeyi sağlayan bitlerdir. Yani A/D çevirme işleminde kullanılacak pinlerin nasıl davranacağını belirlememize olanak sağlarlar.

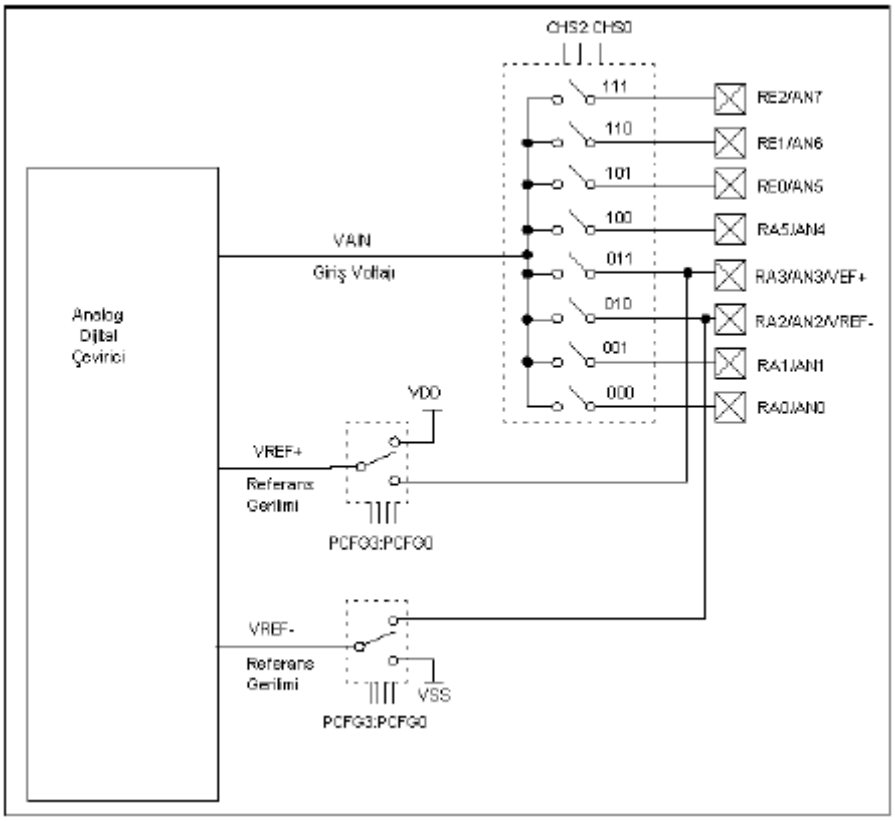
Tablo 1.5: PCFG3:PCFG0 Bitlerinin Aldığı Değere Göre Yaptığı İşlemler

Bu tabloda A=Analog giriş D=Dijital giriş/çıkış anlamına gelir.

| PCFG3: PCFG0 | AN7 RE2 | AN6 RE1 | AN5 RE0 | AN4 RA5 | AN3 RA3 | AN2 RA2 | AN1 RA1 | AN0 RA0 | VREF+ | VREF- | KANAL/ REF |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0000 | A | A | A | A | A | A | A | A | VDD | VSS | 8/0 |
| 0001 | A | A | A | A | VREF+ | A | A | A | RA3 | VSS | 7/1 |
| 0010 | D | D | D | A | A | A | A | A | VDD | VSS | 5/0 |
| 0111 | D | D | D | A | VREF+ | A | A | A | RA3 | VSS | 4/1 |
| 0100 | D | D | D | D | A | D | A | A | VDD | VSS | 3/0 |
| 0101 | D | D | D | D | VREF+ | D | A | A | RA3 | VSS | 2/1 |
| 011x | D | D | D | D | D | D | D | D | VDD | VSS | 0/0 |
| 1000 | A | A | A | A | VREF+ | VREF- | A | A | RA3 | RA2 | 6/2 |
| 1001 | D | D | A | A | A | A | A | A | VDD | VSS | 6/0 |
| 1010 | D | D | A | A | VREF+ | A | A | A | RA3 | VSS | 5/1 |
| 1011 | D | D | A | A | VREF+ | VREF- | A | A | RA3 | RA2 | 4/2 |
| 1100 | D | D | D | A | VREF+ | VREF- | A | A | RA3 | RA2 | 3/2 |
| 1101 | D | D | D | D | VREF+ | VREF- | A | A | RA3 | RA2 | 2/2 |
| 1110 | D | D | D | D | D | D | D | A | VDD | VSS | 1/0 |
| 1111 | D | D | D | D | VREF+ | VREF- | D | A | RA3 | RA2 | 1/2 |

Adcon1,7(ADFM): A/D çevirme işlemi esnasında meydana gelen verinin biçimini belirlemeye yarayan bittir. ADFM=1 ise ADRESH kaydedicisinin MSB kısmındaki altı biti 0 kabul edilir ve A/D çevirme sonucunda elde edilen veri ADRESH'ın 2 bitlik LSB kısmına ve ADRESL'ye yazılır. ADFM=0 ise ADRESL'nin Lsb kısmındaki 6 biti 0 kabul edilir ve A/D çevirme sonucu elde edilen veri ADRESL'nin son iki bitine ve ADRESH'a yazılır.

ADRESH: ADRESL kaydedicileri A/D dönüşümün 10 bit sonucunu kapsar. A/D dönüşümü bittiği zaman, sonuç A/D sonuç kaydedicisinin içine yüklenir. A/D modülü şekil 1.19'da görülmektedir.




Şekil 1.19. A/D Blok Diyagramı

A/D Modülü biçimlendirildikten sonra, dönüştürme işlemi başlamadan önce kanal seçilmiş olmalıdır. Analog giriş kanallarında ilgili TRIS bitleri giriş için seçilmiş olmalıdır. Aşağıdaki adımlar, A/D dönüşüm yapmak için takip edilmelidir.

1- A/D Modülü Konfigürasyonu

• Analog pinler, referans voltajları ve digital I/O konfigürasyonu (ADCON1)

• A/D giriş kanalı seçimi (ADCON0)

• A/D dönüşüm saat sekimi (ADCON0)

• A/D Modülünü açma

2- A/D Kesme Konfigürasyonu

• ADIF bitinin temizlenmesi

• ADIE bitinin ayarlanması

• GIE bitinin ayarlanması

3- Gerekli zamanı bekleme işlemi

4- Dönüşümün başlaması

• GO/DONE bitinin ayarlanması (ADCON0)

5- A/D dönüşümünün beklenmesi

6- A/D dönüşüm sonucunu okuma ve kaydetme

7- Diğer dönüşüm için 1. ve 2. kez adımları tekrarlama

### 1.3.12.3 A/D Girdileri İçin Gereksinimler

Belirlenmiş doğruluğu karşılaştırmak için A/D çeviricinin CHOLD kondansatörü giriş gerilimine şarj edilmelidir. CHOLD kondansatörü kaynak empedansı (RS) ve anahtar iç direnci (RSS) üzerinden şarj olur. Anahtar iç direnci, kaynak voltajının değerine göre değişir. Analog kaynaklar için tavsiye edilen maksimum empedans 10 KΩ dur. Dönüşüm yapılmaya başlamadan önce analog giriş kanalı seçilmiş olmalıdır.(6)

Minimum giriş zamanı hesabı;

TCAQ=TAMP+TC+TCOFF

TACQ=Minimum girdi zamanı

TAMP=Yükselteç yerleşme zamanı

TC= CHOLD şarj zamanı

TCOFF= Sıcaklık katsayısı

Örnek 3.9.

TACQ = TAMP + TC + TCOFF

TACQ = 2 µs + Tc + [(Temp – 25 oC)(0.05 µs/ oC)]

TC = -CHOLD (RIC + RSS + RS) ln(1/2047)

-120 pF (1 kΩ+ 7 kΩ+ 10 kΩ) ln(0.0004885)

-120 pF (18 kΩ) ln(0.0004885)

-2.16 µs (-7.6241)

16.47 µs

TACQ = 2 µs + 16.47 µs + [(50 oC – 25 oC)(0.05 s/ oC)]

18.447 µs + 1.25 µs

19.72 µs

### 1.3.12.4 A/D Dönüşüm Saatinin Seçimi

TAD bit başına A/D dönüşüm zamanı olarak tanımlanır. 10 bit A/D dönüşüm için maksimum 12 TAD gerekir. TAD seçimi için mümkün olan 4 seçenek vardır.

• 2TOSC
• 8TOSC
• 32TOSC
• Dahili RC Osilatörü

A/D Dönüşümün doğru olarak yapılması için , TAD minimum 1,6 μS seçilmiş olmalıdır.

### 1.3.12.5 Analog Port Pinlerini Yapılandırma

ADCON1 ve TRIS kaydedicileri kontrol ve işletim port pinleridir. Analog girişlerin TRU bitlerinin karşılıklı olarak ayarlanması gerekir. TRIS biti temizlenmiş ise dijital çıkış seviyesine ( VOH veya VOL ) dönüştürme yapılmış demektir.

### 1.3.12.6 A/D Dönüşümü

Örnek 3.10'de bir A/D dönüşümünün nasıl yapıldığı gösterilmektedir. Analog pinler, analog girdiler olarak yapılandırılır. Analog referans gerilimleri VDD ve VSS dir. A/D kesmesi seçildi ve A/D dönüşüm saat frekansı (FRS) ile sola yanaşık olarak sonuçlandırıldı. Dönüştürme RA/0/AN0 pinleri ile gerçekleşti.(6)

Örnek 3.10. A/D Dönüşümü

```
bsf  status, rp0        ; Bank 1
bcf  status, rp1        ;
clrf  adcon1            ; A/D girişleri kuruldu
bsf  pie1, adıe         ; A/D komutları etkin
bcf  status, rp0        ; Bank 0
movlw  11000001         ; RC saat, A/D açık, kanal 0 seçildi.
movwf  adcon0           ;
```

```
bcf  pir1, adif        ; A/D komut bayrak bitleri temizlendi
bsf  intcon, peie      ; çevresel komutlar etkin
bsf  intcon, gie       ; tüm komutlar etkin
bsf  adcon0, go        ; A/D dönüşümü başla
```

#### 1.3.12.7 A/D Dönüşümü Sırasında Uyuma

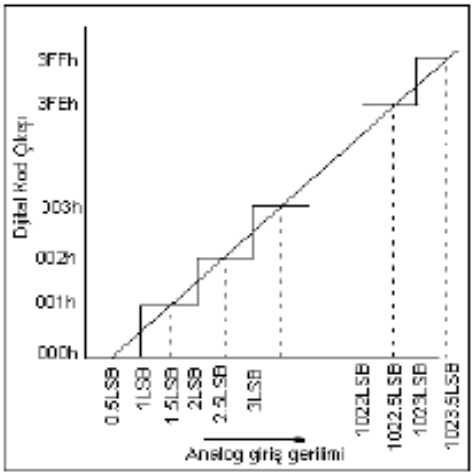
A/D Modül işletim sırasında uyuma modunda olacaktır. Bunun için A/D saat kaynağı ayarlanmalıdır. (ADCS1:ADCS0=11). RC saat kaynağı seçildiği zaman dönüştürme başlamadan önce A/D modül bir saat çevrimi süresi kadar bekler. Bu uyku talimatına izin verir. Dönüşümdeki tüm sayısal anahtarlama gürültüsü elemine edilmiş olur. Dönüştürme işlemi bittiği zaman GO/DONE biti temizlenir ve sonuç adres kaydedicisi içine yüklenir. A/D kesmesi etkinleştirilirse aygıt uykudan uyanır. A/D kesmesi pasifleştirilirse A/D modülü kapalı duruma dönse de bir süre açık kalacaktır.(6)

#### 1.3.12.8 A/D Doğruluk/Hata

Aygıt frekansı RC saatin alçak kullanımı olduğu sistemlerde tercih edilir. Yüksek frekans azaltılırken, TAD aygıt osilatörü türetilmelidir. A/D Dönüştürücü için belirtilen salt doğruluk, miktar ölçme hatası, integral hata, türevsel hata, tam skala hata, sapma hatalarının katkıları toplamını içerir. Herhangi bir kod için ideal geçişe karşı, güncel geçişten maksimum sapma olarak tanımlanır. Verilen bir analog giriş aralığı için sayısal çıkış kodu aynı olur. Bu sayısal kod analog girişten ölçülen miktar kadardır. Analogtan dijitale ölçme işleminde hata tipik olarak ½ LSB kadardır.

### 1.3.12.9 Transfer Fonksiyonu

A/D Dönüştürücünün transfer fonksiyonu aşağıda gösterildiği gibidir. Analog giriş voltajı/1024 ile bulunur.




Şekil 1.20. A/D Transfer fonksiyonu

### 1.3.13. Güç Sarfiyat Bilgileri

Tablo 1.6: EEPROM verisi ile uyumlaştırılan Kayıtlar/Bitler

| Adres | Ad | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | Power-on reset'inden sonraki değerler | Bütün diğer reset'lerden sonraki durumlar |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 08h | EEDATA | EEPROM VERİ KAYDI | | | | | | | | xxxx xxxx | uuuu uuuu |
| 09h | EEADR | EEPROM ADRES KAYDI | | | | | | | | xxxx xxxx | uuuu uuuu |
| 88h | EECON1 | --- | ------ | ----- | EEIF | WRERR | WREN | WR | RD | ---0 x000 | ---0 q000 |
| 89h | EECON2 | EEPROM KONTROL KAYDI | | | | | | | | -------- | -------- |

**Not:** EADRR <7:6> biti temizlenmelidir. Bu bitlerden herhangi birisi kurulduğunda micronun maximum IDD si her iki bitin de temizlenmiş olması halindekinden daha yüksektir. Spesifikasyon 400mA' dir. Silinen EADRR<7:6> ile maximum 150mA civarındadır. İşaretler: x =bilinmeyen, u = değişmeyen, ----- = '0' olarak tamamlanmamış okuma Q = Şartlara bağımlı değer. Bölgelendirilen hücreler EEPROM tarafından kullanılmamaktadır.

### 1.3.14. CPU' nun Spesifik Özellikleri

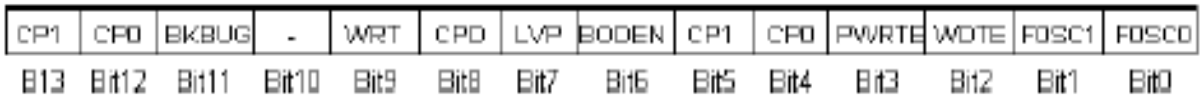
Mikrodenetleyici'yi diğer işlemcilerden ayıran şey , gerçek zaman uygulamalarının gereksinmeleri ile ilgili özel devreleridir. PIC16F877' te sistem güvenliğini maksimize eden, dış elemanları ayırarak maliyeti minimize eden , güç tasarrufu, çalışma modu ve kod koruma gibi özellikleri taşımaktadır. Bu özellikler;

- OSC seçimi
- Reset
    - Güç kaynağı reseti (POR)
    - Yüksek güç timerı (PWRT)
    - Osilatör başlangıç Timer ı (OST)
- Kesmeler
- Watchdog Timer
- Sleep
- Kod koruma
- ID yerleşimleri
- Devre içi seri programlama

PIC16F877' te yalnızca konfigrasyon bitleri tarafından kapatılabilen Watchdog Timer mevcuttur. Güvenliği arttırmak için bu kendi RC osilatörünü de çalıştırmaktadır. Yüksek güçte gereken esas gecikmeleri sağlayan 2 Timer mevcuttur. Bunlardan birisi Osilatör Başlangıç Timer 'ıdır. Bu timer , kristal osilatör durgunlaşıncaya kadar çipi resette tutar. Diğer timer ise yalnızca nominal yüksek güçte 72 ms sabit gecikme üreten Yüksek Güç Timer' ıdır. Bu güç kaynağı stabilize olurken aygıtı resette tutar. Bu iki çip üzeri Timer ile , uygulamaların çoğu hiçbir reset devrelerini gerektirmemektedir. SLEEP modu çok düşük enerjili alçak güç modunu sunmaktadır. Kullanıcı SLEEP ten çıkmak için dış reset, Watchdog Timer zaman aralığı veya kesmeleri kullanabilir. Bazı osilatör seçenekleri, kısımları uygulamaya yerleştirmek için elde edilmektedir. RC osilatör seçeneği sistem maliyetini, LP kristal seçeneği ise güç sarfiyatını düşürmektedir. Çeşitli seçenekleri seçmek için konfigürasyon bitler seti kullanılmaktadır. (1)

### 1.3.14.1. Biçimlendirme (Konfigürasyon) Bitleri

Biçimlendirme bitleri çeşitli aygıt işlevlerini seçmek için programlanabilir( ‘0’ olarak okur) yada programlamadan bırakılabilir. ( ‘1’ olarak okur) Bu bitler 2007h program bellek yerleşiminde saklanır. 2007h adresi kullanıcı program bellek biriminin ötesindedir ve özel test/biçim bellek birimine (2000h- 3FFFh) aittir. Bu birime yalnızca programlama sürecinde erişilebilirdir.




Şekil 1.21. Biçimlendirme bitlerinin bellekteki bit dizilimi

Bit 0,1 (FOSC0,FOSC1): Osilatör seçme bitleridir. Yapılacak uygulamada hangi tür osilatör kullanılacağını tayin eder. (RC,XT,HS,LP) FOSC0 ve FOSC1 bitlerinin aldığı durumlara göre kullanılacak osilatör türleri tablo 3.8’de verilmiştir.

Tablo 1.7. Osilatör seçim tablosu

| FOSC1 | FOSC0 | OSC |
|---|---|---|
| 1 | 1 | RC |
| 1 | 0 | HS |
| 0 | 1 | XT |
| 0 | 0 | LP |

Bit 2 (WDTE): Watchdog timer’in kontrol edildiği bittir. Bu bitin 1 olması WDT’ı devreye sokarken, 0 olması ile WDT devreden çıkarılır.

Bit 3(): Power up timer kontrol bitidir. Bu bitin 1 olması ile PWRT devreden çıkarılır, 0 olması ile ise PWRT devreye sokulur.

Bit 6 (BODEN): Brown out reset özelliğine ait yetkilendirme bitidir. BODEN=1 ise BOR kullanım dışı, BODEN=0 ise BOR etkindir.

Bit 7 (LVP): Düşük gerilim programlaması için yetkilendirme bitidir. LVP=1 ise RB3/PGM pinin PGM özelliği devreye girer. LVP=0 olduğunda ise RB3 özelliği kullanılabilir.

Bit 8(CPD): Bellek EEPROM verisi kod koruması için yetkilendirme bitidir. CPD=1 ise kod koruması devre dışıdır. CPD=0 ise kod koruma etkindir ve EEPROM içerisine kayıtlı veriye bir teşebbüste bulunulamaz.

Bit 9(WRT): Flash program belleğine yazma yetkilendirme bitidir. WRT=1 ise korumasız program belleğine EECON kontrolünde veri yazılabilir. WRT=0 ise korumasız program belleğine EECON kontrolünde veri yazılamaz.

Bit 11(DEBUG): Devre üzeri seri programlama yetkilendirme bitidir. DEBUG=1 ise RB6,RB7 normal G/Ç işlemlerinde kullanılır. DEBUG=0 ise RB6, RB7 aygıt içerisine program yerleştirme işleminde kullanılır.

Bit 12:13 (CP0:CP1): Yapılan uygulamanın kopyalamaya veya herhangi bir teşebbüse karşı kod koruma özelliğinin konması işlemini kontrol eden bitlerdir. Bu bitlerin aldığı farklı değerler ile farklı kod koruma seçenekleri söz konusu olmaktadır.

11 = Kod koruma kapalı

10 = 1F00h tan 1FFFh'a kadar kod koruması

01 = 1000h tan 1FFFh'a kadar kod koruması

00 = 0000h tan 1FFFh'a kadar kod koruması

#### 1.3.14.2. Osilatör Tipleri

PIC16F877 mikrodenetleyicilerinde 4 çeşit osilatör tipi bulunmaktadır. Kullanıcı bu 4 moddan birini seçerek iki biçimlendirme bitini (FOSC1 ve FOSC2) programlayabilir.

