PALS GENİŞLİK MODÜLASYONU KULLANARAK ÜÇ FAZLI BİR İNDÜKSİYON MOTORUNUN HIZ KONTROLU

SALİH GÖRGÜNOĞLU

YÜKSEK LİSANS TEZİ
ELEKTRONİK BİLGİSAYAR EĞİTİMİ

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Şubat 2000
ANKARA

Salih GÖRGÜNOĞLU tarafından hazırlanan PALS GENİŞLİK MODÜLASYONU KULLANARAK ÜÇ FAZLI İNDÜKSİYON MOTORUNUN HIZ KONTROLÜ adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

Doç. Dr. Ömer Faruk BAY
Tez Yöneticisi

Bu çalışma, jürimiz tarafından Elektronik Bilgisayar Eğitimi Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir.

Başkan : Prof. Dr. İnan Güler

Üye : Doç. Dr. Ömer Faruk BAY

Üye : Doç. Dr. İlhami Colak

Üye :

Üye :

Bu tez, Gazi Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygundur.

İÇİNDEKİLER

PALS GENİŞLİK MODÜLASYONU KULLANARAK ÜÇ FAZLI BİR İNDÜKSİYON MOTORUNUN HIZ KONTROLU

Yüksek Lisans Tezi

Salih GÖRGÜNOĞLU

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Şubat 2000

ÖZET

İndüksiyon motorları endüstride kullanılan en popüler makinalardır. DC motorlarla karşılaştırıldıklarında, indüksiyon motorları daha verimli, daha küçük, daha hafif, daha az atalet momentine sahiptir ve daha yüksek hızlarda çalışabilirler. Bununla birlikte, hız kontrolleri daha zordur.

Bu çalışmada, üç fazlı bir indüksiyon motoru hız kontrol cihazı, Sinüsoidal Pals Genişlik Modülasyonu metodu kullanılarak tasarımı yapılmakta ve gerçekleştirilmektedir. Bu metod indüksiyon motorlarının kayıplarını azaltmada oldukça etkin bir şekilde kullanılmaktadır.

Bilim Kodu : 225.10.01
Anahtar Kelimeler : İndüksiyon motoru, Pals Genişlik Modülasyonu(PWM), Hız Kontrolu
Sayfa Adedi : 124
Tez Yöneticisi : Doç. Dr. Ömer Faruk BAY

THE SPEED CONTROL OF THREE PHASE INDUCTION MOTORS BY USING PULSE WIDTH MODULATION METHOD

(M.Sc. Thesis)

Salih GÖRGÜNOĞLU

GAZİ UNIVERSITY

INSTITUTE OF SCIENCE AND TECHNOLOGY

February 2000

ABSTRACT

The induction motors are the most populer machines in industry. When they are compared with DC motors, inductions motors are more efficient, smaller, lighter, have less inertia and can be run higher speeds. However, Its speed control is more difficult.

In this study, a speed control device of three phase induction motor is designed and implemented by using Sinusoidal Pulse Width Modulation control method. This rather effective to reduce the loss of induction motors.

Science code : 225.10.01

Key words : Induction Motors, Pulse width Modulation(PWM), Speed Control

Page Number :124

Adviser : Assoc.Prof. Dr. Ömer Faruk BAY

TEŞEKKÜR

Bu tezin hazırlanması sırasında bana her zaman yardımcı olan ve yönlendiren çok kıymetli hocam sayın Doç. Dr. Ö. Faruk BAY'a, özellikle tezin uygulama devrelerinin yapımında büyük desteğini gördüğüm sevgili ağabeyim İbrahim GÖRGÜNOĞLU ve kardeşim Hüseyin GÖRGÜNOĞLU'na ve değerli arkadaşım Sabri KOÇER'e teşekkür ederim.