LP Kristal veya seramik rezonatör-asgari akım - 40Khz

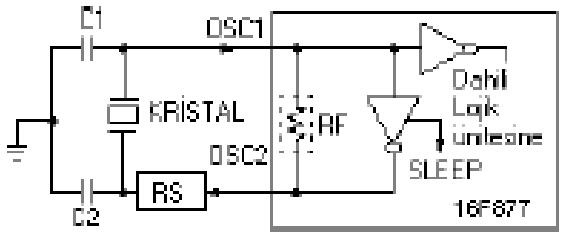
XT Kristal veya seramik rezonatör- genel amaçlı - 4Mhz

HS Kristal veya seramik rezonatör- yüksek hız - 20 Mhz

RC Direnç/ Kapasitör zaman sabitli - düşük maliyet - 4Mhz

### 1.3.14.2.1. Kristal Osilatör / Seramik Rezonatör

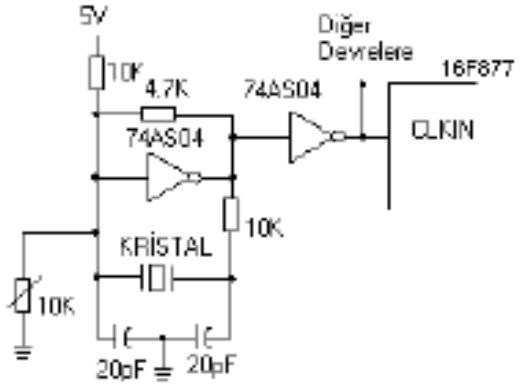
XT, LP ve HS modları, kristal veya seramik rezonatörlerin, OSC1/CLKIN ve SC2/CLKOUT pinlerine bağlanmalarıyla kurulur.PIC16F84' te osilatör dizaynı paralel kesim kristali kullanmayı gerektirir. Kesim kristallerinin seri bağlanması ile, kristallerin üzerindeki frekans değerlerinden farklı bir frekans değeri oluşabilir. XT, LP ve HS modlarında OSC1/CLK1 dışardan sürülebilir.




Şekil 1.22. PIC 16F877'de Kristal Osilatör Kullanımı

### 1.3.14.2.2. Harici Kristal Osilatör Devresi

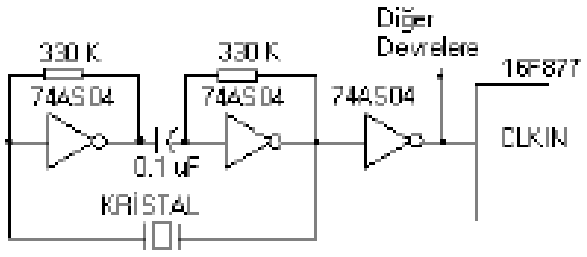
Ambalaj öncesi osilatör TTL girişli basit osilatör devresi kurulabilir. Ambalaj öncesi osilatör geniş işlem alanı ve denge sunmaktadır. İyi tasarlanmış kristal osilatör TTL girişleri ile iyi performans sağlayacaktır. İki tip kristal osilatör devresi mevcuttur. Birisi seri rezonanslı ve diğeri de paralel rezonanslı osilatördür.




Şekil 1.23.Paralel Rezonanslı Osilatör Devresi

Şekil 1.23 Paralel resonanslı osilatör devresini göstermektedir. Devre, kristalin temel frekansını kullanmak için tasarlanmıştır. 74AS04 Inverter, paralel osilatörün gerektirdiği 180 dereceli faz kaymasını yürütmektedir. 4.7 KΩ direnci kararlılık için negatif geri besleme sağlamaktadır. 10 KΩ potasiyometre 74AS04'ü lineer bölgede çalıştırır. Bu devre osilatör tasarımı için de kullanılabilir.

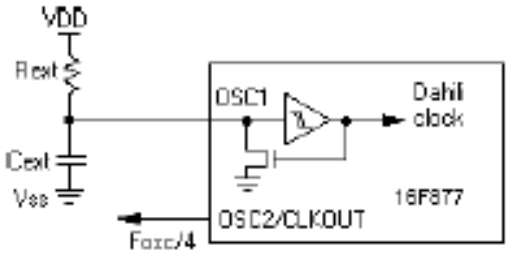
Şekil 1.24 ise seri rezonanslı osilatör devresini göstermektedir. Bu devre kristalin esas frekansını kullanmak için tasarlanmıştır. Inverter 180o faz kaymasını yürütmektedir. 330 KΩ lık direnç, Inverter'lere kendi lineer bölgesinde etkilenmesi için negatif geri beslemeyi sağlamaktadır.




Şekil 1.24. Seri Rezonanslı Osilatör Devresi

**1.3.14.2.3 RC Osilatörü**

Zamanlamaya duyarsız uygulamalar için RC osilatörü fazla maliyeti azaltmaktadır. RC osilatör frekansı, voltaj ihtiyacına, direnç (Rext) değerine, kondansatör (Cext) değerine, çalışma ısı derecesinin değerine bağlıdır. Bunun ötesinde ambalaj tipindeki şekil kapasitansındaki farklılıkları da, özellikle düşük Cext değerlerinde, osilatörün frekansını etkileyebilmektedir. Kullanıcı dış R ve C elemanlarının toleransı nedeniyle meydana gelen değişiklikleri de dikkate almaktadır. Şekil 1.25 de RC kombinasyonunun PIC16F877'ye nasıl bağlandığı görülmektedir.




Şekil 1.25. RC Osilatör Bağlantısı

Rext < 2.2KΩ için, osilatör işlemi kararsız hale gelebilir hatta tamamıyla durabilir. Çok yüksek Rext değerleri için (yani 1 MΩ gibi) osilatör, gürültüye, rutubete ve sızmalara karşı duyarlı hale gelir. Bunun için Rext değeri 3Ω ile 100Ω arasında tutulmalıdır.

Osilatörün hiçbir dış kondansatörü olmaksızın çalışmasına rağmen (Cext=0pf) , gürültü ve kararsızlık nedenleri ile 20pf'ın üzerindeki değerlerin kullanılması yerinde olur. Çok düşük kapasitans veya hiç kapasitans olmadan, osilatör frekansı , devredeki kaçak kapasitans gibi dış kapasitanslardaki değişmeler nedeniyle önemli ölçüde değişebilmektedir.

Frekastaki değişme ;yüksek R değerleri (çünkü gerilim sızıntı değişmeleri R den daha büyük değerde RC frekansını etkileyecektir.) ve düşük C değerlerinin etkisi ile artar. (Çünkü giriş kondansatörü değişmeleri RC frekansında daha büyük etkiye sahiptir ).

OSC2/CLKOUT pininde 4 ile bölünen ösilatör frekansı mevcuttur ve bu frekans, sürücüleri test etmek için yada diğer lojik üniteleri test etmek için kullanılabilir.

## 1.4 PIC Assembly ve Pıc 16f877 Programlama Temelleri

### 1.4.1.Assembler ve PIC Assembly

#### 1.4.1.1 Assembler

Bir text editöründe assembly kurallarına göre yazılmış olan komutları PIC'in anlayabileceği hexadecimal kodlara çeviren (derleyen) bir programdır. Microchip firmasının hazırladığı MPASM bu işi yapan assembler programıdır. Assembler'e çoğu zaman compiler (derleyici) de denir.

#### 1.4.1.2 PIC Assembler

Assembly dili, bir PIC'e yaptırılması istenen işlerin belli kurallara göre yazılmış komutlar dizisidir. Assembly dili komutları İngilizce dilindeki bazı kısaltmalardan meydana gelir. Bu kısaltmalar genellikle bir komutun çalıştırılmasını ifade eden cümlenin baş harflerinden oluşur. Böylece elde edilen komut, bellekte tutulması kolay (mnemonic) bir hale getirilmiştir. Örneğin:BTFSC (**B**it **T**est **S**kip if **C**lear): İlgili biti test et, eğer sıfırsa bir sonraki komutu atla, anlamında kullanılan İngilizce cümlenin kısaltmasıdır.

### 1.4.2 PIC Assembly Dili Yazım Kuralları

PIC assembly programlarının bilgisayar ortamında yazılabilmesi için notepad, edit gibi yazım aracılarına ihtiyaç vardır. Microchip firmasının PIC uygulamaları için özel olarak hazırlamış oduğu MPLAB programını kullanmamız halinde bu gibi yazım aracılarına ihtiyaç duyulmaz. Çünkü MPLAB içerisinde assemply PIC assembly dilinde program yazmak için text tabanlı bir yazım editörü ve ayrıca MPASM mevcuttur.

MPASM assembler programının yazılan komutları doğrun olarak algılayıp, PIC'in anlayabileceği hexadecimal kodlara dönüştürebilöesi için şu bilgilerin program içerisinde özel formatta yazılması gerekir.

• Komutların hangi PIC'e ait olduğu (PIC 16F877,PIC 16F84, PIC 16C71...vb)

• Programın bellekteki hangi adresten başlayacağı.
• Komutların ve etiketlerin neler olduğu
• Programın bitiş yeri

Örnek 4.1.

PIC 16F877'de PORTC'nin 4 bit LSB bitlerini giriş olarak, 4 bit MSB bitlerini çıkış olarak tanıtan ve PORTC'nin 2. Bitinin 1 olması durumunda PORTC nin 6. Bitini 1 yapan programın yazımı:

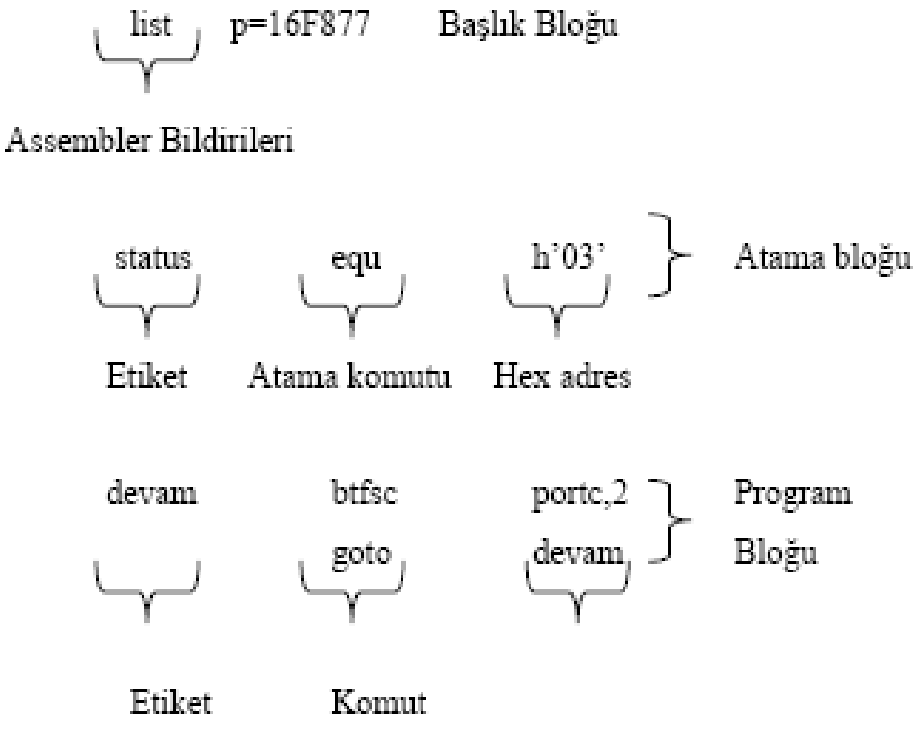
```
;-------------Örnek Program 1-----------
list p=16f877 ;kullanılacak PIC in seçimi
;---------------------------------------------
;Adres tanımlama bloğu
status equ h'03'
portc equ h'07'
trisc equ '87'
;---------------------------------------------
org h'00' ;programı 00 adresinden başlat
;---------------------------------------------
;PORTC nin G/Ç tanımlamasının yapılması
clrf portc
bsf status,5 ;bank1 seçildi
movlw h'0F'
movwf trisc
bcf status,5 ;bank0 seçildi
;---------------------------------------------
;Program bloğu
basla bcf portc,6
btfss portc,2
goto basla
bsf portc,2
devam btfsc portc,2
```

goto devam
goto basla
end ;sonlandırma

### 1.4.2.1 Program Yazımında Noktalı Virgül (;) Kullanımı

Baş tarafında (;) bulunan satırlar, assembler tarafından hexadecimal kodlara dönüştürülmez. Bu satırlar programın geliştirilmesi esnasında hatırlatıcı açıklamaların yazılmasında kullanılır. Ayrıca program bölümlerni birbirinden ayırmak için (------- veya =======) çizgileri kullanmak, programı görsel olarak daha okunur halegetirdiği gibi bu çizgiler arasına uyarılar ve açıklamalar da yazılabilir. Bu sayede programda bir hata veya sorun meydana geldiğinde takip işlemi daha rahat yapılır.

### 1.4.2.2 Bir Program Satırının Kısımları



#### 1.4.2.2.1 Etiket

PIC belleğindeki bir adresin atandığı, hatırlamayı kolaylaştıran kısaltmalardan meydana gelen sembolik işaretlere etiket denir. Örneğin portc etiketi, PIC 16F877'nin kaydedici dosyası belleğindeki C portunun bulunduğu adresi temsil eden etikettir. Etiketler program içerisinde 1. kolana yazılır.

Portc equ h'07' yazıldıktan sonra C portunun hangi adreste olduğunu akılda tutmaya gerek yoktur. Programın herhangi bir yerinde portc etiketi kullanıldığında,C portunun adresi olan h'07' yazılmış gibi işlem görülür.

Birinci kolona yazılan ve adres atanmayan etiketler de kullanılabilir. Örneğin basla ve devam bu tip etiketlerdendir. Bu etiketler program akışını istenilen yere dallandırmak için kullanılırlar. Bu tip etiketlerin adresi özel kaydedici adresi gibi fiziksel bir adres değildir. Bu şekilde tanımlanan bir etikete assembler otomatik olarak adres verir. Bu adresi programcının bilmesi gerekmez.

Etiket tanımlarken uyulması gereken kurallar şunlardır.

• Etiketler birinci kolona yazılmalıdır.
• Etiketler bir harfle veya alt çizgi ( _ ) ile başlamalıdır.
• Etiketler içerisinde Türkçe karakterler kullanılamaz.
• Etiketler bir asembly komutundan oluşamaz.
• Etiketlerin içerisinde alt çizgi, rakam, soru işareti bulunabilir.
• Etiketler en fazla 31 karakter uzunluğunda olabilir.
• Etiketlerde büyük/küçük harf duyarlılığı vardır. ("Devam" diye tanımlanmış bir etiketi program içerisinde "devam" yazarak kullanmak mümkün değildir.)

#### 1.4.2.2.2 Atama Deyimi (EQU)

EQU deyimi PIC 16F877'nin belleğindeki bir hexadecimal adresi belirlenen bir etikete atamak için kullanılır. Aşağıda bu atama deyimine birkaç örnek gösterilmiştir.

```
portb equ h'06'
say1 equ h'10'
portc equ h'07'
```

#### 1.4.2.2.3 Sabitler

PIC assembly dilinde hexadecimal,binary ve decimal sayılar birer sabittir. Sabitler movlw, sumlw, andlw gibi bazı komutların içerisinde ve atama işlemlerinde kullanılırlar. Sabitler program içerisinde kullanılırken hexadecimal olduuna dair "h" veya 0x.., binary olduğuna dair "b", decimal olduğuna dair "d" karakterleri ile birlikte kullanılmalıdırlar.

```
movlw b'01000111' ;binary sabit
sublw d'126' ;decimal sabit
sublw h'0F' ;hexadecimal sabit
movlw 0x0F ;hexadecimal sabit
```

#### 1.4.2.2.4 ORG Deyimi

ORG İngilizcedeki "origin" kelimesinden gelmektedir. ORG deyimi program içerisinde iki amaç için kullanılır.

• Program komutlarının hangi adresten itibaren başladığını gösterir.

```
org h'00'
```

• PIC 16F877'nin interrupt (kesme) alt programlarının başlangıç adresini belirlemede kullanılır.

```
org h'04'
```

### 1.4.2.3 PIC Assembly Komutlarının Yazılışı

PIC 16f877'nin toplam 35 tane komutu vardır. Bu komutların yazılış biçimini dört grupta toplayabiliriz.

• Byte yönlendirmeli komutlar
• Bit yönlendirmeli komutlar
• Sabit işleyen komutlar
• Kontrol komutları

Komutların yazılış biçimlerini açıklarken bazı tanımlama harfleri kullanacağız. Öncelikle bu harflerin anlamlarını vermekte fayda vardır.

f=File register (Kaydedici)

d=destination (gönderilen yer)

d=0→w kaydedicisi (akümülatör)

d=1→f kaydedicisi

k=sabit veya adres etiketi

b=bit tanımlayıcı

b=binary sayıları belirleyen harf

d=decimal sayıları belirten harf

h=hexadecimal sayıları belirten harf

#### 1.4.2.3.1 Byte Yönlendirmeli Komutlar

f,d→f: Hexadecimal adres veya kaydedici adı

Komut d:Komutun çalıştırılmasından sonra verinin yazılacağı yer.

d=0→w kaydedicisi

d=1→f kaydedicisi

Örnek 4.2.

movf h'03',0 ;h'03' adresindeki kaydedicinin içeriğini w kaydedicisi içerisine kopyalar movf status,0 ;STATUS kaydedicisinin içeriğini w kaydedicisine kopyalar. movf status,1 ;STATUS kaydedicisinin içeriğini yine STATUS kaydedicisine kopyalar.

**Not:**Bit yönlendirmeli komutlarda destination (gönderilecek yer) belirleyen d'nin yazıldığı yere 0 veya 1 yazmak hatırlatıcı olmayabilir. MPASM bunu dikkate alarak 0 yerine w, 1 yerine f yazmaya izin verir. MPASM'nin MS-DOS versiyonunda ise w ve f harflerinin otomatik olarak kullanılmasına izin verilmez. Bu durumda her programın tanımlama bölümüne aşağıdaki eşitlikler yazılmalıdır.

```
w  equ  0
f  equ  1
```

Bu eşitliklerden sonra komutlarda destination belirlemek için w ve f harfleri kullanılabilir. Örneğin:

```
decfsz  say1,f
movf  say2,w
```

#### 1.4.2.3.2 Bit Yönlendirmeli Komutlar

f,b→f: Hexadecimal adres veya kaydedici adı

Komut b:0-7 arasında hexadecimal sayı veya etiket (EQU komutu ile adresi tanımlanmış olmalıdır)

Örnek 4.3

```
bcf  h'03',5     ;h'03' adresindeki verinin 5. Bitini sıfırla
bsf  porta,2     ;PORTA'nın 2. bitini birle
```

#### 1.4.2.3.3 Sabit İşleyen Komutlar

Komut

k→k: sabit (b'00110011', h'0F', d'255' gibi)

Örnek 4.4.

```
movlw  h'2f'          ;w kaydedicisine 2F hexadecimal sayısını yükler
addlw  d'221'         ;w kaydedicisindeki sayıya 221 decimal sayısını ekler.
```

#### 1.4.2.3.4 Kontrol Komutları

Komut

k→k: Adres etiketi

Örnek 4.5.

```
goto  dongu           ;programı dongu ile belirtilen yere dallandır.
call  timer           ,program akışı timer etiketi ile belirtilen alt programa dallanır
```

### 1.4.2.4 Program Bölümleri

Bu kısımda programın metin belgesi olarak nasıl yazıldığı konusuna değinilecektir. Yazılan programı temel olarak 5 kısımda toplayabiliriz.

- PIC serisinin tanıtılması ve kaynak dosyanın belirtilmesi
- Kaydedici tanımlamalarının yapılması

• Bit tanımlamalarının yapılması
• Giriş ve çıkış pinlerinin belirlenmesi
• PIC'in yapacağı işlemlerin yazılması (Brogram bloğu)

#### 1.4.2.4.1 PIC Serisinin Tanıtılması Ve Kaynak Dosyanın Belirtilmesi

Piyasada değişik isimlerle değişik işlemleri gerçekleştiren PIC'ler satılmaktadır. Her bir PIC'in değişik yapı ve tanımlamaları vardır. Bu tanımlamaların kayıtları ise PIC'in özellikleri göz önünde tutularak yazılmış kaynak dosyalarda tutulur. Ancak her bir seriye ait özellikleri tek tek bilme imkanımız çok azdır. Bu nedenle PIC üreten firmalar üretilen elemana ait özellik ve kaynak dosyaları hazır halde bize sunmaktadır. Bizler bu kaynak dosyalardan kullanacağımız PIC'e ait olanını alabilir ve kendimize göre değişiklikler yaparak kullanabiliriz. Ancak yapılacak değişiklikler standart özellikleri etkilememelidir. Bir kaynak dosyada temel olarak standart register tanımlamaları (W, F, STATUS, EEDATA, EEADR,PORTA,PORTB….gibi), standart bit tanımlamaları (WREN ZERO, CARRY….gibi) ve var ise kendimize ait macrolar bulunmaktadır.

Her program metninin baş tarafında yer alması gereken seri tanıtımı ve kaynak dosya belirtme işlemi aşağıdaki şekillerde gerçekleştirilebilir.

```
list p=16f877
#include <16f877.ınc>
```

Yukarıdaki tanımlamanın ilk satırında kullanılacak PIC'in serisi belirtilmiştir.2. satırda ise program metninin bulunduğu klasör içerisinde aranması istenen kaynak dosya (INCLUDE dosyası) belirtilmiştir.

#### 1.4.2.4.2 Kaydedici Tanımlamalarının Yapılması

Yukarıda bize sunulan kaynak dosyaların içerisinde standart kaydedicilerin tanımlandığından söz etmiştik. Ancak çoğu çalışmada tanımlanan bu kaydediciler yetersiz kalmaktadır ve yeni RAM bölgeinden kaydediciler tanımlanması gerekmektedir. Bu durumda kullanılmak istenen kaydedici bellek bölgesinin adresi ile eşleştirilip bir isim verilerek program içerisinde belirtilebilir ve ilerleyen işlemlerde rahatlıkla kullanılabilir. Ancak bu işlem gerçekleştirilirken kaynak dosya içerisinde tanımlanan standart adresleri bilmek ve tanımlayacağımız kaydedicilerin adresleini bu adreslerin dışında seçmemiz gerekmektedir. PIC 16F877 kaynak dosyası içerisinde yer alan standart kaydediciler aşağıdaki gibidir.

```
indf  equ h'0000'
tmr0  equ h'0001'
pcl  equ h'0002'
status  equ h'0003'
fsr  equ h'0004'
porta  equ h'0005'
portb  equ h'0006'
portc  equ h'0007'
portd  equ h'0008'
porte  equ h'0009'
pclath  equ h'000a'
intcon  equ h'000b'
pir1  equ h'000c'
pir2  equ h'000d'
tmr1l  equ h'000e'
tmr1h  equ h'000f'
t1con  equ h'0010'
tmr2  equ h'0011'
t2con  equ h'0012'
sspbuf  equ h'0013'
sspcon  equ h'0014'
```

```
ccpr1l  equ h'0015'
ccpr1h  equ h'0016'
ccp1con  equ h'0017'
rcsta  equ h'0018'
txreg  equ h'0019'
rcreg  equ h'001a'
ccpr2l  equ h'001b'
ccpr2h  equ h'001c'
ccp2con  equ h'001d'
adresh  equ h'001e'
adcon0  equ h'001f'
option_reg  equ h'0081'
trisa  equ h'0085'
trisb  equ h'0086'
trisc  equ h'0087'
trisd  equ h'0088'
trise  equ h'0089'
pie1 equ h'008c'
pie2  equ h'008d'
pcon  equ h'008e'
sspcon2  equ h'0091'
pr2  equ h'0092'
sspadd  equ h'0093'
sspstat  equ h'0094'
txsta  equ h'0098'
spbrg  equ h'0099'
adresl  equ h'009e'
adcon1  equ h'009f'
eedata  equ h'010c'
eeadr  equ h'010d'
eedath  equ h'010e'
```

```
eeadrh  equ h'010f'
eecon1  equ h'018c'
eecon2  equ h'018d'
```

#### 1.4.2.4.3 Bit Tanımlamalarının Yapılması

Program içerisinde programdaki işlemleri daha rahat anlamak ve yazım esnasında hatayı azaltmak için çıkış pinlerine ve entegre içerisinde kontrolü sağlayacak bitlere kolay anlaşılabilecek isimler verilebilir. İsimlendirilecek bu bitlerin ise programda belirlenmesi gerekir. Yani verilecek ismin hangi kaydedicinin hangi bitine ait olduğu programa yazılmalıdır. Bu kısımda yine kaynak dosyada kullanılan bazı standart bitler karşımıza çıkar. Bitleri isimlendirir iken bu standart bitleri göz önünde bulundurmak gerekir. Standart olarak tanımlanan bitlerin bazıları aşağıda sunulmuştur.

```
;----- status bitleri --------------------------------------------------
#define irp status,7
#define rp1 status,6
#define rp0 status,5
#define not_to status,4
#define not_pd status,3
#define z status,2
#define dc status,1
c status,0
;----- intcon bitleri --------------------------------------------------
#define gie intcon,7
#define peie intcon,6
#define t0ie intcon,5
#define inte intcon,4
#define rbie intcon,3
#define t0if intcon,2
#define intf intcon,1
#define rbif intcon,0
```

```
;----- adcon0 bitleri ----------------------------------------------------
#define adcs1 adcon0,7
#define adcs0 adcon0,6
#define chs2 adcon0,5
#define chs1 adcon0,4
#define chs0 adcon0,3
#define go adcon0,2
#define not_done adcon0,2
#define go_done adcon0,2
#define chs3 adcon0,1
#define adon adcon0,0
```

Bu standart bit tanımlamaları yanında standart kaydedicilerin bazılarının ve kişisel kaydedicilerin bitlerine özel isimler verme olanağımız da vardır. Örnek aşağıda sunulmuştur.

```
#define tus porta,2
#define clock portb,1
#define kontrol say1,7
#define en reg1,0
```

#### 1.4.2.4.4 Giriş ve Çıkış Tanımlamaları

PIC 16F877 giriş ve çıkışın yapıldığı pinler PORTA, PORTB, PORTC, PORTD, PORTE pinleridir. Bu portlara ait tüm pinleri hem giriş hem de çıkış olarak kullanabiliriz. Ancak program içerisinde pinlerin hangisini giriş için hangisini çıkış için kullandığımızı belirtmek gerekir. Çıkış için tanımlanan bir pine gerilim girişi yapmak bu pinin bozulmasına neden olabilir. Bu yüzden işlemlerde kullanılmayan pinleri programda giriş olarak tanımlamak faydalı olmaktadır. Bir program içerisinde yapılabilecek giriş ve çıkış tanımlamaları aşağıda örnekler halinde sunulmuştur.

```
bank1
movlw b'00011100'
movwf trisa
movlw b'01100110'
```

movwf trisb
bank0

#### 1.4.2.4.5 PIC'in Yapacağı İşlemlerin Yazılması

PIC'in yapacağı işlemleri yazmak demek komutları bir algoritma veya akış şemasına göre programda yerlerine yazmak demektir. (Bu kısımda yapılacak işlemler sonraki kısımda örnek ve uygulamalar ile genişçe anlatılacaktır.)

#### 1.4.2.4.6 Neden PLC değilde PIC

Yapacağımız uygulamalar için istediğimiz özelliklere sahip PIC leri bulabiliriz. Kullanmamız gereken pek çok devreden veya fazladan elemandan bizi kurtarır. Mesela PIC12F675 isimli mikrokontrolör sadece 8 bacaklıdır ve 4 adet 10 bitlik analog dijital dönüştürücüye sahiptir. Bunun yanında sahip olduğu kesmeler ve timerlar sayesinde ufak uygulamalarımızda faydalı olmaktadırlar. Mesela bu tezde kullanılan ve çekilen akıma göre devreyi çalıştıran veya korumaya alan devremizde bu mikrokontrolör kullanılmıştır. Hepimizin aklına "Neden PIC kullanıldı da PLC kullanılmadı?" sorusu gelmiştir. Bunun cevabı yukarıda anlatıldığı gibi farklı PIC tiplerinin her türlü ihtiyaca cevap verebilmesidir. Bunun yanında da ucuzluğu, kolay temin edilebilmesi gibi pek çok yönden bu tez için tercih etmemizi sağlamıştır.

### 1.4.3 PIC 16F877 Komut Takımı

PIC 16F877'nin programlanmasında kullanılan 35 komutluk komut takımı tablo 1.8' de verilmiştir.

Tablo 1.8. PIC 16F877 Komut Takımı

| Komut | | Süresi(T) | Etkilenen Bit |
|---|---|---|---|
| ADDWF | f,d | 1 | C,DC,Z |
| ANDWF | f,d | 1 | Z |
| CLRF | f | 1 | Z |
| CLRW | - | 1 | Z |
| COMF | f,d | 1 | Z |
| DECF | f,d | 1 | Z |
| DECFSZ | f,d | 1(2) | |
| INCF | f,d | 1 | Z |
| INCFSZ | f,d | 1(2) | |
| IORWF | f,d | 1 | Z |
| MOVF | f,d | 1 | Z |
| MOVWF | f | 1 | |
| NOP | - | 1 | |
| RLF | f,d | 1 | C |
| RRF | f,d | 1 | C |
| SUBWF | f,d | 1 | C,DC,Z |
| SWAPF | f,d | 1 | |
| XORWF | f,d | 1 | Z |
| BCF | f,b | 1 | |
| BSF | f,b | 1 | |
| BTFSC | f,b | 1(2) | |
| BTFSS | f,b | 1(2) | |
| ADDLW | k | 1 | C,DC,Z |
| ANDLW | k | 1 | Z |
| CALL | k | 2 | |
| CLRWDT | - | 1 | TO,PD |
| GOTO | k | 2 | |
| IORLW | k | 1 | Z |
| MOVLW | k | 1 | |
| RETFIE | - | 2 | |
| RETLW | k | 2 | |
| RETURN | - | 2 | |
| SLEEP | - | 1 | TO,PD |
| SUBLW | k | 1 | C,DC,Z |
| XORLW | k | 1 | Z |

### 1.4.4 PIC Assembly Ne Zaman Vazgeçilmezdir?

Assembly için söylenmiş eski bir söz vardır. Assembly kullanmak için 3 neden vardır Hız hız ve daha çok hız. Bu söz her şeyi açıklayabilmektedir. Assembly ve bir alt dil olan PIC Assembly diğer üst düzey programlama dillerinden daha hızlıdırlar. MHz'ler mertebesindeki frekanslı işlemcilerin bile hayal olduğu zamanlarda Assembly kesinlikle tek seçenek idi. Örneğin C programlama dilinde en basit programda bile stdio.h kütüphane dosyasını yazdığınız programın içine eklemeniz gerekmektedir. Bu da tabi ki programa yük getirmektedir ve programın boyutu büyümektedir. Assembly dilinde böyle bir kütüphane dosyası olmadığından, yazılan her satır kodun tamamı ile gerekli olduğundan yazıldığından, Assembler kod dosyaları daha küçük boyutlu olmaktadırlar. Hızlarının sırrı buradan gelmektedir. küçük olan program kısa sürede yüklenir ve işleme geçilebilir.

Günümüzde en basit işlemci veya denetleyicilerin bile hızları 20 MHz düzeyine ulaşmıştır. Ayrıca komut işleme süreleri gelişen elektronik bilimi ile nano saniye gibi çok kısa zaman birimlerine düşmüştür. Yüksek düzey dillerde yazılan kodların fazladan getirdiği baytlar artık işlemcilere ve yük olmamaktadır. Bu nedenle birçok uygulamada Assembly' den daha yüksek düzeyde bir dil kullanmak zaman bakımından tasarımcıyı rahatlatmaktadır. Fakat çok kısıtlı hafıza birimleri ile çalışırken veya son derece yüksek hızlarda çalışılması gerektiğinde Assembly dilitercih edilebilir.

Ayrıca Assembly dili ile, üst düzey programlama dilleri ile çok zor yazılabilecek veya kesinlikle yazılamayacak kodlar yazma imkanı bulunmaktadır. Kısaca diğer dillere göre daha geniş ufka sahip bir programlama dilidir. Bu durumu C programlama dili ve diğer üst düzey diller, Assembly kodlarını kodun içine kabul etme yöntemiyle kendi lehlerine çevirmeye çalışmışlardır. Örneğin C ile program yazarken, birkaç satır Assembly kodu da gerek duyulursa araya eklenebilir ve sonra tekrar C koduna devam edilebilir. Bu durum yukarıda da belirtildiği gibi, yüksek düzey dillerle yazılması imkansız durumlarda ve çok hassas hız gereksinimlerinin olduğu durumlarda günümüzde kullanılmaktadır.

### 1.5 PIC BASIC Nedir?

PIC BASIC; üst düzey programlama dili olan BASIC ‘in, bir denetleyici olan PIC için düzenlenmiş sürümüdür. Bu programlama dili kullanılarak, PIC çok daha kolay ve hızlı biçimde istenilen işlevleri yerine getirebilmektedir. Deniz suyu termometresi projesinde tercih edilmesinin sebepleri; kolay kullanılabilir olması ve C programlama diline benzerliğidir.

#### 1.5.1 PIC BASIC Komutları

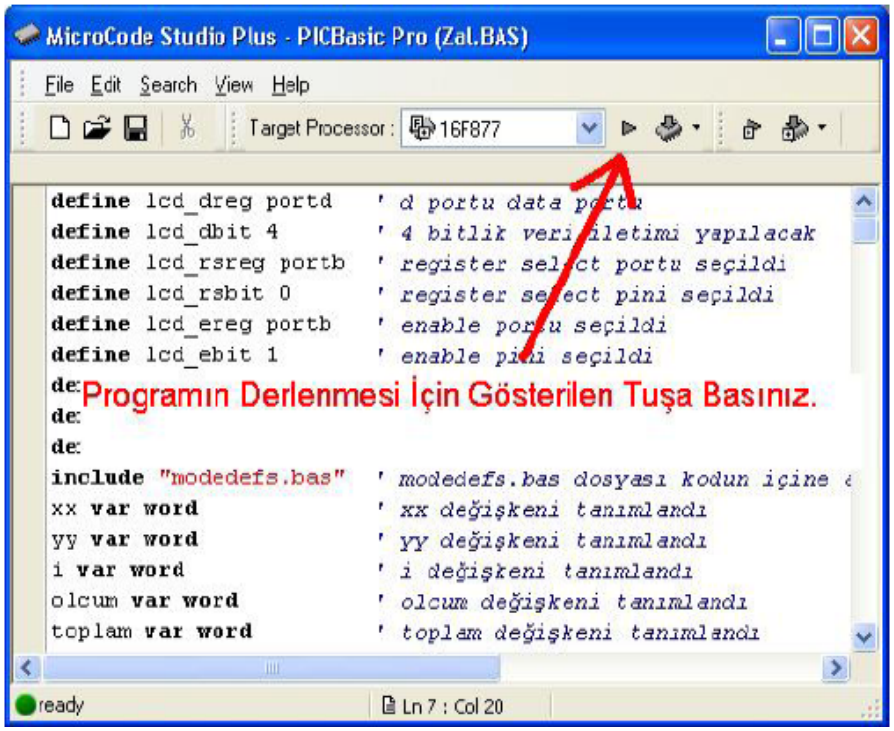
PIC BASIC 65 adet komut içermektedir. Bu 65 komut her türlü programlama ihtiyacına cevap verebilmektedir. Ayrıca Assembly kodu kullanılmak istendiğinde ASM … ENDASM komutlarının arasına yazarak Assembly kodları da BASIC kodlarıyla birlikte kullanılabilmektedir. Komutlar Tablo 1.9’ te toplu halde verilmiştir. Komutların teker teker açıklanması ve peşlerine gelebilen değişkenler, raporun arka kapağına iliştirilmiş olan CD’deki elektronik kitaplarda mevcuttur.

Tablo 1.9**.** PIC BASIC Komutları

| | | | | |
|---|---|---|---|---|
| @ | ASM..ENDASM | ADCIN | BRANCH | BRANCHL |
| BUTTON | CALL | CLEAR | CLEARWDT | COUNT |
| DATA | DTMFOUT | EEPROM | END | FREQOUT |
| FOR-NEXT | GOSUB | GOTO | HIGH | HSERIN |
| HPWM | HSEROUT | I2CREAD | I2CWRITE | INPUT |
| IF-THEN-ELSE | LCDOUT | LCDIN | LET | LOOKDOWN |
| LOOKDOWN2 | LOOKUP | LOOKUP2 | LOW | NAP |
| OUTPUT | OWIN | OWOUT | PAUSE | PAUSEUS |
| POT | PULSIN | PULSOUT | PWM | RANDOM |
| RCTIME | READ | READCODE | RETURN | REVERSE |
| SELECT-CASE | SERIN | SERIN2 | SEROUT | SEROUT2 |
| SHIFTIN | SHIFTOUT | SLEEP | SOUND | STOP |
| SWAP | TOGGLE | WRITE | WRITECODE | WHILE-WEND |

### 1.5.2 PIC BASIC Derleyicisi : Microcode Studio

Bir programlama dili olan PIC BASIC kullanılarak, gerekli algoritma ile birlikte yazılan kod mutlaka mikrodenetleyicinin anlayacağı hale getirilmelidir. Mikrodenetleyici, kısaca HEX kodu dediğimiz on altılık tabandaki rakam ve harflerden oluşan bir sisteme ait kodları içine alıp, o kodların istediği şekilde çalışmasını düzenlemektedir. PIC BASIC kodlarının HEX kodlarına çevrilmesi Microcode Studio adındaki yazılımla mümkün olabilmektedir. Şekil 1.26 de görüldüğü gibi uygun PIC modeli seçildikten ve uygun kodu yazdıktan sonra, okla gösterilen tuşa basmak, HEX dosyasını oluşturmak için yeterli olacaktır. Microcode Studio HEX kodundan başka Assembler kodunu da, aynı dizine aktarmaktadır.




Şekil 1.26. Microcode Studio HEX Dosyası Oluşturuyor

### 1.6 PIC Programının Tasarımı

Hazırlanan kodlar açıklamaları ile birlikte ilgili kartların bilgilerinin bulunduğu bölümünlerin sonlarında verilmiştir. Ayrıca Microcode Studio yazılımı ile oluşturulan Hex kodu verilmemiştir. Rapora ait CD'de ise programlama için kullanılan yazılımlar ve tüm kodlar dosyalar halinde verilmiştir.

### 1.7 PIC İçine Program Yüklenmesi ve IC-Prog Yazılımı

#### 1.7.1 IC-Prog Yazılımının Tanıtımı

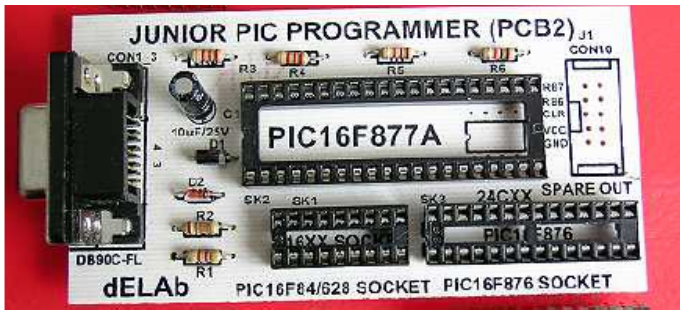
IC-Prog; çeşitli modellerdeki mikrodenetleyicilere ve mikro işlemcilere hexadecimal program kodunu yüklemeye yarayan bir yükleme yazılımıdır. 12Cxx, 16Cxxx, 16Fxx, 16F87x, 18Fxxx, 16F7x, 24Cxx, 93Cxx, 90Sxxx, 59Cxx, 89Cx051, 89S53, 250x0, 80C51 mikrodenetleyici, EEPROM ve mikro işlemci serilerini destekleyen IC-Prog, aynı zamanda hexadecimal kodun assembly koduna çevrilmesi işlemini de başarıyla gerçekleştirebilmektedir. Programın yükleneceği elektronik devre ile bağlantı bilgisayarın COM veya LPT portlarından sağlanabilmektedir. Yazılım, en az 8 mb RAM ve 386 işlemcili Windows işletim sistemi yüklü bir bilgisayara ihtiyaç duymaktadır. Yazılımın tasarımcıya sağladığı yararlara ve kullanım sırasında dikkat edilmesi gereken hususlara ileriki bölümlerde değinilmiştir.

#### 1.7.2 IC-Prog Yazılımının Kullanımı

##### 1.7.2.1 Bağlantı Portu Seçimi

Şekil 4.1 de görülen devre, PIC programlama devresidir. Şekilde devrenin üstünde de görüldüğü gibi, bu kart PIC16F877, PIC16F876, PIC16F84 ve PIC16F628 tipi mikrodenetleyicileri programlayabilmektedir. Ayrıca 24Cxx tipi EEPROM'ları da programlayabilmektedir. Proje boyunca PIC programlamada bu kart kullanılmıştır. Mikrodenetleyici programlama devresine takıldıktan sonra, programlama kartının, IC-Prog

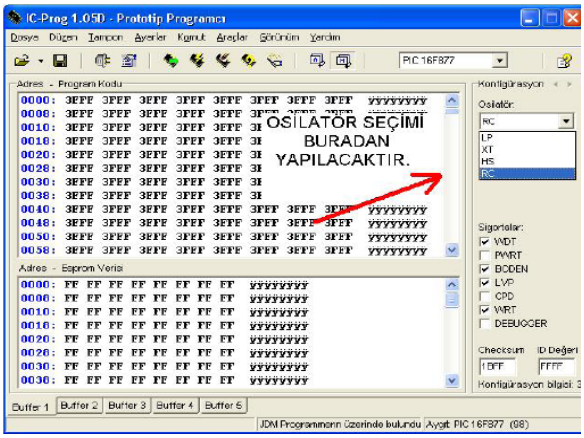
programının yüklü olduğu bir bilgisayarın COM1 veya COM2 portuna bağlanması gerekmektedir. Şekil 1.26‘ da de görüldüğü gibi programlama kartının bağlantı ucu DB-9 dişi bağlantı ucudur ve bilgisayarların COM portuna uyumludur. Programlama kartının bilgisayarın hangi portuna takıldığı IC-Prog yazılımına bildirilmelidir. Bu bildirimin nasıl yapılacağı da Şekil 1.27 de gösterilmiştir.