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılan bazı simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	Açıklama
e	Volt olarak indüklenen emk.
N	Sarım sayısı
ϕ	Manyetik akı
f	Frekans
f_c	Taşıyıcı dalga frekansı
f_r	Referans dalga frekansı
V_c	Taşıyıcı dalga genliği
V_r	Referans dalga genliği
M	Modülasyon indeksi
m	Taşıyıcı oranı
ω	Açısal frekans
n_s	Senkron devir sayısı
n_r	Rotor devir sayısı
s	Kayma
I_r	Kilitli rotor faz akımı
E_{kr}	Kilitli rotor faz EMK değeri
X_{kr}	Kilitli rotor faz reaktansı
Z_r	Rotor empedansı
P_{cu}	Faz başına rotor bakır kayıpları
P_a	Rotor milinden alınan güç
Z_e	Eş değer toplam empedans

I_s	Motordan çekilen toplam akım
M_d	Motorun döndürme momenti
R_r	Rotor direnci
R_s	Stator direnci
a	Motor dönüştürme oranı
T	Peryot
I_0	Motorun boş çalışmada çektiği akım
R_0	Motor sargılarının boş çalışmadaki etkin direnci
X_0	Motor sargılarının boş çalışmadaki reaktansı
p	Çift kutup sayısı

Kısaltmalar

PWM	Pals Genişlik Modülasyonu
EMK	Elektro Motor Kuvvet
AC	Alternatif akım
DC	Doğru akım
LCD	Sıvı kristal ekran
RAM	Rasgele Erişimli Bellek
ROM	Sadece okunabilir Bellek
EEPROM	Elektrikle silinip yazılabilen ROM
MOSFET	Alan etkili transistör

1. GİRİŞ

Günümüzde fabrikalarda, evlerde, bürolarda en çok kullanılan motorlardan birisi de indüksiyon motorlarıdır. İndüksiyon motorları asenkron motorlar olarak da bilinmektedir. İndüksiyon motorlar bir fazlı, iki fazlı ve üç fazlı olarak yapılmaktadırlar. Sanayi sektöründe kullanılan DC motorlarla kıyaslandığında, maliyetlerinin düşük olması, az arıza yapması, bakım ve tamirinin kolay olması, daha verimli çalışması, daha küçük boyutlarda ve hafif olması, daha yüksek hızlarda çalıştırılabilmeleri, devirlerinin yükle az değişmesi gibi nedenlerle indüksiyon motorlar tercih edilmektedirler. DC motorların tek bir üstün tarafı hızlarının kolay ayarlanabilmesidir. İndüksiyon motorların hızları ise daha karmaşık devrelerle ayarlanabilmektedir. Gelişen yarı iletken ve mikrodenetleyici teknolojisi ile birlikte indüksiyon motorun hız ayarı için çeşitli ve ekonomik yöntemler geliştirilmektedir.

Bu çalışmanın ikinci bölümünde tasarlanan devrenin daha iyi anlaşılabilmesi için indüksiyon motoru genel olarak tanıtılmaktadır. İndüksiyon motorun yapısı, çalışma prensibi, motor eşdeğer devresi ile ilgili bağıntılara yer verilmektedir.

Üçüncü bölümde genel olarak indüksiyon motorunun hız kontrol yöntemleri anlatılmakta ve bu yöntemlerle ilgili moment hız karakteristikleri açıklanmaktadır.

Dördüncü bölümde frekans çeviriciler tanıtılmaktadır. Daha sonra altı adımlı inverter incelenmekte, inverter kontrol yöntemleri ve üç fazlı PWM inverterin çalışması izah edilmektedir.

Beşinci bölümde devre tasarımında kullanılan mikrodenetleyicinin yapısı, özellikleri ve kısaca programlanması temel olarak anlatılmaktadır.

Altıncı bölümde ise hız kontrol devresinin gerçekleştirilmesi sırası ile anlatılmaktadır. Tuş takımı devresi, LCD ekran kontrolü, MOSFET sürücü devresi, Snubber devresi ve aşırı akım koruma devreleri açıklanmaktadır.

Yedinci bölümde Sinüsoidal PWM sinyallerinin nasıl elde edildiği gösterilmektedir. PWM anahtarlama sinyalleri elde edebilmek için öncelikle Delphi programlama dili ile bir simülatör programı yapılmakta ve bu simülatör ile her frekans için mikrodenetleyicide kullanılacak frekans tablosu değerleri üretilmesi sağlanmaktadır. Daha sonra kontrol yazılımının gerçekleştirilmesine ilişkin program akış şemaları verilmektedir.