Şekil 1.27. PIC Programlama Kartı

#### 1.7.2.2.1 Osilatör Tipinin Seçilmesi ve Osilatör Çeşitleri

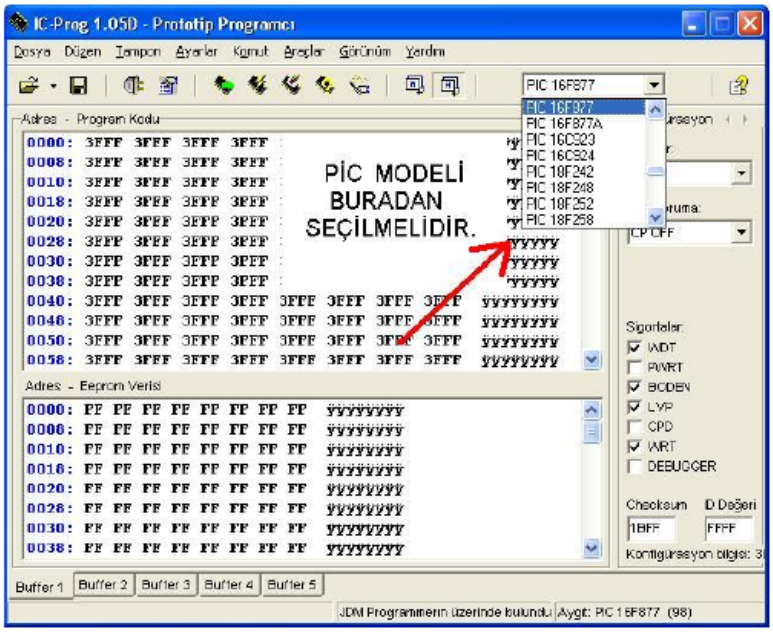
Osilatörler mikrodenetleyicilerin çalışma hızını belirleyen saat işareti üreteçleridir. Raporun 2. bölümünde osilatörler hakkında bilgi verilmiştir. Devre fiziksel olarak gerçeklenirken seçilen osilatör, yazılıma bildirilmelidir. Aksi takdirde PIC yüklenen programı düzgün çalıştırmayacaktır. 4 MHz Kristal osilatör XT; direnç-kapasite osilatör RC, 4MHz üstü osilatör HS, düşük frekanslı osilatör LP ifade etmektedir.




Şekil 1.28. Osilatör Seçimi

### 1.7.2.3 PIC Modelinin Seçilmesi

PIC modelinin doğru seçilmesi, programlamanın başarılı olması açısından hayati önem taşımaktadır. Programlama devresi ile bilgisayara bağlanan PIC modeli ile IC-Prog yazılımına bildirilen model farklı olursa; mikrodenetleyici kullanılmaz hale gelebilir ve doğru çalışmaz. Şekil 1.29'te PIC modelinin nasıl seçileceği gösterilmiştir.



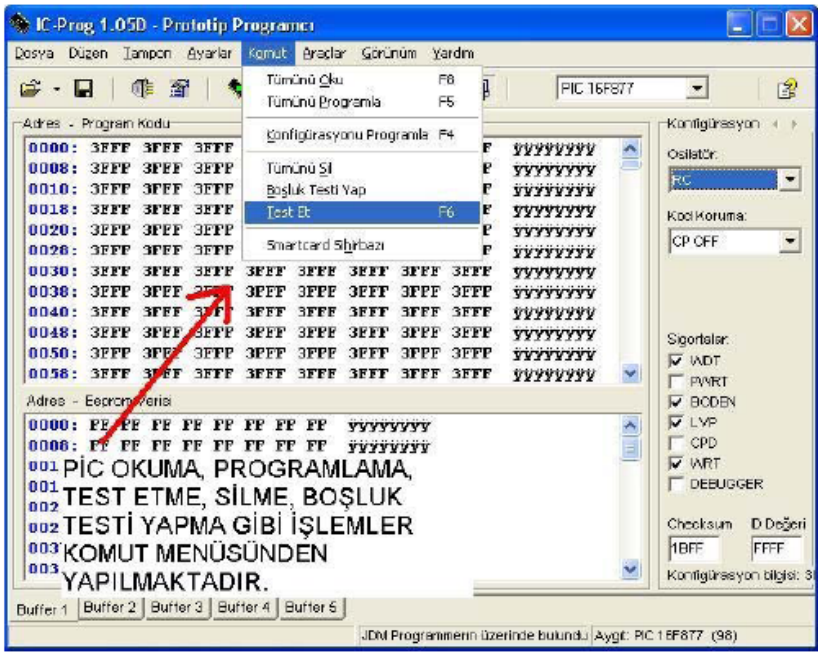

Şekil 1.29. PIC Modelinin Seçilmesi

### 1.7.3 Hex Dosyasının PIC'e Yüklenmesi ve Uyarılar

HEX dosyasının PIC'e yüklenmesi son aşamadır. Dosya menüsünden Dosya Aç seçilir ve elimizdeki HEX dosyası açılır. Dosya açıldıktan sonra PIC programlama devresi, bilgisayarın COM 1 portuna takılır. Yukarıda anlatıldığı gibi uygun osilatör ve PIC seçilmiş ise F5 tuşuna basmak yeterli olacaktır. Böylece HEX dosyası PIC 'e yüklenmeye başlayacaktır. Yükleme bitince ve "Aygıt doğrulaması başarılı" mesajı alındıktan sonra programlama devresi bilgisayardan dikkatle sökülerek, PIC tasarlanan devreye yerleştirilir.

Aygıt doğrulaması başarılı mesajından başka uyarı içeren bir mesaj ile karşılaşılması durumunda problem var demektir. İlk kontrol edilecek nokta, programlama devresi ile

bilgisayarın COM1 portu arasındaki bağlantıdır. Bu bağlantı tam olarak sağlanmış olmalıdır, temassızlık olmamalıdır. Ayrıca PIC, programlama devresine monte edilirken de son derece dikkatli olunmalıdır. Herhangi bir bacağın tam yerleşmemesi durumunda programlama gerçekleşemeyecektir. Eğer bütün bu kontroller sonunda her şeyin doğru yapıldığı düşünülüyor ise PIC tümdevresi test edilmelidir. Programlama devresi, bilgisayarın COM1 portuna bağlı durumda iken klavyeden F6 tuşuna basmak test başlangıcı için yeterli olacaktır.




Şekil 1.30. PIC'in Test Edilmesi

• IC-Prog yazılımını kullanabilmek için PIC, programlama devresine takılarak bilgisayarın com veya LPT portuna bağlanmalıdır. Hangi porta bağlanmış ise, yazılıma bu bildirilmelidir. Com portu kullanım kolaylığı nedeniyle tavsiye edilmektedir.

• Mikrodenetleyicilere program yazdırabilmek için mutlaka .HEX uzantılı dosyaya ihtiyaç duyulmaktadır. BAS, ASM, C gibi diğer uzantılı dosyalar mikrodenetleyicilere yüklenememektedirler. Yüklenecek olan dosyanın seçimi, dosya menüsünden dosya aç seçilerek yapılabilmektedir.

• Tasarlanan devrede kullanılan osilatör tipi ( RC, LP, XT, HS ) mutlaka konfigürasyon menüsünden doğru seçilmelidir. Aksi takdirde programlama doğru sonuçlar verememektedir.

• Bilgisayara programlama devresi ile takılan mikrodenetleyicinin tüm bacaklarının yerine oturmuş olduğuna çok dikkat edilmelidir. Herhangi bir topraklama bacağındaki temassızlık devrenin yanmasına sebebiyet verebilir.

• IC-Prog, mikrodenetleyicinin kullanılabilir olup olmadığını da kontrol edebilmektedir. Bu özelliğin kullanımı için, programlama kartı ile birlikte tümdevre, bilgisayarın uygun portuna bağlandıktan sonra klavyeden F6 tuşuna basmak yeterli olacaktır. Programlama sürerken, programlama kartını bilgisayardan çekmek çok risklidir ve mikrodenetleyici kullanılmaz hale gelebilir.

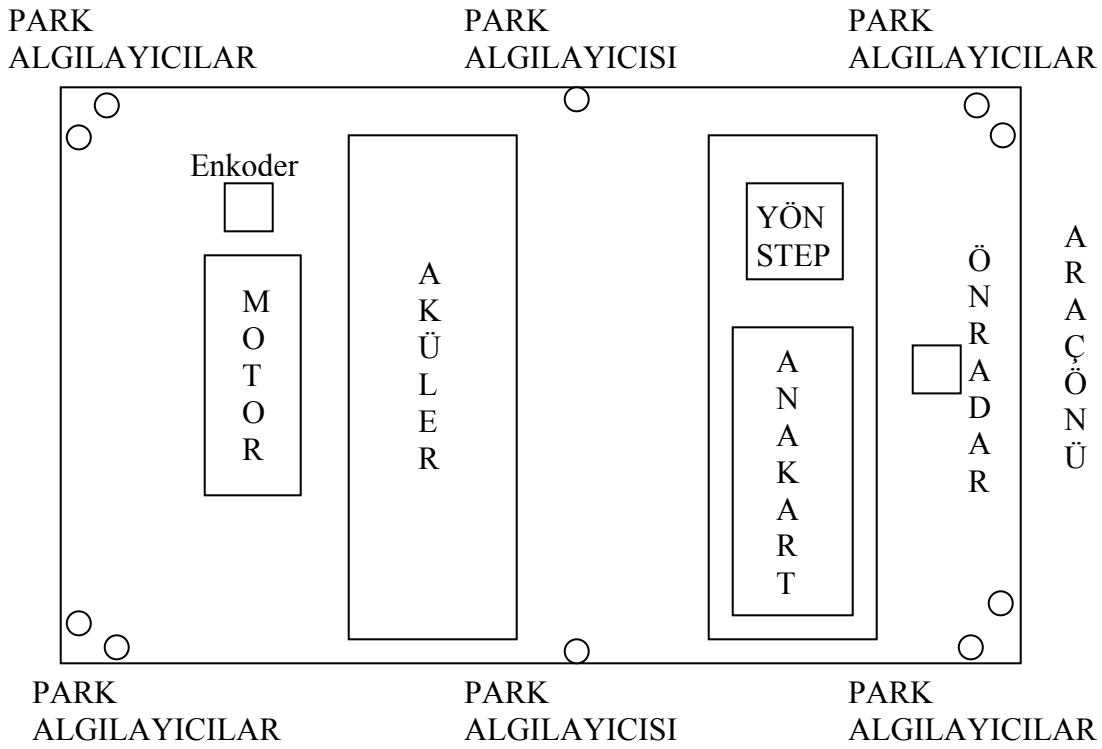
• Bu yazılım http://www.ic-prog.com/index1.htm web adresinden ücretsiz temin edilebilmektedir. Bütün tez boyunca programın kullanılan sürümü 1.05D sürümüdür. Ayrıca IC-Prog ile ilgili tüm kaynaklar ve programın 1.05D sürümü ekteki CD'de yer almaktadır.

## 2. YAPILAN ÇALIŞMA VE BULGULAR

### 2.1. Materyal

Çalışmada, verilen rotada hareket edecek bir araç için sürücü kartı tasarlanmıştır. Aracın kontrolü yine bu araç için tasarlanmış bir anakart ile sağlanmaktadır. Ana kart, hazırda çalışan ultrasonic radar sensöründen ve enkoderden aldığı verileri değerlendirmekte, aracın kalkış ve duruşunu bu şekilde sağlamaktadır. Araçta kullanılması muhtemel olan park sensörleri de tasarlanıp ana karta girişleri sağlanmıştır. Ancak bu proje için park sensörleri kısmı kullanılmamıştır. Böylece bu projenin geliştirilmesi için faydalı bir kart ortaya çıkarılmıştır.

### 2.2. Araç, Kartlar ve Sürüş Yardımcı Elemanları (algılayıcılar) Yerleşim Planı




Şekil 2.1. Aracın Şematik Görünüşü

### 2.2.1. Araç Kısımları

#### 2.2.1.1. Motor

Araç, 24 volt doğru gerilimle çalışan, 0,75 kW gücünde, 30 Amper akım çeken, 1000 rpm devir yapan Femsan marka bir motora sahiptir. Bir dişli sistemi ile momenti tekerleklere aktarır. Motor sürücü devresindeki mikrokontrolörden elde edilen Pwm ile motor kendi hız limitleri dahilinde ileri ve geri yönlerde sürülebilmektedir.


Şekil 2.2. Araç Motorunun Görünüşü

#### 2.2.1.2. Ana Kart

Aracın hareketini kontrol etmek için tasarlanmıştır. Tüm sensörler ana karta bilgi iletirler. Böylece ana kart gelen bilgilere göre aracı, verilen rotaya ve yol durumuna uygun olarak sürecek komutları motor sürücü kartına gönderebilmektedir. Ayrıca eklencek bir step motor sürücü için kontrol uçları bırakılmış ve bu sayede aracın dönüş hareketi de yine bu kart üzerinden yapılabilecektir.

### 2.2.1.3. Enkoder

Motorun hızını okuyabilmemiz için kullanılan enkoder her turunda 50 kare dalga üretmektedir. Her 50 kare dalga tekerleğin bir turunu ifade etmektedir. Tekerlek çevresi, mikrokontrolör tarafından 50 ye bölünerek her kare dalga için gidilen mesafe elde edilmektedir. Enkoderin ürettiği kare dalgalar ana kart tarafından yol ve hız verisine dönüştürülmektedirler. Böylece aracın hızı ve aldığı yol verisi ana mikrokontrolöre iletilmektedir.


Şekil 2.3. Enkoderin Görünüşü

### 2.2.1.4. Akü Grubu

Motor ve kartlar için gerekli olan enerjiyi elde edebilmek için 2 adet 12 volt doğru gerilim çıkışı veren 65 A lik aküler kullanılmıştır. Aküler seri bağlanmak suretiyle motorun ihtiyacı olan 24 Volt gerilim elde edilir. Ayrıca aracı süren kartların enerjisi de farklı gerilim çıkışlarını elde etmek için tasarlanmış ana besleme kartı sayesinde elde edilir.


Şekil 2.4. Kullanılan akünün görünüşü

### 2.2.1.5. Ultrasonik Radar

Ultrasonik radar araç ileri hareket halinde iken aracın önünü tarar, 0 – 2,55 metre arasındaki mesafede kalan cisimlerin uzaklığını ana karta iletir. Ana kart, eğer aracın hareketini engelleyecek bir durum varsa motor sürücü kartına yeni hız verisini iletir. Tabii ki bu devrede bu duruş hızı kontrol edilmemiştir. Sadece hız sıfıra indirilmektedir. Sebebi de aracımızın bir mekanik veya elektriksel frene sahip olmamasıdır.

#### 2.2.1.6. Park Algılayıcılar

Araca 0 - 18 - 36 cm yakınlıktaki cisimleri algılamasını sağlamak için tasarlanmış ve aracın üzerine monte edilmiştir. Bu kart sonraki kontrol uygulamalarında faydalı olabileceği düşüncesi ile tasarlanmıtır. Park algılayıcılar olarak adlandırılmasının sebebi park anındaki yaklaşma mesafelerini ayarlamak için kullanılmasından kaynaklanmaktadır. Ancak sürüş halinde de aracın yanlardan çarpmasını engellemek için kullanılır. On adet park algılayıcısı bulunmaktadır. Ayrıca anakart üzerinde yedek 2 adet olmak üzere 12 adet park sensörü girişi mevcuttur. Bu algılayıcılar, eğer bir cisim algılanırsa durumu ana karttaki park algılama kontrolörüne bildirir. Harekete mani bir yakınlık algılandıysa ana kart tasarıma uygun olarak bir sonuç çıkarabilecek ve bunu uygulayacaktır.

#### 2.2.1.7. Yön Adım Motoru

Aracın dönüşlerini sağlamak için ön tekerlekler bir direksiyon kutusuna bağlanmıştır. Bu direksiyon kutusunun mili, redüktörlü bir adım motoru tarafından döndürülmekte ve böylece hareketin yönlendirilmesi ana kart tarafından yapılabilmektedir. Yön adım motoru sürücüsü, ana karttan aldığı dönüş hareketini gerçekleştirmek için tasarlanmıştır. Ancak bu uygulamada yukarıda da belirtildiği gibi kullanılmamıştır.

## 2.3. Tasarlanan Kartlar

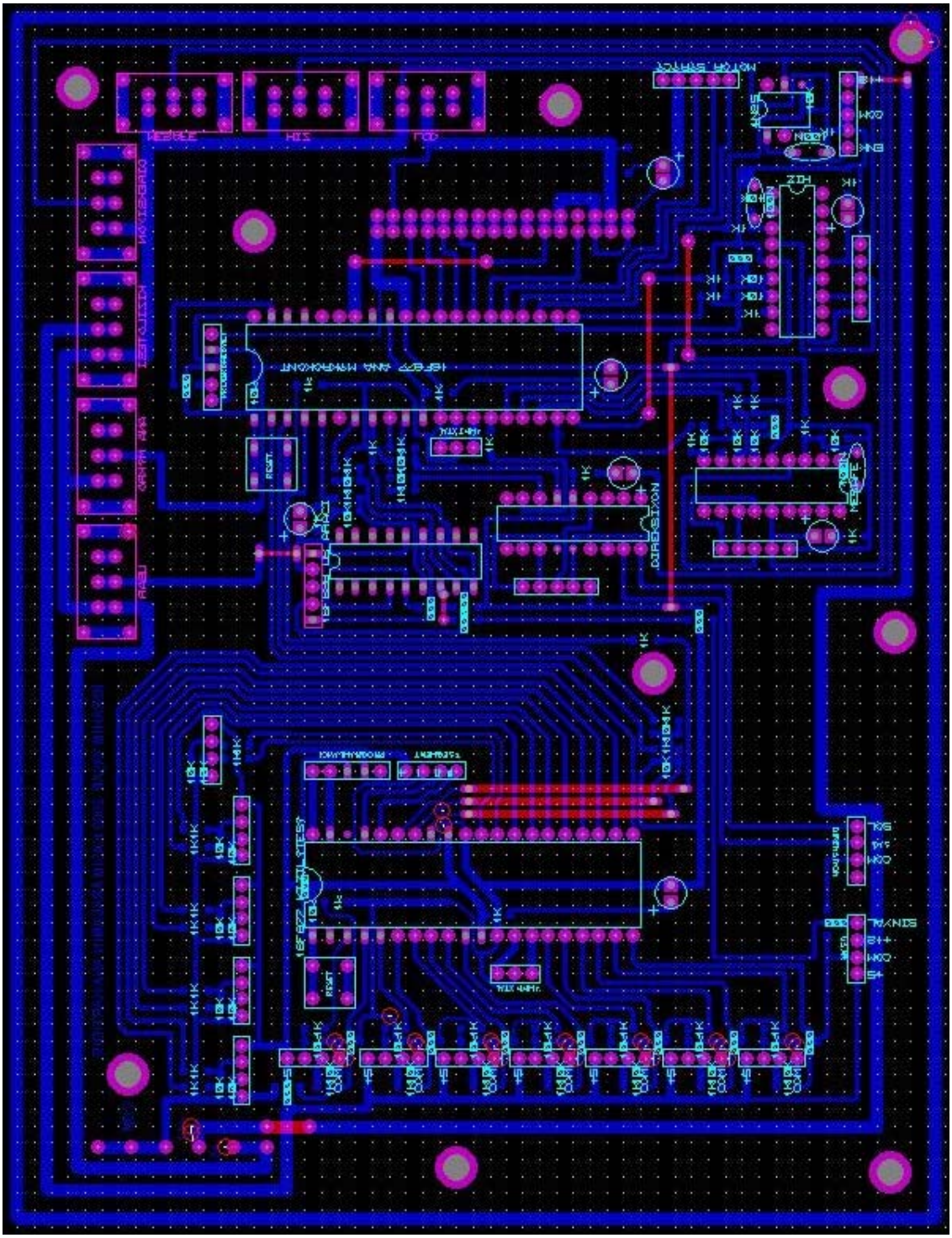
### 2.3.1. Ana Kart

Sistemin tüm verilerinin toplandığı ve yapılması gereken tüm talimatların gönderildiği kısımdır. Üzerinde çevresel elemanlardan gelen veriler toplanır. Toplanan veriler ilgili mikrokontrolörlerde işlendikten sonra ana mikrokontrolöre iletilir. Ana mikro kontrolörde durum değerlendirmesi yaparak aracın hareketlerini tayin eder. Ana kart üzerinde ana kart mikrokontrolörü, kızılötesi toplayıcı, ultrasonik aracı, direksiyon, mesafe ve hız mikrokontrolörü olmak üzere toplam 6 adet mikrokontrolör bulunmaktadır. Ana mikrokontrolör ve kızılötesi toplayıcı PIC16F877A, ultrasonik aracı, direksiyon, mesafe ve hız mikrokontrolörleri ise PIC16F628 dir. Bu mikrokontrolörler birbirleri ile 3 er pin aracılığı ile haberleşmektedir. Birinci pin verici için kesme gönderir, kesme gelince alıcı kesme alındı veriyi göndermeye başla sinyali gönderir ve verici sinyalini diğer bir pin üzerinden göndermeye başlar. Veri tamamen gönderilince alıcı teyit sinyalini keser ve verici de kesmeyi iptal eder. Böylece veri problemsiz bir şekilde gönderilmiş olur.

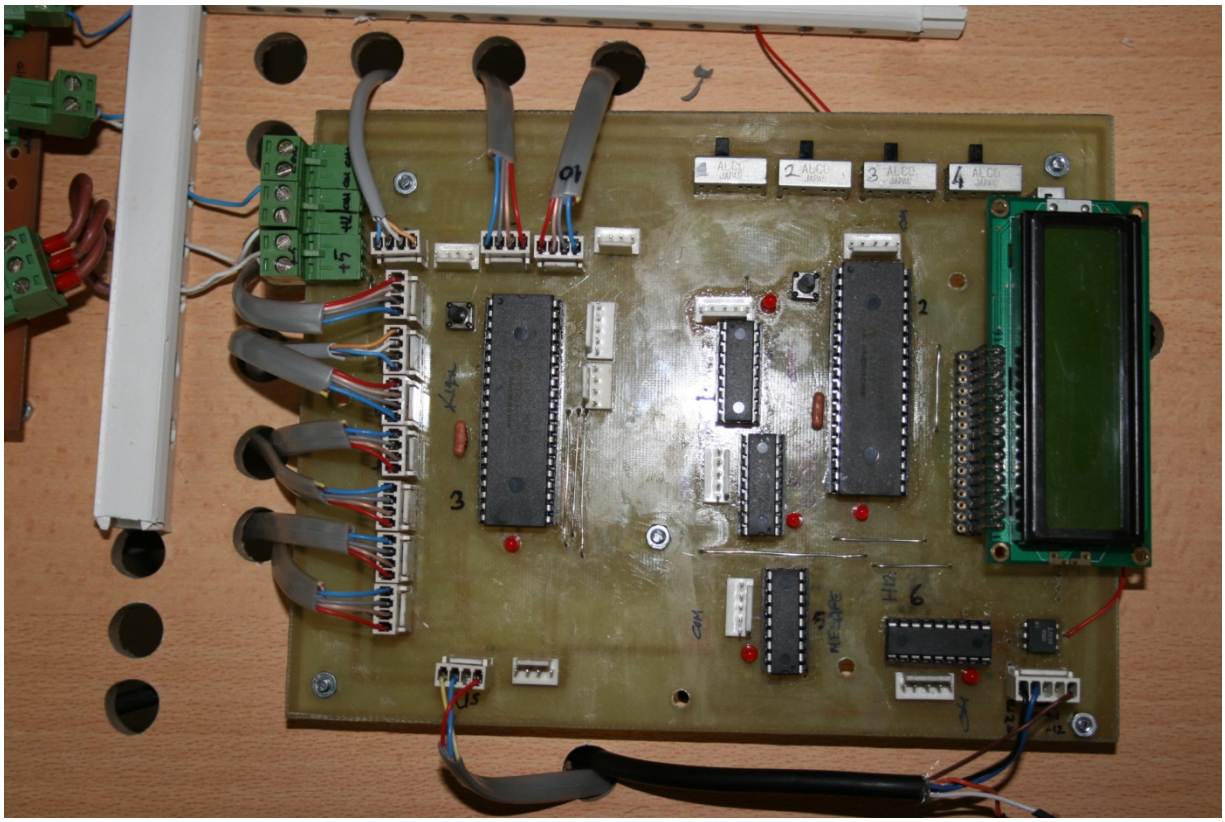
Ana kart üzerindeki her bir mikrokontrolör tek tek ana kart üzerinde programlanabilmektedir. Bu da program kodunun yazımı sırasında devamlı olarak mikrokontrolörleri çıkartmaktan bizi kurtarır. Ayrıca ana kartta her bir mikro kontrolör için birer tane anahtar konulmuştur ve bu anahtarlar istediğimiz mikrokontrolörü kapatıp açabilmemizi sağlamaktadır.

Yine ana kart üzerinde bulunan bir adet lcd ekran da istediğimiz verileri burada görüntülememizi sağlamak maksadı ile konulmuştur.

Ana kart üzerindeki mikrokontrolörlere ait anakartın ares çizimi ve devrenin üstten görünüşü aşağıda verilmiştir. Anakart üzerindeki mikrokontrolörlere ait program kodları Ek 1’ de verilmiştir.



Şekil 2.5. Anakart devresinin ares çizimi

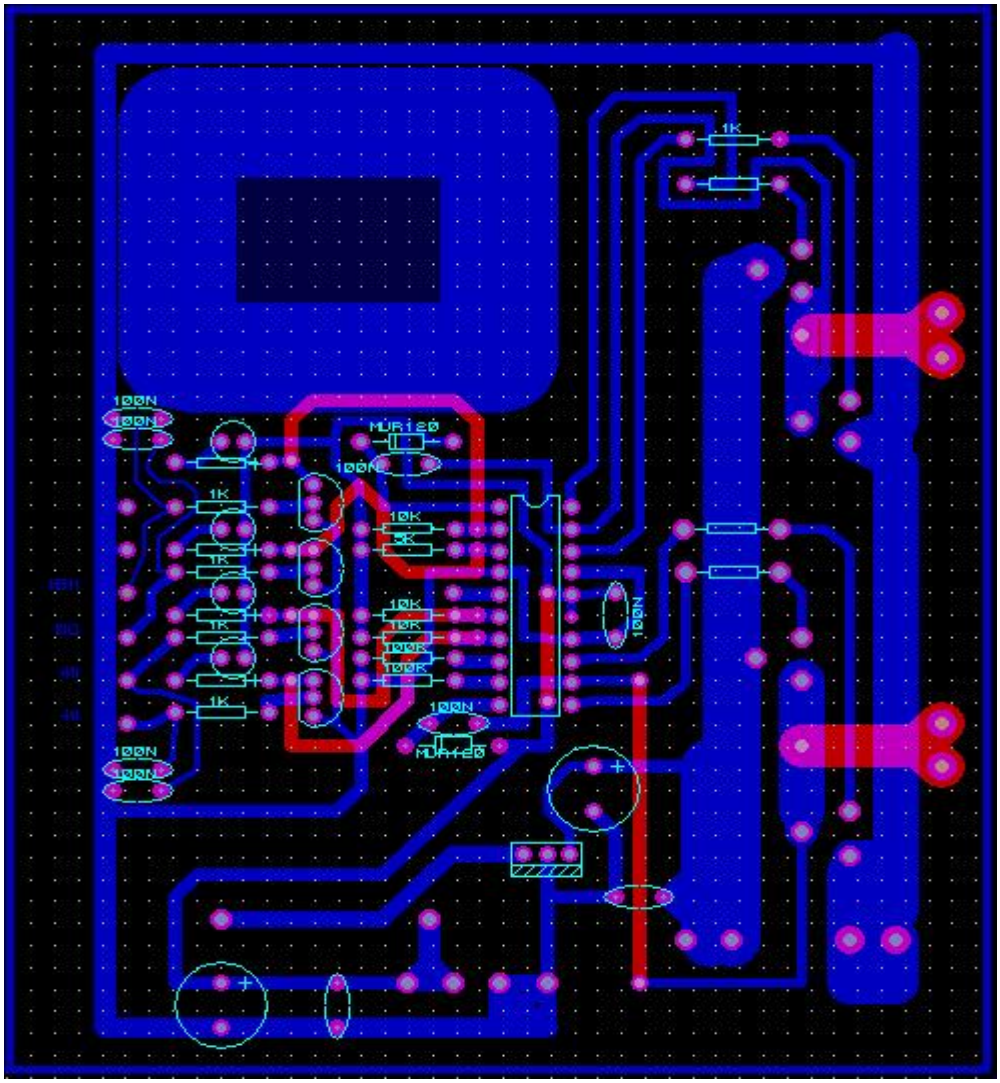


Şekil 2.6. Ana kart devresinin Üstten Görünüşü

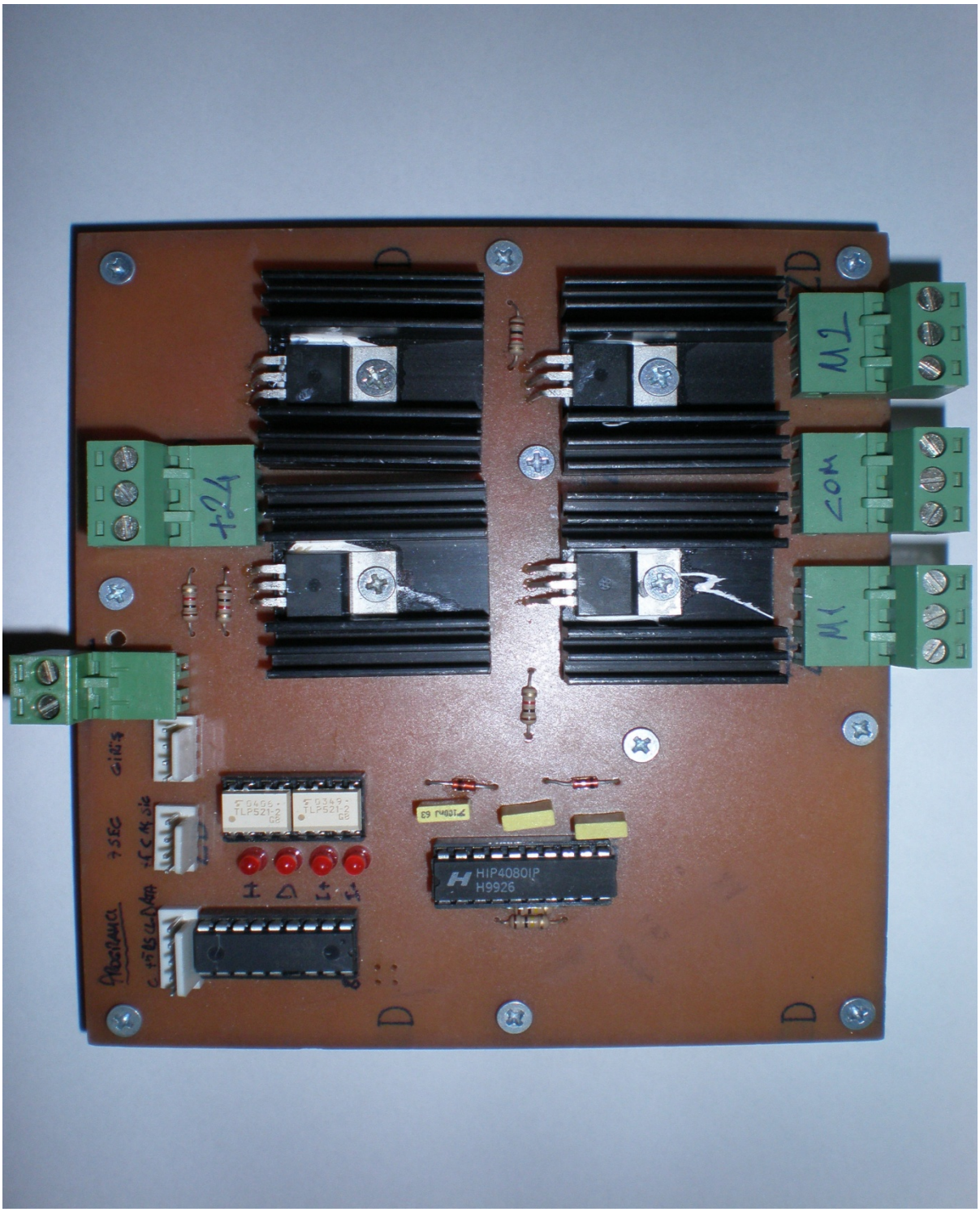
### 2.3.2. Motor Sürücü Kartı

Motor sürücü kartı ana karttan aldığı hız ve yön verisine göre Darbe Genişlik Modülasyonlu olarak ürettiği işaretle motoru değişik güçlerde ve ileri-geri sürebilme özelliğine sahip olan kartımızdır. Ana kartla seri haberleşme metodu ile haberleşmek üzere tasarlanmıştır ancak daha sonra mikro işlemci hızının yüksek olmaması nedeniyle 6 tanesi hızı ve 2 tanesi de yönü ayarlayacak pinlerle haberleşmektedir. Tasarlanan kartın proteus çizimi ve baskı devresi aşağıdaki şekillerde gösterilmektedir. Ayrıca üzerindeki mikrokontrolördeki PICBASIC programı Ek 2' de verilmiştir. Motor sürücü devresi üzerinde 16F628A mikrokontrolörü haberleşmeyi ve kontrolü sağlamaktadır. Kullanılan 4 adet mosfet ise

motorun yönünü ve hızını H-bridge yöntemi ile bağlanarak kontrol etmektedir. Kullanılan mosfetler 75NF75 dir. Her bir mosfet 1 er tane soğutucu ile soğutulmaktadır.



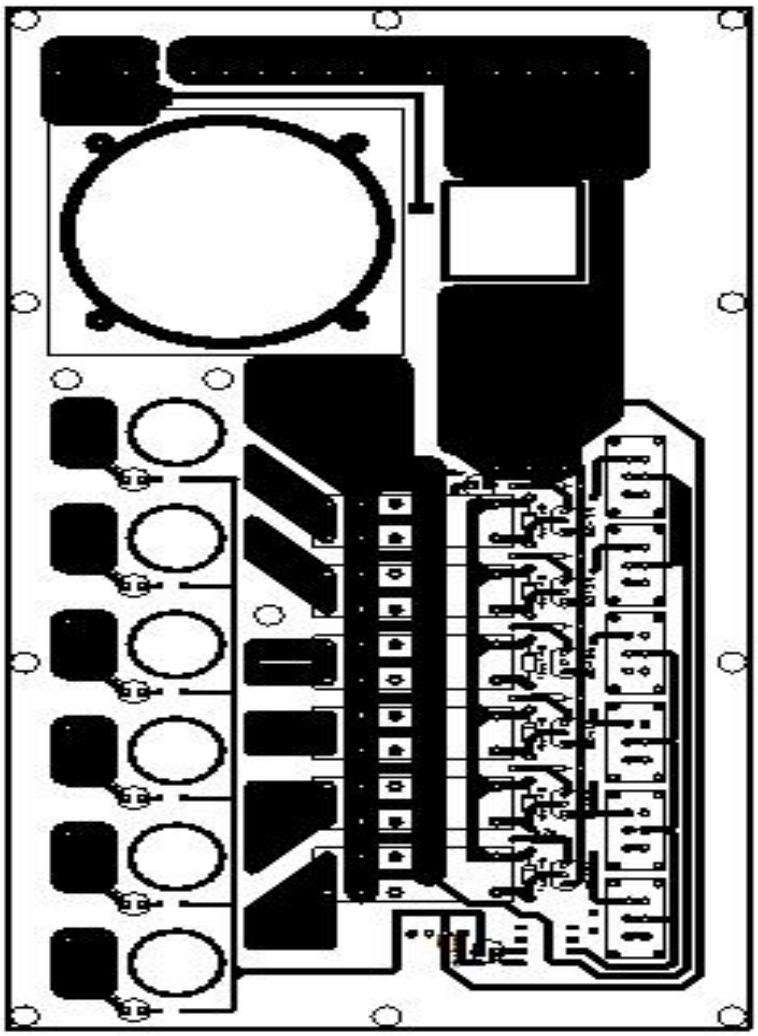

Şekil 2.7. Motor Sürücü Kartının Ares Çizimi



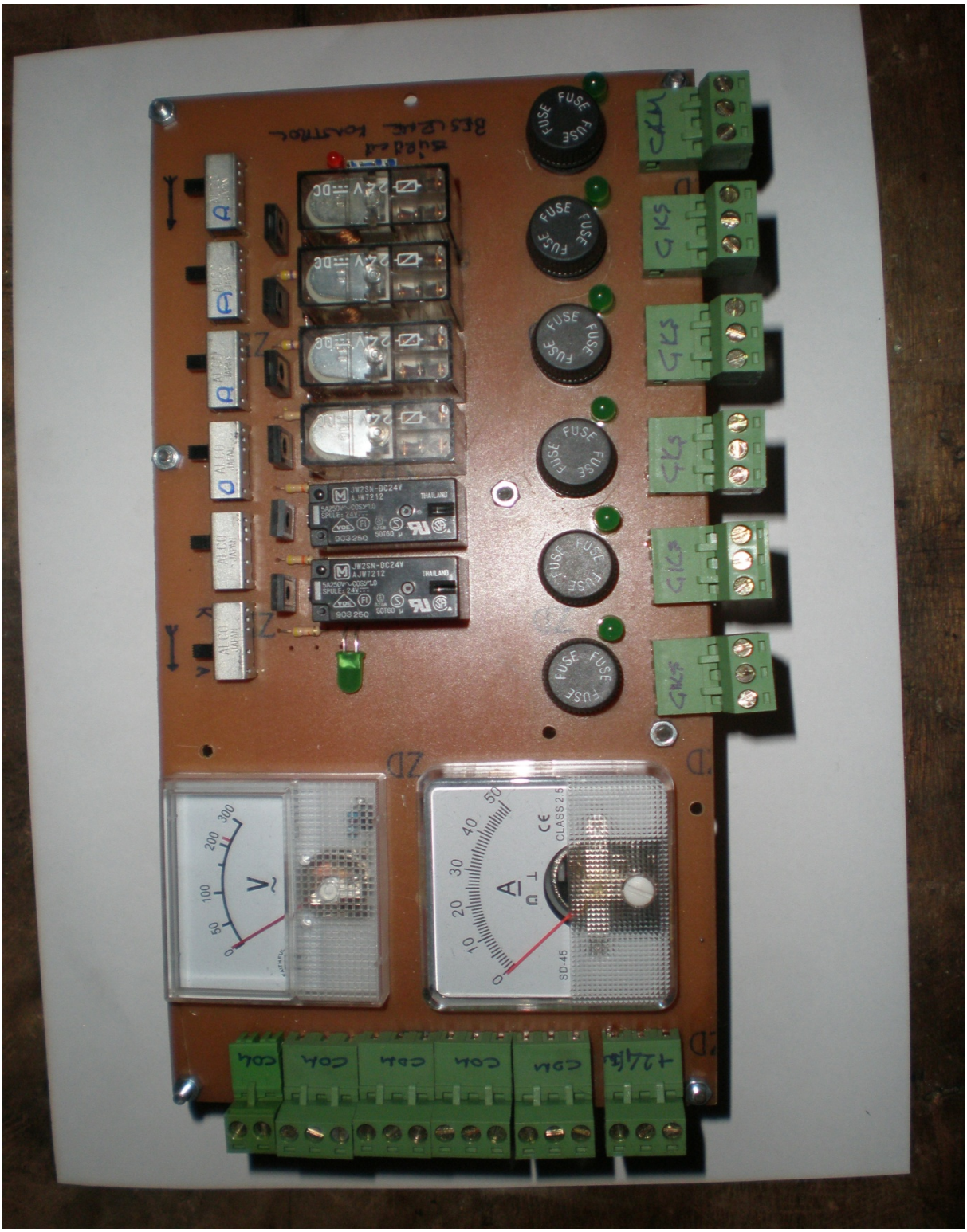

Şekil 2.8. Motor Sürücü Kartının Üstten Görünüşü

### 2.3.3. Ana Güç Besleme Kartı

Ana güç besleme kartı akülerden aldığı gücü röleler ile motoru ve sensörleri besleyen kartlar ile motora yönlendirir. Yedekli olarak yapılmıştır. 2 adet çıkış kartların gücünü tedarik etmek için kullanılır ve elle istenilen çıkıştan besleme alınabilir. Diğer 4 adet uç ise motor sürücü kartını besler ve açılıp kapatılmaları bir mikrokontrolör tarafından yapılır. Ayrıca her bir röle için 1 adet anahtar konularak kullanılacak rölelerin elle seçilebilmesi de sağlanmıştır. Ana güç besleme kartının ares devre çizimi ve resmi aşağıda verilmektedir.