Sekizinci ve dokuzuncu bölümlerde ise simülasyon ve deney sonuçları verilmekte ve bazı önerilerde bulunulmaktadır.

2. İNDÜKSİYON MOTORUN YAPISI

İndüksiyon motorları alternatif akımla çalışan makinalardır. Genellikle sabit duran statoru ve dönen rotoru bulunmaktadır. Statorlarında bir, iki yada üç fazlı sargılar bulunabilmektedir. Stator sargıları adı verilen bu sargılara alternatif gerilimler uygulanır. Hızları yük ile çok az değişen motorlardandır. Dönen rotorda bir, üç yada daha fazla sargı bulunur. Rotor sargıları adı verilen bu sargılarda gerilimler indüksiyon yolu ile oluşurlar. Özel amaçlı kullanımlar dışında rotor sargısına dış bir kaynaktan gerilim uygulanmaz [1].

2.1 Stator

Stator indüksiyon motorun temel hareketsiz parçasıdır. Üzerinde döner manyetik alanın oluşmasını sağlayan sargılar yer alır. İndüksiyon motorları bir fazlı ve üç fazlı olmak üzere iki çeşit olarak yapılırlar. Bir fazlı motorda en az bir ve 3 fazlı motorda ise en az 3 sargı yer alır. 3 fazlı indüksiyon motorunda sargılar döner alanın oluşmasını sağlayacak şekilde yerleştirilirler. Her bir sargıya bir birinden 120 derece faz farklı bir AC sinyal uygulanır [1].

2.2 Rotor

Rotor indüksiyon motorunun hareketli dönen parçasıdır. İki şekilde yapılırlar. Bunlar :

1. *Sincap kafesli(kısa devreli) rotorlar*
2. *Sargılı rotorlar.*

Daha çok sincap kafesli rotorlar kullanılır. Çünkü bunların yapımı daha kolay ve maliyetleri daha düşüktür. Rotoru sargılı olanlar rotorunda sargı bulundururlar. Bu sargılar birbirinden 120 derece faz farklı olarak rotor

oluklarına yerleştirilir. Rotor miline yalıtılarak yerleştirilmiş bulunan bileziklere bu sargılar üçgen veya yıldız olarak bağlanır. Bu sargılar bileziklere bağlanan reostalar yardımı ile devreye sokulur. Rotoru sargılı indüksiyon motorların maliyetleri biraz yüksektir ama bu motorların kalkınma momentleri yüksek olduğundan yüklü olarak kalkınması gereken asansör, vinç gibi aletlerde kullanılırlar [1].

2.3 İndüksiyon Motorun Çalışma Prensibi

Bir indüksiyon motoru kısaca şu şekilde çalışır. Stator sargılarına zamanla değişen bir voltaj uygulandığında bu değişken voltaj zamanla değişen bir manyetik alan oluşturur. Bu manyetik alan rotor sargılarını keser ve rotor sargılarında bir akım oluşturur. Bu akım da rotor sargılarında bir manyetik alan oluşturur. Stator ve rotor manyetik alanlarının karşılıklı etkileşimi ile itme ve çekme kuvvetleri meydana gelir. Bu kuvvetlerin etkisi ile rotor dönmeye başlar [1].

2.4 Stator Ve Rotor Sargılarında İndüklenen emk

Stator sargılarına uygulanan 3 fazlı alternatif akım bir döner alan meydana getirir. Bu döner alan yine sinüsoidaldir. Faraday'ın elektromagnetik indüksiyon kuralına göre bu indüklenen EMK şu şekilde ifade edilir. Herhangi bir iletkenden bir saniyede bir weberlik manyetik akı geçiyorsa bu iletkende 1 volt gerilim indüklenir ve aşağıdaki şekilde ifade edilir..