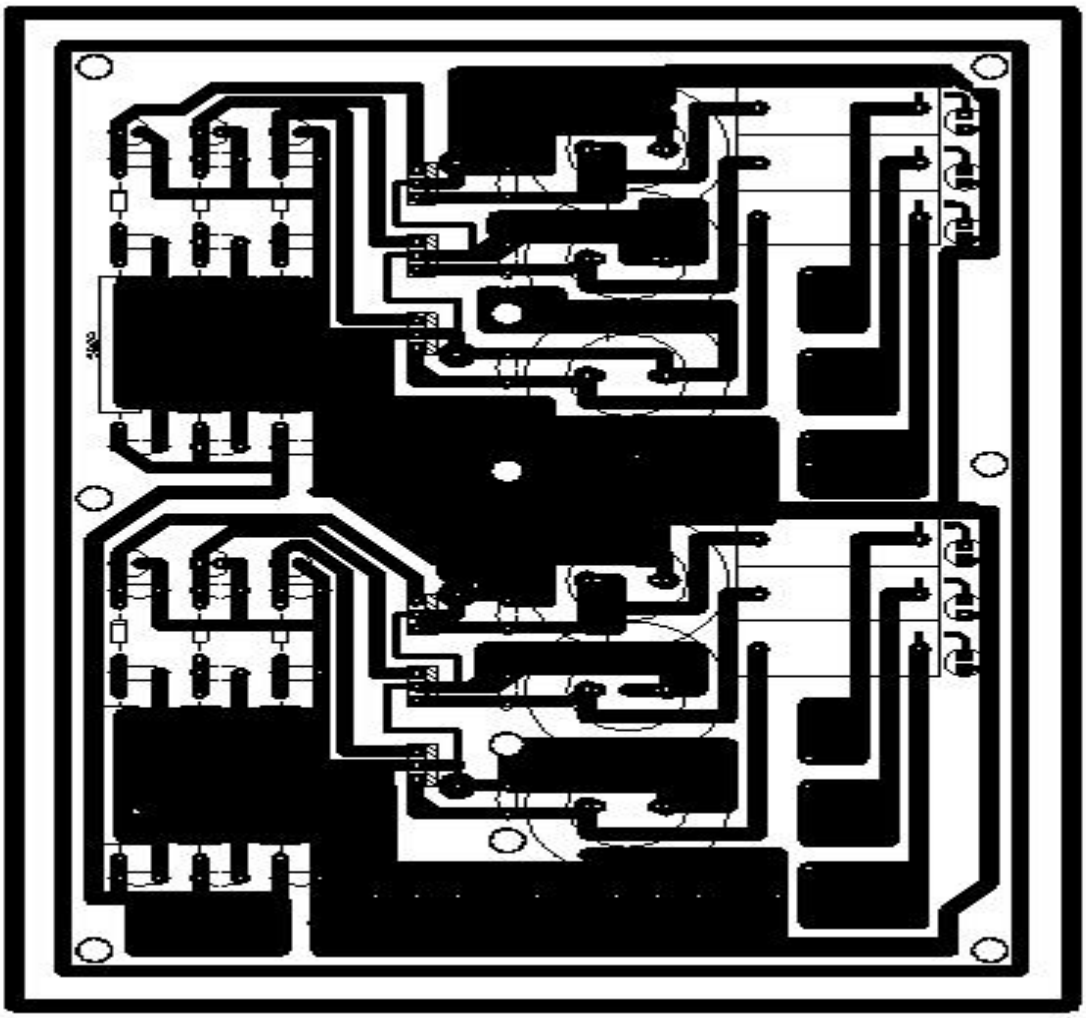
Şekil 2.9. Ana Güç Besleme Kartının Ares Çizimi



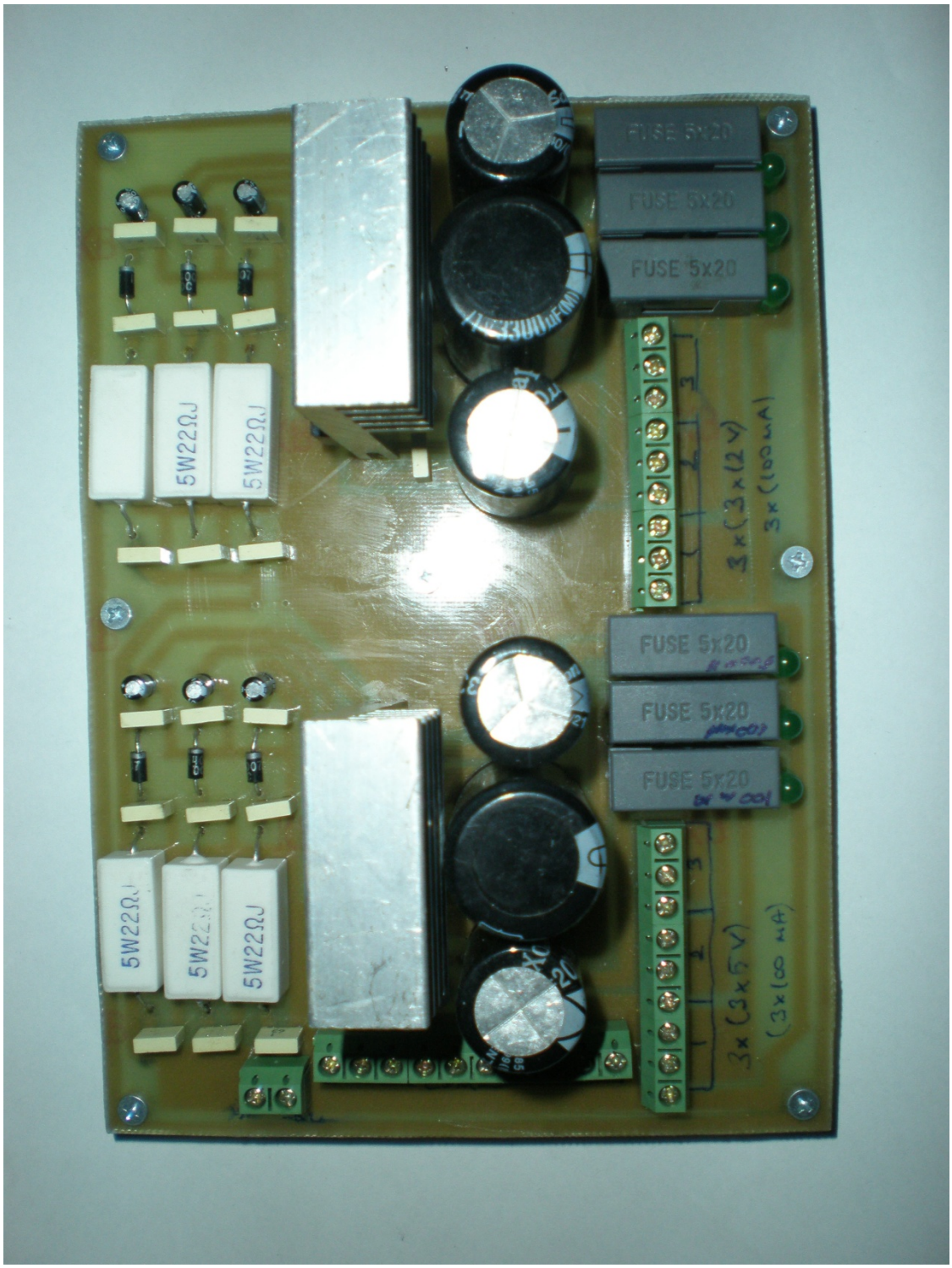

Şekil 2.10. Ana Güç Besleme Kartının Üstten Görünüşü

### 2.3.4. 5,6,8 Volt ve 12 Volt Gerilim Çıkışları İçin Regülatör Kartı

Ana kart ve sensörler için gerekli olan 5,6,8 ve 12 Voltluk gerilim çıkışlarını vermesi için tasarlanmış olan kart, ayrıca sigortalar sayesinde de akım korumalıdır.Ve bu sayede deney aşamasında beslediği kartlarda meydana gelebilecek pek çok arızayı engelleyebilmiştir. Kondansatör grupları beslenen kartların motorun veya rölelerin ana hatta verebileceği gürültüleri engellemektedir. Bu kartı yapmam sayesinde deney aşamasında yaşadığım pek çok sıkıntıyı giderdim. Tasarlanan kartın proteus çizimi ve baskı devreleri aşağıda verilmektedir.



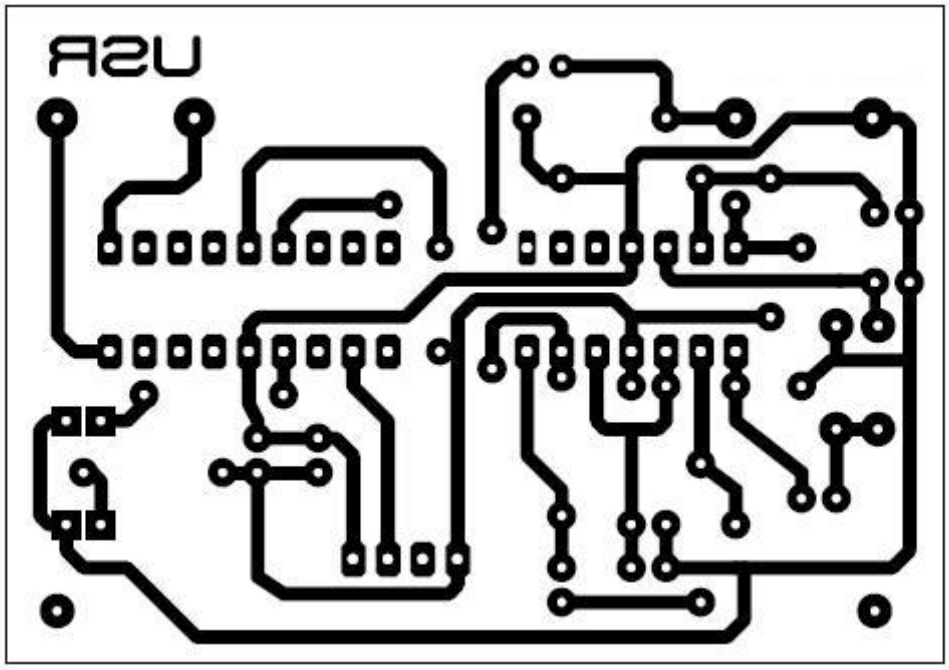
Şekil 2.11. Gerilim Regülatör Kartının Ares Çizimi



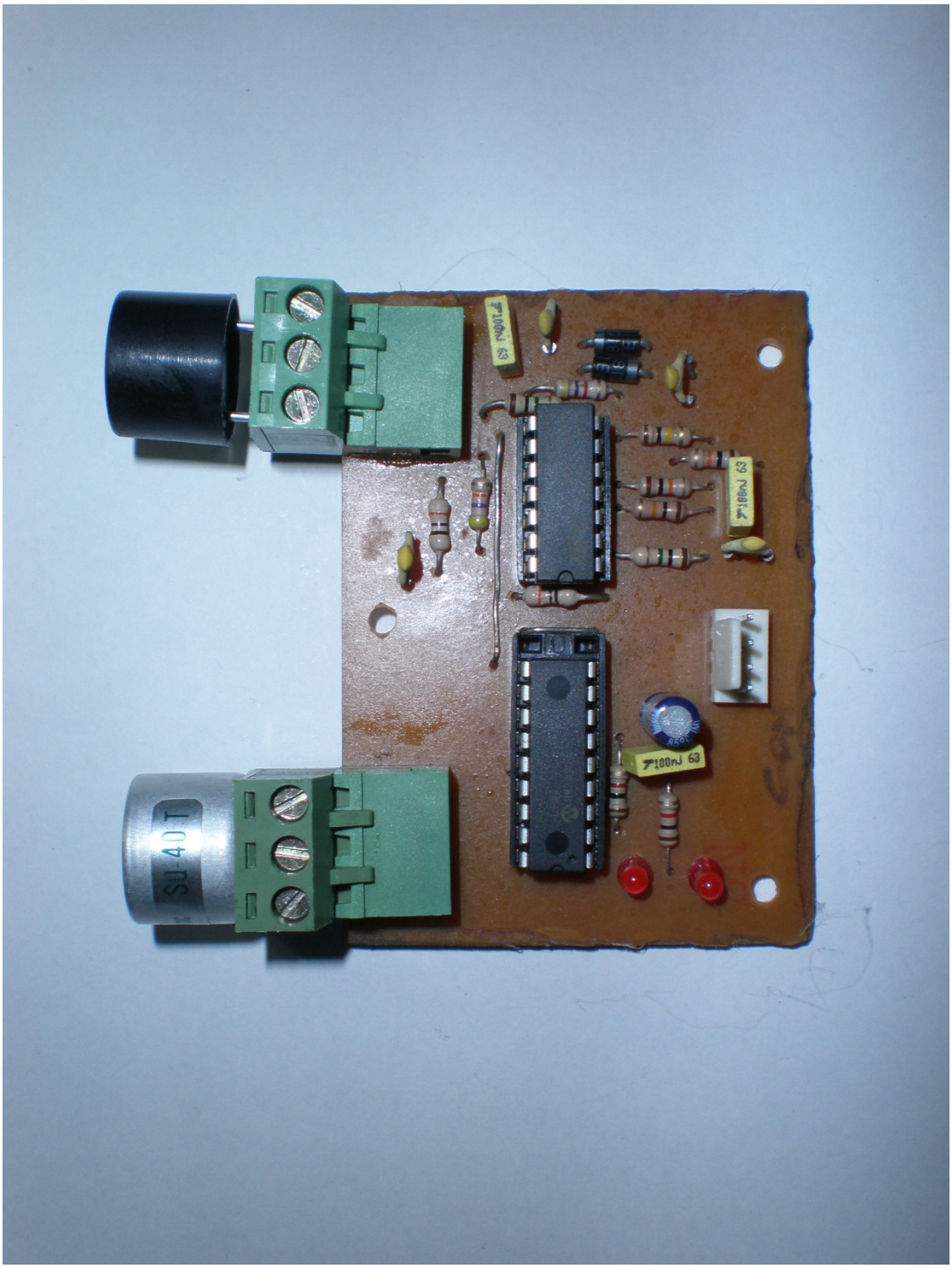

Şekil 2.12. Gerilim Regülatör Kartının Üstten Görünüşü

### 2.3.5. Ultrasonik Radar Kart

Bu kart gönderdiği 40 kHz lik ses üstü dalgaların bir yerden yansıması prensibi ile çalışmaktadır. Mikrokontrolör ürettiği 40 kHz lik sinyali verici ile bir cisimden yansıtır ve sinyalin gidiş - dönüş süresi PIC serisi 16f628A nın sahip olduğu timer1 sayesinde hesaplanır ve sonuç olarak eğer algılayıcının karşısında bir cisim var ise uzaklığı bize santimi santimine bildirilir. Bu sayede aracın ana kartına önde bir cisim olup olmadığı bildirilir. Bu veri ana karta bildirilir ve elde edilen veriye göre hareketin seyri belirlenir. Tasarlanan kartın proteus çizimi ve baskı devreleri aşağıda verilmektedir. Mikrokontrolörlere ait program kodu Ek 3' de verilmiştir.



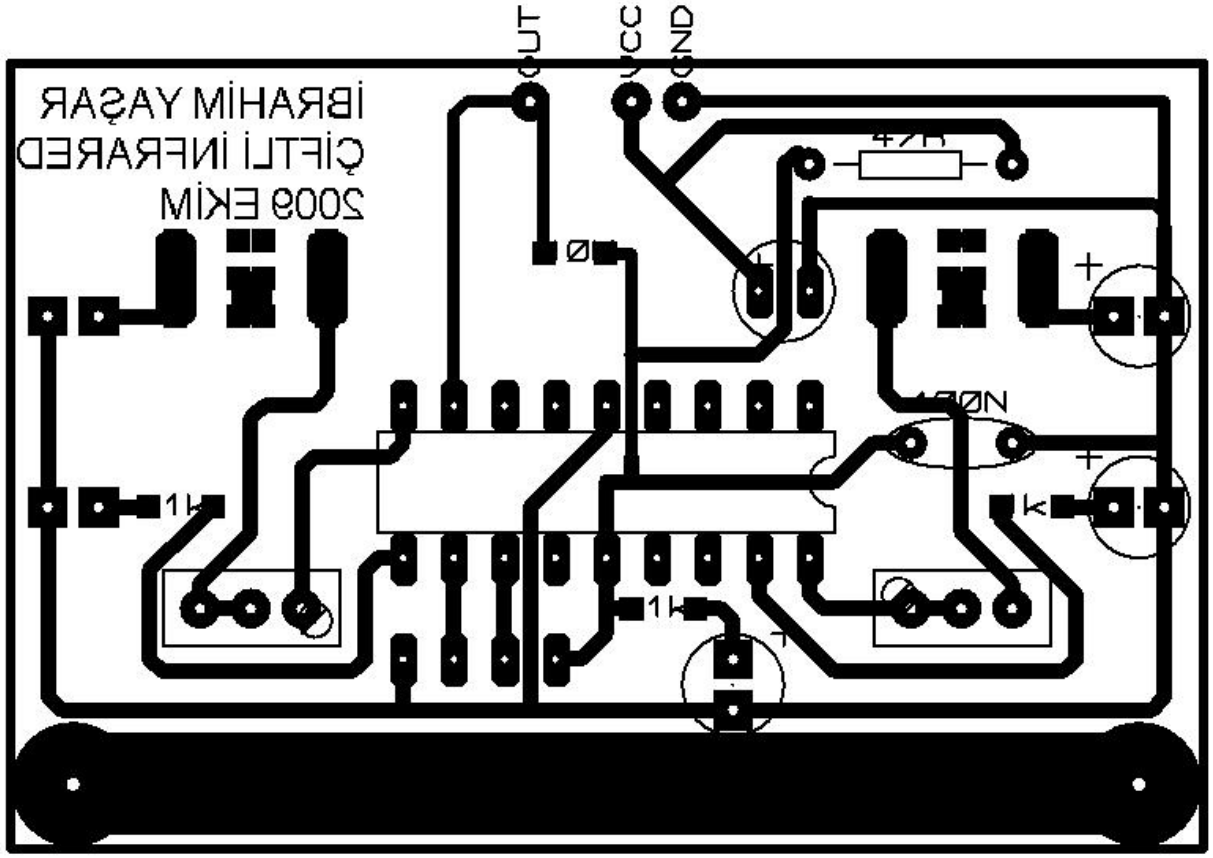
Şekil 2.13. Ultrasonik Radar Kartının Ares Çizimi



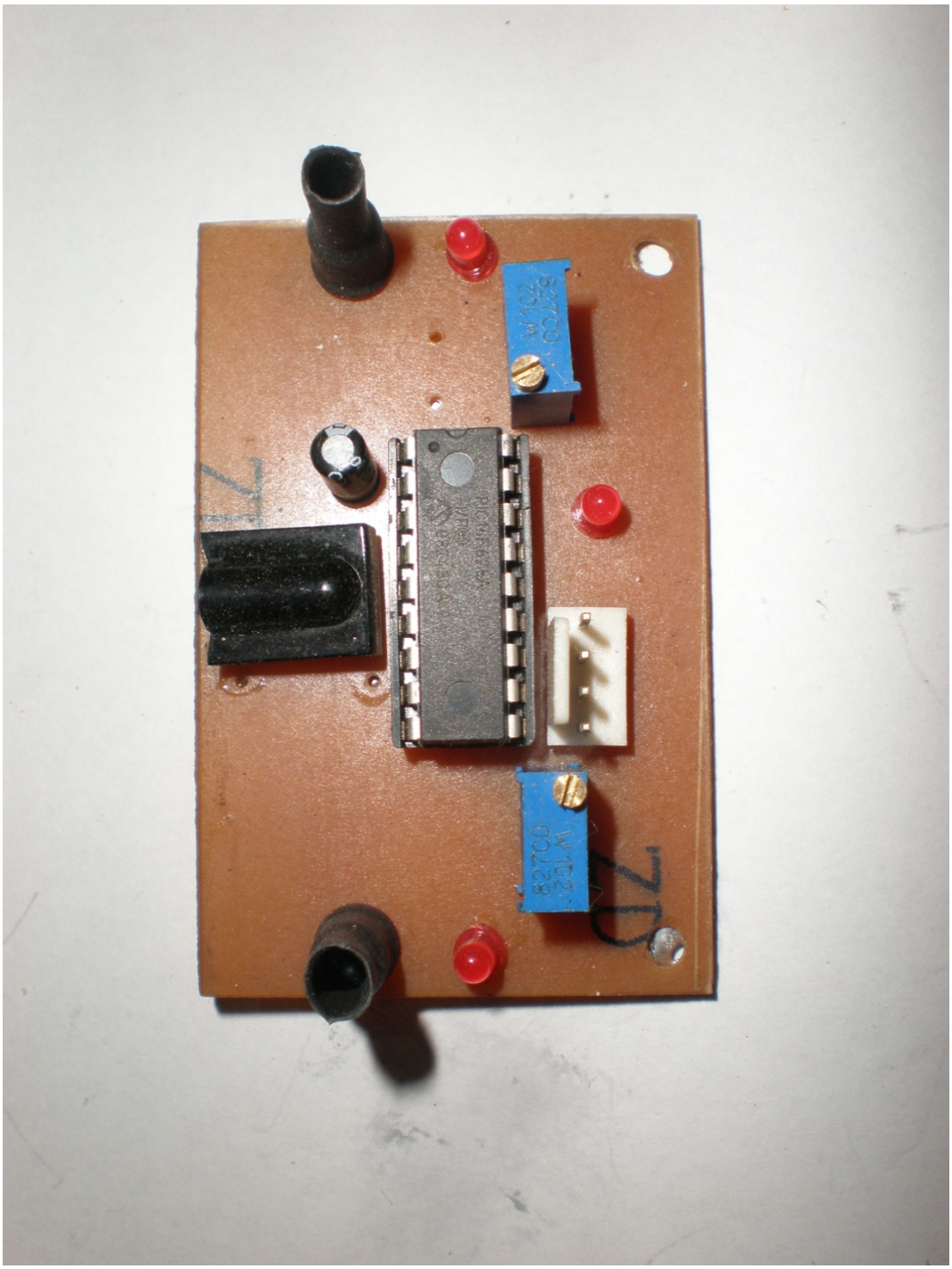

Şekil 2.14. Ultrasonik Radar Radar Kartının Üstten Görünüşü

### 2.3.6. Kızıl Ötesi Park ve Yaklaşma Algılayıcıları

Bu kartta amaçlanan araca 2 ayrı yakınlıktaki ve araca zarar verebilecek cisimlerin çarpmasını veya cisimlere çarpmayı engellemektir. Bir adet verici 18 cm ve diğeri 36 cm mesafe için kızılötesi sinyalleri göndermektedir. Bu sayede bu sınırlar içerisinde bir cisim olup olmadığı belirlenebilmektedir ve bu sayede de ana kart tarafından mesafe ayarı ve hareketler belirlenmektedir.




Şekil 2.15. Kızılötesi Park Algılayıcı Kartının Ares Çizimi



Şekil 2.16. Kızılötesi Park Algılayıcı Kartının Üstten Görünüşü

## 3. TARTIŞMA VE SONUÇLAR

İnsansız araç prensip olarak tamamen insan kontrolünden bağımsız sadece verilen görevi yerine getiren bir sistemdir. Sistem temelde bir adet sesüstü radardan aldığı yaklaşma mesafesi verisini, enkoderden aldığı hız ve mesafe verilerini işleyerek varma mesafesine hareketi sağlamaktadır. Kullanılan algılayıcılar bazı dez avantajları yanında getirmektedir. Ses yansıtıcılığı zayıf olan cisimleri algılamada, ses üstü radar zayıf kalmaktadır. Yün kazak giyen bir kişiyi algılamada mesafe kısalmaktadır. Normalde bir cismi 2,55 metreden algılayabilirken kazaklı bir insanda bu 1 metreye kadar düşmektedir. Bu ses üstü radar için bir dezavantajdır. Bu dezavantajlar ileriki çalışmalarda takviye algılayıcılarla giderilebilir.

## 4. ÖNERİLER

Bu çalışma, sayısal görüntü işaret işleme tekniği kullanılarak yapılabilecek çalışmalar için bir prototip olabilir. Cihaz üzerine konulacak, sisteme uyarlanacak bir kamera sayesinde, görüntü sayısallaştırılıp, araç bu veriler doğrultusunda daha verimli bir şekilde kontrol edilebilir. Ayrıca şu an üzerinde mevcut olmayan bir elektronik diferansiyel sistemi ile araç dönüşlerini daha rahat yapabilir. Zira şu anki çalışmada arka tekerleklerden bir tanesi milde boş dönmektedir. Diferansiyelin olmaması testler sırasında motordan aşırı akım çekmesine sebep olmuştur. Hatta birkaç keskin dönüş hareketi denemesi akım sensörünün aşırı akım algılamasını ve kartı korumaya almak için gücü kesmesi ile sonuçlanmıştır.

## 6. KAYNAKLAR


1. Frank N. Kleın, Member, Ieee, And Mark E. Kenyon, Permanent Magnet Dc Motors Design Criteria And Operation Advantages, Ieee Transactions On Industry Applications, Vol. Ia-20, No. 6, November/December 1984
2. http://www.intersil.com/design/psg/power.pdf, Power Management 2009 Product Selection Guide, Eylül 2009
3. http://www.intersil.com/data/fn/fn3658.pdf, 80V/2.5A Peak, High Frequency Full Bridge FET Driver, Eylül 2009
4. http://www.intersil.com/data/an/an9404.pdf, HIP4080A, 80V High Frequency H-Bridge Driver Application Note, Eylül 2009
5. http://ww1.microchip.com/downloads/en/DeviceDoc/39582b.pdf PIC16F87XA data sheet 28/40 Pin Enhanced Flash Microcontroller, Ocak 2009
6. http://ww1.microchip.com/downloads/en/DeviceDoc/31002a.pdf, Oscillator - PICmicro Mid-Range MCU Family, Ocak 2009
7. http://ww1.microchip.com/downloads/en/DeviceDoc/31015a.pdf, Synchronous Serial Port (SSP) - PICmicro Mid-Range MCU Family, Ocak 2009
8. http://ww1.microchip.com/downloads/en/DeviceDoc/40044G.pdf, PIC16F628A veri sayfası, Ocak 2009
9. http://pdf1.alldatasheet.com/datasheet-pdf/view/82833/FAIRCHILD/LM7805.html, MC78XX/LM78XX/MC78XXA 3-Terminal 1A Positive Voltage Regulator, Mayıs 2008
10. http://www.st.com/stonline/products/literature/ds/8451/stp75nf75.pdf, STB75NF75 veri sayfası, Eylül 2009
11. http://pdf1.alldatasheet.com/datasheet-pdf/view/32438/TOSHIBA/TLP521.html , Optocoupler TLP521-2, TLP521-4, Ağustos 2009
12. Sergey Edward Lyshevski, Control Systems Theory with Engineering Applications, Springer International Edition
13. P.Vas, Electrical machines and drives : a space-vector theory approach, Oxford University Press, New York, 1992.
14. Kızılbey, Deniz Suyu Termometresi, Lisans Tezi, İstanbul Teknik Üniversitesi, 2005
15. David Benson, Easy PIC'n: a beginner's guide to using PIC16/17 microcontrollers from square 1, Square 1 electronics , 1997.
16. Michael Predko, Programming and Customizing PIC Microcontrollers, McGraw Hill , 2001.
17. Martin Bates, Introduction to Microelectronic Systems, London : Arnold, 2000.

18. http://utkuonline.tripod.com/PIC16f877.html, PIC16F877 yapısı, Haziran 2009

**EKLER**

**Ek-1 : Ana Kart Üzerindeki Mikrokontrolörlere Ait Program Kodları**

```
'*****************************************************************
'*  Name    : ANAPIC.BAS                                         *
'*  Author  : [İBRAHİM YAŞAR]                                    *
'*  Notice  : İNSANSIZ KARA ARACI PROJESİNİN ANA MİKROİŞLEMCİSİDİR*
'*  Date    : 03.11.2009                                         *
'*  Version : 1.0                                                *
'*  Notes   :                                                    *
'*          :                                                    *
'*****************************************************************
'CONFIFURATION REGISTERIN AYARLAMALARI
@ DEVICE PROTECT_OFF,WDT_OFF,PWRT_ON
@ DEVICE    XT_OSC
INCLUDE "MODEDEFS.BAS"
'*****************************************************************
'STANDART REGISTER AYARLAMALARI
CMCON=%00000111 'COMPARATOR LERİ KAPATTIK.
OPTION_REG=%10000000 'PULL UP DİRENÇLERİNİ DISABLE ETTİK.
ADCON1=%00000110
'*****************************************************************

'DENEME DEĞİŞKENLERİ
DENUS   VAR BYTE
DENUS=0
DENKIZ  VAR BYTE
DENKIZ=0
DENMES  VAR BYTE
DENMES=0
```

**Ek 1'in devamı**

```
SAY VAR WORD
SAY=0
KONTROL VAR BYTE
KONTROL=0
'*************************************************************
'DEĞİŞKENLERİN TANIMLANMASI, PINLENLERİN İSİMLENDİRİLMESİ
'VE PORT DURUMLARININ TANIMLANMASI
'KIZILÖTESİ ALGILAYICI SERİ VERİ DEĞİŞKENLERİNİN TANIMLANMASI
KIZIL_1 VAR BYTE
KIZIL_2 VAR BYTE
KIZIL_3 VAR BYTE
KIZIL_1=0
KIZIL_2=0
KIZIL_3=0
SYMBOL  KIZIL_KESME=PORTA.0
SYMBOL  KIZIL_TEYIT=PORTA.1
SYMBOL  KIZIL_VERI=PORTA.2
TRISA.0=0
TRISA.1=1
TRISA.2=1
KIZIL_KESME=0
'-------------------------------------------------------------
'ULTRASONİK VERİ ARACISI SERİ VERİ DEĞİŞKENLERİNİN TANIMLANMASI
BILGIUS    VAR BYTE
BILGIUS=0
SYMBOL  ANAUSKESME=PORTA.3
SYMBOL  ANAUSTEYIT=PORTA.5
SYMBOL  ANAUSVERI=PORTE.0
TRISA.3=0
TRISA.5=1
```

**Ek 1'in devamı**

```
TRISE.0=1
ANAUSKESME=0
'-------------------------------------------------------------
'MESAFE VERİSİ HESAPLAYICISI SERİ VERİ DEĞİŞKENLERİNİN TANIMLANMASI
METRE VAR BYTE
CM    VAR BYTE
METRE=0
CM=0
SYMBOL  MESAFE_KESME=PORTC.1
SYMBOL  MESAFE_TEYIT=PORTC.2
SYMBOL  MESAFE_VERI=PORTC.3
SYMBOL  MESAFE_SIL=PORTD.0
TRISC.1=0
TRISC.2=1
TRISC.3=1
TRISD.0=0
MESAFE_KESME=0
MESAFE_SIL=0
'-------------------------------------------------------------
'DEVİR HIZI VERİSİ HESAPLAYICISI SERİ VERİ DEĞİŞKENLERİNİN
TANIMLANMASI
HIZSN VAR BYTE
HIZSN=0
HIZREF VAR BYTE
HIZREF=0
SYMBOL  HIZ_KESME=PORTD.1
SYMBOL  HIZ_TEYIT=PORTD.2
SYMBOL  HIZ_VERI=PORTD.3
TRISD.1=0
TRISD.2=1
```

**Ek 1'in devamı**

```
TRISD.3=1
HIZ_KESME=0
'-------------------------------------------------------------
'YON VERİSİ HESAPLAYICISI SERİ VERİ DEĞİŞKENLERİNİN TANIMLANMASI
YONTUR  VAR BYTE
YON_SON  VAR BYTE
YONTUR=0
YON_SON=2
SYMBOL  YON_KESME=PORTE.1
SYMBOL  YON_TEYIT=PORTE.2
SYMBOL  YON_VERI=PORTC.0
TRISE.1=0
TRISE.2=1
TRISC.0=1
YON_KESME=10
'-------------------------------------------------------------
'MOTOR SÜRÜCÜSÜ İÇİN DEĞİŞKENLER
MOTOR   VAR BYTE
MOTOR=28
TRISC.4=0
TRISC.5=0
TRISC.6=0
TRISB.0=0
TRISB.2=0
TRISB.3=0
TRISB.4=0
TRISB.5=0
GOSUB   MOTORATAMA
'-------------------------------------------------------------
'*************************************************************
```

**Ek 1'in devamı**

```
'LCD YE VERİ YAZDIRILMASI PROTOKOLÜ
 'LCD data bacakları hangi Porta bağlı?
DEFINE LCD_DREG                 PORTD
'LCD data bacakları hangi bitten başlıyor?
DEFINE LCD_DBIT          4
'LCD Enable Bacağı Hangi Porta bağlı?
DEFINE LCD_EREG                 PORTB
'LCD Enable Bacağı Hangi bite bağlı ?
DEFINE LCD_EBIT          1
"LCD R/W Bacağı Hangi Porta bağlı?
'DEFINE LCD RWREG       PORTB
"LCD R/W Bacağı Hangi bite bağlı ?
'DEFINE LCD_RWBIT       5
'LCD RS Bacağı Hangi Porta bağlı ?
DEFINE LCD_RSREG        PORTC
'LCD RS bacağı Hangi Bite bağlı  ?
DEFINE LCD_RSBIT         7
'LCD 4 bit mi yoksa 8 bit olarak bağlı?
DEFINE LCD_BITS          4
'LCD Kaç sıra yazabiliyor
DEFINE LCD_LINES         2
'*****************************************************************
'EKRANI SİL
LCDOUT       $FE,1
'LCD nin açılması için gerekli süredir.
pause   200
'*****************************************************************
'PROGRAM BAŞLANGICI
ANADONGU:
   'KIZILÖTESİ MİKROKONTROLÖRÜNDEN VERİ ALINMASI
```

**Ek 1'in devamı**

```
   KIZIL_KESME=1
   KIZIL_BEKLE:
      IF  KIZIL_TEYIT==1  THEN
         SERIN   KIZIL_VERI,N2400,["K"],KIZIL_1,KIZIL_2,KIZIL_3
         KIZIL_KESME=0
      ELSE
         GOTO    KIZIL_BEKLE
      ENDIF
'-----------------------------------------------------------------
'ULTRASONİK ARACI MİKROKONTROLÖRÜNDEN VERİ ALINMASI
   ANAUSKESME=1
   USBEKLE:
      IF  ANAUSTEYIT==1  THEN
         SERIN   ANAUSVERI,N2400,["A"],BILGIUS
         ANAUSKESME=0
      ELSE
         GOTO    USBEKLE
      ENDIF
'-----------------------------------------------------------------
'MESAFE MİKROKONTROLÖRÜNDEN VERİ ALINMASI
   MESAFE_KESME=1
   MESAFE_BEKLE:
      IF  MESAFE_TEYIT==1  THEN
         SERIN   MESAFE_VERI,N2400,["A"],METRE,CM
         IF  METRE>=250   THEN
            MESAFE_SIL=1
            PAUSE   1
            MESAFE_SIL=0
         ENDIF
         MESAFE_KESME=0
```

**Ek 1'in devamı**

```
      ELSE
        GOTO    MESAFE_BEKLE
      ENDIF
'-------------------------------------------------------------
'HIZ ARACI MİKROKONTROLÖRÜNDEN VERİ ALINMASI
   HIZ_KESME=1
   HIZBEKLE:
      IF  HIZ_TEYIT==1  THEN
        SERIN   HIZ_VERI,N2400,["A"],HIZSN
        HIZ_KESME=0
      ELSE
        GOTO    HIZBEKLE
      ENDIF
'-------------------------------------------------------------
'YON VE TUR MİKROKONTROLÖRÜNDEN VERİ GÖNDERİLMESİ
   IF  YON_SON<>0  THEN
      YON_KESME=1
      YONBEKLE:
        IF  YON_TEYIT==1  THEN
          SEROUT YON_VERI,N2400,[("A"),YON_SON]
          YON_KESME=0
          IF  YON_TEYIT==1    THEN
            GOTO    YONBEKLE
          ENDIF
        ELSE
          GOTO    YONBEKLE
        ENDIF
      YON_SON=0
   ENDIF
'-------------------------------------------------------------
```

**Ek 1'in devamı**

```
'MOTOR SÜRÜCÜSÜNÜN DEĞERİNİ BELİRLER
'   IF  MOTOR<90    THEN
'      MOTOR=MOTOR+4
'      GOSUB   MOTORATAMA
'   ELSE
'      GOSUB   MOTORSIFIRLA
'      GOSUB   MOTORATAMA
'      PAUSE   100
'      MOTOR.0=1
'      MOTOR.1=!MOTOR.1
'      GOSUB   MOTORATAMA
'   ENDIF
'-------------------------------------------------------------
'   IF  BILGIUS>100 THEN
'      GOSUB   MOTORSIFIRLA
'      MOTOR=97
'      MOTOR.0=1
'      MOTOR.1=1
'      GOSUB   MOTORATAMA
'   ELSE
'      GOSUB   MOTORSIFIRLA
'      MOTOR=0
'      MOTOR.0=0
'      MOTOR.1=0
'      GOSUB   MOTORATAMA
'   ENDIF
'-------------------------------------------------------------
IF  METRE<92    THEN
   IF  BILGIUS>200 THEN
      MOTOR=96
```

**Ek 1'in devamı**

```
      MOTOR.0=1 'MOTOR ÇALIŞ
      MOTOR.1=1 'MOTOR İLERİ
      GOSUB   MOTORATAMA
   ELSE
      GOSUB   MOTORSIFIRLA
      GOSUB   MOTORATAMA
   ENDIF
ELSE
      GOSUB   MOTORSIFIRLA
      GOSUB   MOTORATAMA
ENDIF
'TOPLANAN VERİLERİN EKRANA YAZDIRILMASI
   SAY=SAY+1
   LCDOUT  $FE,1,"HR:",#HIZREF,"-HS:",#HIZSN
'    LCDOUT  $FE,1,#BILGIUS," CM"
'    LCDOUT  $FE,1,#HIZ_VERI," DEVIR"
'    LCDOUT  $FE,$C0,
   LCDOUT  $FE,$C0,#METRE," MT"
GOTO ANADONGU
'************************************************************
MOTORSIFIRLA:
   MOTOR.0=0
   MOTOR.1=0
   MOTOR.2=0
   MOTOR.3=0
   MOTOR.4=0
   MOTOR.5=0
   MOTOR.6=0
   MOTOR.7=0
RETURN
```

**Ek 1'in devamı**

```
MOTORATAMA:
   PORTC.4=MOTOR.0
   PORTC.5=MOTOR.1
   PORTC.6=MOTOR.2
   PORTB.0=MOTOR.3
   PORTB.2=MOTOR.4
   PORTB.3=MOTOR.5
   PORTB.4=MOTOR.6
   PORTB.5=MOTOR.7
RETURN
END


'****************************************************************
'*  Name    : DIREKSIYON-REV02.BAS                              *
'*  Author  : [İBRAHİM YAŞAR]                                   *
'****************************************************************
'CONFIFURATION REGISTERIN AYARLAMALARI
@ DEVICE PROTECT_OFF,WDT_OFF,PWRT_ON,MCLR_OFF
@ DEVICE    INTRC_OSC_NOCLKOUT
INCLUDE "MODEDEFS.BAS"
'****************************************************************
'DENEME DEĞİŞKENLERİ
PORTA=0
PORTB=0
'STANDART REGISTER AYARLAMALARI
ON INTERRUPT GoTo KESME     'kesme oluşursa KESME adlı etikete git.
CMCON=%00000111 'COMPARATOR LERİ KAPATTIK.
OPTION_REG=%11000000 'PULL UP DİRENÇLERİNİ DISABLE ETTİK.
INTCON=%10010000          'Tüm Kesmeler aktif ve RB0/INT kesmesi aktif
```

**Ek 1'in devamı**

```
PCON=%00001000 'KRISTAL 4 MHZ OLARAK AYARLANDI.
'************************************************************
'ANA MİKROKONTROLÖR İLE HABERLEŞMENİN SAĞLANMASI İÇİN KULLANILAN
'DEĞİŞKENLER
YONTUR VAR WORD
YONTUR=0
BEKLEME VAR BYTE
BEKLEME=0
SYMBOL  YON_KESME=PORTB.0
SYMBOL  YON_TEYIT=PORTB.1
SYMBOL  YON_VERI=PORTB.2
TRISB.0=1
TRISB.1=0
TRISB.2=0
YON_TEYIT=0
'************************************************************
'YÖNLENDİRME DEĞİŞKENLERİ
SYMBOL  SAG=PORTA.0
SYMBOL  SOL=PORTA.1
TRISA.0=0
TRISA.1=0
SAG=0
SOL=0
'************************************************************
'ANA PROGRAM BURADAN İTİBAREN BAŞLAR
BEKLEME=100
ANADONGU:
   SAG=1
   PAUSE   BEKLEME
```

**Ek 1'in devamı**

```
   SAG=0
   PAUSE   BEKLEME
GOTO ANADONGU
'*****************************************************************
DISABLE
KESME:
   IF  INTCON.1==1 THEN
      YON_TEYIT=1
      AL:
         SERIN   YON_VERI,N2400,["A"],YONTUR
         IF  YON_KESME==0  THEN
            YON_TEYIT=0
         ELSE
            GOTO    AL
         ENDIF
         YONTUR=YONTUR*500
      INTCON.1=0
   ENDIF
RESUME
ENABLE
END

'*****************************************************************
'*  Name    : HIZOLCER-REV02.BAS                                 *
'*  Author  : [İBRAHİM YAŞAR]                                    *
'*****************************************************************
'CONFIFURATION REGISTERIN AYARLAMALARI
@ DEVICE PROTECT_OFF,WDT_OFF,PWRT_ON,MCLR_OFF
@ DEVICE    INTRC_OSC_NOCLKOUT
INCLUDE "MODEDEFS.BAS"
```

**Ek 1'in devamı**

```
'*****************************************************************
'DEĞİŞKENLER
KARE    VAR WORD
PORTA=0
PORTB=0
'STANDART REGISTER AYARLAMALARI
ON INTERRUPT GoTo KESME     'kesme oluşursa KESME adlı etikete git.
CMCON=%00000111 'COMPARATOR LERİ KAPATTIK.
TRISA=%00010000 'PORTA.4 BİTİ KARE DALGA SAYICI OLDU.
OPTION_REG=%11100000 'PULL UP DİRENÇLERİNİ DISABLE ETTİK.
INTCON=%10010000        'Tüm Kesmeler aktif ve RB0/INT kesmesi ile TMR0 kesmesi aktif
PCON=%00001000 'KRISTAL 4 MHZ OLARAK AYARLANDI.
'*****************************************************************
'ANA MİKROKONTROLÖR İLE HABERLEŞMENİN SAĞLANMASI İÇİN KULLANILAN
'DEĞİŞKENLER
DEVSN VAR BYTE
DEVSN=0
SYMBOL  HIZ_KESME=PORTB.0
SYMBOL  HIZ_TEYIT=PORTB.1
SYMBOL  HIZ_VERI=PORTB.2
TRISB.0=1
TRISB.1=0
TRISB.2=0
HIZ_TEYIT=0
TMR0=0
'*****************************************************************
ANADONGU:
   TMR0=0
```

**Ek 1'in devamı**

```
   PAUSE   100
   DISABLE
   KARE=TMR0
   KARE=KARE*14
   DEVSN=KARE
   ENABLE
GOTO ANADONGU
DISABLE
KESME:
   IF  INTCON.1==1 THEN
      HIZ_TEYIT=1
      GONDER:
         SEROUT HIZ_VERI,N2400,[("A"),DEVSN]
         IF  HIZ_KESME==0  THEN
            HIZ_TEYIT=0
         ELSE
            GOTO    GONDER
         ENDIF
      INTCON.1=0
   ENDIF
RESUME
ENABLE
END

'*****************************************************************
'*  Name    : MESAFEOLCER-REV03.BAS                              *
'*  Author  : [İBRAHİM YAŞAR]                                    *
'*****************************************************************
'CONFIFURATION REGISTERIN AYARLAMALARI
@ DEVICE PROTECT_OFF,WDT_OFF,PWRT_ON,MCLR_OFF
```

**Ek 1'in devamı**

```
'@ DEVICE    INTRC_OSC_NOCLKOUT
@ DEVICE    XT_OSC
INCLUDE "MODEDEFS.BAS"
'************************************************************
'DENEME DEĞİŞKENLERİ
DENMES   VAR BYTE
PORTA=0
PORTB=0
'------------------------------------------------------------
'STANDART REGISTER AYARLAMALARI
ON INTERRUPT GoTo KESME     'kesme oluşursa KESME adlı etikete git.
'PCON=%00001000 'KRISTAL 4 MHZ OLARAK AYARLANDI.
CMCON=%00000111 'COMPARATOR LERİ KAPATTIK.
TRISA=%00010000 'PORTA.4 BİTİ KARE DALGA SAYICI OLDU.
OPTION_REG=%11100000 'PULL UP DİRENÇLERİNİ DISABLE ETTİK.
INTCON=%10010000          'Tüm Kesmeler aktif ve RB0/INT kesmesi ile TMR0 kesmesi
aktif

'************************************************************
'ANA MİKROKONTROLÖR İLE HABERLEŞMENİN SAĞLANMASI İÇİN
KULLANILAN
'DEĞİŞKENLER
METRE   VAR BYTE
METRE1  VAR BYTE
CM      VAR BYTE
ARTAN   VAR BYTE
METRE=0
METRE1=0
CM=0
ARTAN=0
```

**Ek 1'in devamı**

```
SYMBOL  ANAMESKESME=PORTB.0
SYMBOL  ANAMESTEYIT=PORTB.1
SYMBOL  ANAMESVERI=PORTB.2
SYMBOL  ANAMESSIL=PORTB.3
TRISB.0=1
TRISB.1=0
TRISB.2=0
TRISB.3=1
ANAMESTEYIT=0
'************************************************************
'KARE DALGALARIN SAYILMASI İÇİN GEREKLİ DEĞİŞKENLER
TMRKESSAY   VAR     BYTE
KARE        VAR     BYTE
KARE=0
TMR0=0
'************************************************************
ANADONGU:
   CM=(TMR0*2+ARTAN)*14/10
   IF  CM>=100  THEN
      TMR0=0
      METRE1=CM/100
      ARTAN=CM-METRE1*100
      METRE=METRE1+METRE

   ENDIF
GOTO ANADONGU

DISABLE
KESME:
   IF  INTCON.1==1 THEN
```

**Ek 1'in devamı**

```
      ANAMESTEYIT=1
      GONDER:
        SEROUT ANAMESVERI,N2400,[("A"),METRE,CM]
        IF  ANAMESSIL==1 THEN
          METRE=0
          CM=0
        ENDIF
        IF  ANAMESKESME==0  THEN
          ANAMESTEYIT=0
        ELSE
          GOTO    GONDER
        ENDIF
      INTCON.1=0
   ENDIF
RESUME
ENABLE
END

'*****************************************************************
'*  Name    : KIZILOTESI.BAS                                     *
'*  Author  : [İBRAHİM YAŞAR]                                    *
'*****************************************************************
'CONFIFURATION REGISTERIN AYARLAMALARI
@ DEVICE PROTECT_OFF,WDT_OFF,PWRT_ON,XT_OSC
INCLUDE "MODEDEFS.BAS"
'*****************************************************************
BILGI1  VAR BYTE
BILGI2  VAR BYTE
BILGI3  VAR BYTE
PORTA=0
```

**Ek 1'in devamı**

```
PORTB=0
'*****************************************************************
'STANDART REGISTER AYARLAMALARI
ON INTERRUPT GoTo KESME     'KESME OLUŞURSA KESME ADLI ETİKETE GİT.
INTCON=%10010000            ' RB0 KESMESİNİ AKTİF EDER.
OPTION_REG=%11000000        'PULL UP DİRENÇLERİNİ DISABLE ETTİK.
PORTA=0
PORTB=0
PORTC=0
PORTD=0
PORTE=0
TRISA=%00111110
TRISB=%00000001
TRISC=%11111111
TRISD=%11111111
TRISE=%00000111
ADCON1=%00000110
"*****************************************************************
'DEĞİŞKENLERİN TANIMLANMASI, PINLENLERİN İSİMLENDİRİLMESİ
'VE PORT DURUMLARININ TANIMLANMASI
'KIZILÖTESİ ALGILAYICI SERİ VERİ DEĞİŞKENLERİNİN TANIMLANMASI
KIZIL_1 VAR BYTE
KIZIL_2 VAR BYTE
KIZIL_3 VAR BYTE
KIZIL_1=0
KIZIL_2=0
KIZIL_3=0
SYMBOL  KIZIL_KESME=PORTB.0
SYMBOL  KIZIL_TEYIT=PORTB.1
SYMBOL  KIZIL_VERI=PORTB.2
```

**Ek 1'in devamı**

```
TRISB.0=1
TRISB.1=0
TRISB.2=0
KIZIL_TEYIT=0
'-------------------------------------------------------------

'*************************************************************
'SENSÖR DEĞİŞKENLERİNE SENSÖRLEDİN DURUMLARI AKTARILIYOR.
SYMBOL  ON_SAG_YAK=PORTA.1      ' 1.NUMARALI GİRİŞ
SYMBOL  ON_SAG_UZA=PORTA.2      ' 1.NUMARALI GİRİŞ
SYMBOL  ON_SOL_YAK=PORTA.4      ' 2.NUMARALI GİRİŞ
SYMBOL  ON_SOL_UZA=PORTA.3      ' 2.NUMARALI GİRİŞ
SYMBOL  SOL_ON_YAK=PORTA.5      ' 3.NUMARALI GİRİŞ
SYMBOL  SOL_ON_UZA=PORTE.0      ' 3.NUMARALI GİRİŞ
SYMBOL  SOL_AR_YAK=PORTE.1      ' 4.NUMARALI GİRİŞ
SYMBOL  SOL_AR_UZA=PORTE.2      ' 4.NUMARALI GİRİŞ
SYMBOL  AR_SOL_YAK=PORTC.0      ' 5.NUMARALI GİRİŞ
SYMBOL  AR_SOL_UZA=PORTC.1      ' 5.NUMARALI GİRİŞ
SYMBOL  AR_SAG_YAK=PORTC.3      ' 6.NUMARALI GİRİŞ
SYMBOL  AR_SAG_UZA=PORTC.2      ' 6.NUMARALI GİRİŞ
SYMBOL  SAG_AR_YAK=PORTD.0      ' 7.NUMARALI GİRİŞ
SYMBOL  SAG_AR_UZA=PORTD.1      ' 7.NUMARALI GİRİŞ
SYMBOL  SAG_ON_YAK=PORTD.2      ' 8.NUMARALI GİRİŞ
SYMBOL  SAG_ON_UZA=PORTD.3      ' 8.NUMARALI GİRİŞ
SYMBOL  SOL_OR_UZA=PORTC.4      ' 9.NUMARALI GİRİŞ
SYMBOL  SOL_OR_YAK=PORTC.5      ' 9.NUMARALI GİRİŞ
SYMBOL  SAG_OR_UZA=PORTC.7      '10.NUMARALI GİRİŞ
SYMBOL  SAG_OR_YAK=PORTC.6      '10.NUMARALI GİRİŞ

'*************************************************************
```

**Ek 1'in devamı**

```
'DEĞİŞKENLERİN BAŞLANGIÇ DEĞERLERİ
BILGI1=0
BILGI2=0
BILGI3=0
'************************************************************
ANADONGU:
   GOSUB    SENSAKTAR
GOTO ANADONGU
'SENSÖRLERDEN ALINAN VERİLER DEĞİŞKENLERE AKTARILIYOR.
SENSAKTAR:
   BILGI1.0=ON_SAG_YAK
   BILGI1.1=ON_SAG_UZA
   BILGI1.2=ON_SOL_YAK
   BILGI1.3=ON_SOL_UZA
   BILGI1.4=SOL_ON_YAK
   BILGI1.5=SOL_ON_UZA
   BILGI1.6=SOL_AR_YAK
   BILGI1.7=SOL_AR_UZA

   BILGI2.0=AR_SOL_YAK
   BILGI2.1=AR_SOL_UZA
   BILGI2.2=AR_SAG_YAK
   BILGI2.3=AR_SAG_UZA
   BILGI2.4=SAG_AR_YAK
   BILGI2.5=SAG_AR_UZA
   BILGI2.6=SAG_ON_YAK
   BILGI2.7=SAG_ON_UZA

   BILGI3.0=SOL_OR_YAK
   BILGI3.1=SOL_OR_UZA
```

**Ek 1'in devamı**

```
   BILGI3.2=SAG_OR_YAK
   BILGI3.3=SAG_OR_UZA
RETURN


DISABLE
KESME:
   IF  INTCON.1==1 THEN
      KIZIL_TEYIT=1
      GONDER:
         SEROUT KIZIL_VERI,N2400,[("K"),BILGI1,BILGI2,BILGI3]
         IF  KIZIL_KESME==0  THEN
            KIZIL_TEYIT=0
         ELSE
            GOTO    GONDER
         ENDIF
      INTCON.1=0
   ENDIF
RESUME
ENABLE
END

'*****************************************************************
'*  Name    : USARACI.BAS                                        *
'*  Author  : [İBRAHİM YAŞAR]                                    *
'*****************************************************************
'CONFIFURATION REGISTERIN AYARLAMALARI
@ DEVICE PROTECT_OFF,WDT_OFF,PWRT_ON,MCLR_OFF
@ DEVICE    INTRC_OSC_NOCLKOUT
INCLUDE "MODEDEFS.BAS"
```

**Ek 1'in devamı**

```
DEFINE HSER_RCSTA 90h     '(alma yazmacının Enable edilmesi)
DEFINE HSER_BAUD  2400    '(Haberleşme hızının belirlenmesi)
'************************************************************
'DENEME DEĞİŞKENLERİ
PORTA=0
PORTB=0
'STANDART REGISTER AYARLAMALARI
ON INTERRUPT GoTo KESME     'kesme oluşursa KESME adlı etikete git.
CMCON=%00000111 'COMPARATOR LERİ KAPATTIK.
OPTION_REG=%11000000 'PULL UP DİRENÇLERİNİ DISABLE ETTİK.
INTCON=%10010000          'Tüm Kesmeler aktif ve RB0/INT kesmesi aktif
PCON=%00001000 'KRISTAL 4 MHZ OLARAK AYARLANDI.
'************************************************************
'ANA MİKROKONTROLÖR İLE HABERLEŞMENİN SAĞLANMASI İÇİN
KULLANILAN
'DEĞİŞKENLER
BILGI1 VAR BYTE
BILGI1=0
SYMBOL  ANAUSARKESME=PORTB.0
SYMBOL  ANAUSARTEYIT=PORTB.2
SYMBOL  ANAUSARVERI=PORTB.3
TRISB.0=1
TRISB.2=0
TRISB.3=0
ANAUSARTEYIT=0
'************************************************************
ANADONGU:
   DISABLE
      HSERIN [BILGI1]
   ENABLE
```

**Ek 1'in devamı**

```
  IF  (BILGI1>=30) AND  (BILGI1<=200)
    BILGI1=BILGI1
  ELSE
    BILGI1=0
  ENDIF
GOTO ANADONGU

DISABLE
KESME:
  IF  INTCON.1==1 THEN
    ANAUSARTEYIT=1
    GONDER:
      SEROUT ANAUSARVERI,N2400,[("A"),BILGI1]
      IF  ANAUSARKESME==0  THEN
        ANAUSARTEYIT=0
      ELSE
        GOTO    GONDER
      ENDIF
    INTCON.1=0
  ENDIF
RESUME
ENABLE
END
```

**Ek 2.**

**Motor Sürücü Üzerindeki Mikrokontrolörlere Ait Program Kodu**

```
'*****************************************************************
'*  Name    : motorsur628-rev03.BAS                                  *
'*  Author  : [İbrahim YAŞAR]                                        *
'*****************************************************************
'CONFIFURATION REGISTERIN AYARLAMALARI
@ DEVICE PROTECT_OFF,WDT_OFF,PWRT_ON,MCLR_OFF
@ DEVICE    INTRC_OSC_NOCLKOUT
INCLUDE "MODEDEFS.BAS"
'*****************************************************************
'STANDART REGISTER AYARLAMALARI
CMCON=%00000111 'COMPARATOR LERİ KAPATTIK.
OPTION_REG=%10000000 'PULL UP DİRENÇLERİNİ DISABLE ETTİK.
INTCON=%00000000    'TÜM KESMELER KAPALI
TRISA=%00100000 '0,0,0,RST,IN-,IN+,DIS,HEN
TRISB=%11111111 'B PORTU MOTOR ÇALIŞMA GÜCÜNÜ BELİRLEYECEK
PCON=%00001000 'KRISTAL 4 MHZ OLARAK AYARLANDI.
'*****************************************************************
'DEĞİŞKENLER TANIMLANIYOR
SYMBOL  HEN=PORTA.3
SYMBOL  DIS=PORTA.2
SYMBOL  INAR=PORTA.0
SYMBOL  INEK=PORTA.1
'*****************************************************************
'DEĞİŞKENLERİN BAŞLANGIÇ DEĞERLERİ
PORTA=0
PORTB=0
'*****************************************************************
```

**Ek 2' nin devamı**

```
'ANA PROGRAM
PAUSE   100
ANA_PROGRAM:
INAR=1
PAUSE   100
WHILE   (PORTB.0=1) AND (PORTB.1=1)
   DIS=1
   PAUSE   PORTB
   DIS=0
   PAUSE   100-PORTB
WEND
INAR=0
PAUSE   100
INEK=1
PAUSE   100
WHILE   (PORTB.0=1) AND (PORTB.1=0)
   DIS=1
   PAUSE   PORTB
   DIS=0
   PAUSE   100-PORTB
WEND
INEK=0
PAUSE   100
WHILE   (PORTB.0=0)
   DIS=0
   INAR=0
   INEK=0
WEND
GOTO    ANA_PROGRAM:
End
```

**Ek 3.**

**Ultrasonik Sensör Üzerindeki Mikrokontrolörlere Ait Program Kodu**

```
include 16f628_4i
include jlib
include jmath
include h_rs232

-- Ultrasonic mesafe algılayıcısı JAL programı

disable_a_d_functions

var volatile byte TMR1L          at 0x0E
var volatile byte TMR1H          at 0x0F
var volatile byte T1CON          at 0x10
var volatile bit  PIR1_TMR1IF                 at PIR1          :        0
var volatile bit  T1CON_TMR1ON           at T1CON :     0

var volatile bit tx1                    is        pin_a1
var volatile bit tx1_dir                      is   pin_a1_direction
var volatile bit tx2                    is        pin_a2
var volatile bit tx2_dir                      is   pin_a2_direction
var volatile bit led                    is        pin_b0
var volatile bit led_dir                      is   pin_b0_direction
var volatile bit echo            is         pin_b7
var volatile bit echo_dir        is        pin_b7_direction

var byte  r1, r0

led       = low
```

**Ek 3' ün devamı**

```
led_dir   = output
tx1       = low
tx1_dir   = output
tx2       = low
tx2_dir   = output
echo_dir  = input

t1con = 0b00_00_0000

procedure burst is
var byte x = 4
assembler
local t1
 t1:
   movlw 0x02
   movwf port_a
   nop nop nop nop nop nop
   nop nop nop nop nop
   movlw 0x04
   movwf port_a
   nop nop nop
   nop nop nop nop
           decfsz x,f
           goto t1
   bcf tx1
   bcf tx2
end assembler
end procedure
function usr_oku ( byte out cm1, byte out cm0 )return bit is
```

**Ek 3' ün devamı**

```
 var byte x1, x0

  burst

  TMR1L  = 0 TMR1H = 0
  PIR1_TMR1IF  = false
  T1CON_TMR1ON = on

  while (echo == low) -- & (PIR1_TMR1IF  == false)
 loop   end loop

  x0 = TMR1L
  x1 = TMR1H
  T1CON_TMR1ON = off

  divide_16b ( x1, x0, 0, 58,cm1, cm0  )
  if PIR1_TMR1IF then cm1 = 0xff cm0 = 0xff end if
  return ! PIR1_TMR1IF
end function

serial_setup ( 24 )
forever loop
 if usr_oku ( r1, r0 ) then
   if r1 == 0 then
     if r0 <= 255 then
       h_asynch = r0
       led = high
     end if
--       if r0 <= 255  then led = high    end if
   end if
```

**Ek 3' ün devamı**

```
    if r1 > 0 then
      r0 = 0xff
      h_asynch = r0
    end if
  end if
  delay_100ms
  led = low
end loop
```

**Ek 4.**

**Kızılötesi Sensör Mikrokontrolörlerine Ait Program Kodu**

```
include 16f628_4i
include jlib
include comp

disable_comp

pin_a1_direction = output -- sol ır
pin_b2_direction = input  -- ir recv
pin_b3_direction = output -- sag ır
pin_b5_direction = output -- sol algılandı çıkışı
pin_b6_direction = output -- sag algılandı çıkışı
pin_b4_direction = output -- sol algılandı Ledı
pin_a0_direction = output -- sag algılandı Ledı
var byte sol,sag , i

procedure module_1ms_sol ( byte in x ) is
 for x loop
   for 40 loop
     assembler
       bsf port_a , 1
       nop nop nop nop nop nop
       nop nop nop nop nop
       bcf port_a , 1
       nop nop nop nop nop
       nop nop nop nop
     end assembler
   end loop
```

**Ek 4' ün devamı**

```
 end loop
end procedure

procedure module_1ms_sag ( byte in x ) is
 for x loop
   for 40 loop
     assembler
       bsf port_b , 3
       nop nop nop nop nop nop
       nop nop nop nop nop
       bcf port_b , 3
       nop nop nop nop nop
       nop nop nop nop
     end assembler
   end loop
 end loop
end procedure

procedure sol_say is
 i = 0
  for 10 loop
    module_1ms_sol ( 2 )
    if pin_b2 == low then i = i + 1 end if
    module_1ms_sol ( 1 )
    delay_10ms ( 2 )
    if pin_b2 == high then i = i + 1 end if
    delay_10ms ( 1 )
  end loop
sol = i
end procedure
```

**Ek 4' ün devamı**

```
procedure sag_say is
 i = 0
  for 10 loop
    module_1ms_sag ( 2 )
    if pin_b2 == low then i = i + 1 end if
    module_1ms_sag ( 1 )
    delay_10ms ( 2 )
    if pin_b2 == high then i = i + 1 end if
    delay_10ms ( 1 )
  end loop
sag = i
end procedure

pin_b4 = low  pin_a0 = high delay_500ms
pin_b4 = high pin_a0 = low  delay_500ms
pin_b4 = low  pin_a0 = low
pin_a1 = low
pin_b3 = low
forever loop
  sag_say
  sol_say
  if sag > 18 then pin_b5 = high pin_b4 = high else pin_b5 = low pin_b4 = low end if
  if sol > 18 then pin_b6 = high pin_a0 = high else pin_b6 = low pin_a0 = low end if
end loop
```

# ÖZGEÇMİŞ

İstanbul Bakırköy' de 1982 yılında doğdu. İlk ve orta öğrenimini Bağcılar Füruzan Sadıkoğlu İ.Ö.O. tamamladı. 1996 da Kuleli Askeri Lisesini kazandı ve 2000 yılında buradan mezun oldu. Kara Harp Okulu 1. sınıftan terk etti ve Karadeniz teknik Üniversitesi Elektrik Elektronik Mühendisliğini Kazandı. 2006 yılında üniversiteden mezun oldu. Mezun olduktan sonra bir süre kojenerasyon sistemleri üzerine özel bir firmada çalışmıştır. Şu anda Bağcılar belediyesi Fen İşleri Müdürlüğünde Kontrol Mühendisi olan İbrahim YAŞAR aynı zamanda Karadeniz Teknik Üniversitesinde yüksek lisansa devam etmekte olup, iyi derecede İngilizce bilmektedir.