$$e = -N\frac{d\phi}{dt} \quad \text{(Volt)} \qquad (2.1)$$

N: Sarım sayısı

ϕ :Toplam manyetik akı (Weber)

e : Volt olarak indüklenen EMK

Stator sargılarına uygulanan sinüsoidal voltaj nedeniyle oluşan Manyetik akı da sinüsoidaldir. Buna göre,

$$\phi = \phi_m \mathrm{Sin}\omega t \tag{2.2}$$

ve stator ve rotor sargılarında indüklenen gerilimler ise şöyledir.

$$e_1 = -\omega N_1 \phi_m \mathrm{Cos}\omega t = \omega N_1 \phi_m \mathrm{Sin}(\omega t - \frac{\pi}{2}) \tag{2.3}$$

$$e_2 = -\omega N_2 \phi_m \mathrm{Cos}\omega t = \omega N_2 \phi_m \mathrm{Sin}(\omega t - \frac{\pi}{2}) \tag{2.4}$$

Görüldüğü gibi indüklenen emk kendisini oluşturan manyetik akının 90 derece gerisindedir.

İndüklenen emk nın maksimum değeri şöyledir.

$$E_{1m} = \omega N_1 \phi_m = 2\pi f N_1 \phi_m \tag{2.5}$$

İndüklenen EMK nın etkin veya rms değeri şu şekilde ifade edilir[2].

$$E_1 = V_1 = \frac{2\pi}{\sqrt{2}} f N_1 \phi = 4.44 f N_1 \phi_m \tag{2.6}$$

2.5 Rotorun Devir Sayısı ve Kayma

Stator sargılarında meydana gelen döner alanın devir sayısına senkron devir (n_s) denir. Rotorun devir sayısı (n_r) arttıkça döner alanın rotor çubuklarını kesmesi azalacağından, rotor çubuklarında indüklenen EMK ler ve kısadevre çubuklarından geçen indüksiyon akımları azalır dolayısı ile rotoru döndüren moment de azalır. Rotorun devir sayısında artış olmaz. Rotor boşta çalışırken, rotorun devir sayısı senkron devir sayısına yaklaşır. Döner alanın devir sayısı (senkron devir) ile rotor devir sayısı arasındaki farka rotorun kayması adı verilir. Kayma **s** harfi ile gösterilir ve senkron devir sayısının yüzdesi ile ifade edilir.

$$s = \frac{n_s - n_r}{n_s}.100 \tag{2.7}$$

$$n_s = \frac{60.f}{p} \tag{2.8}$$

$$n_r = n_s(1-s) \tag{2.9}$$

Görüldüğü gibi rotorun devir sayısı hiçbir zaman döner alanın devir sayısına eşit olamaz. Rotor senkron devirden daha az bir devirle döner.

Stator sargılarındaki döner alanın devir sayısı n_s şu formülle ifade edilir.

$$f_r = s.f \tag{2.10}$$

Rotorun frekansı ve rotorda oluşan emk kaymaya bağlı olarak

$$E_r = s.E_{kr} \tag{2.11}$$

Şeklinde ifade edilir [1].

2.6 İndüksiyon Motorun Eşdeğer Devresi

İndüksiyon motorun güç, döndürme momenti, verim, güç katsayısı vb. hesaplamaları yaparken motorun eşdeğer devresini bilmek büyük kolaylıklar sağlar.

2.6.1 Rotorun eşdeğer devresi

Rotorun eşdeğer devresi çizilirken indüksiyon motorun kilitli rotor deneyinden faydalanılır. Rotor kilitlenerek statora anma geriliminin %3 ile %5i arası bir gerilim uygulanır. Bu durumda rotorda indüklenen gerilime kilitli rotor gerilimi denir. Rotor kilitli iken rotordan geçen akım ise kilitli rotor akımı olarak ifade edilir.

Kilitli rotor faz akımı

$$I_r = \frac{E_{kr}}{\sqrt{R_r^2 + X_{kr}^{\ 2}}} \tag{2.12}$$

I_r :Kilitli rotor faz akımı

R_{kr} : Kilitli rotor faz emk değeri

X_{kr} : Kiliti rotor faz reaktansı

Motor çalışırken rotorun faz akımı

$$I_r = \frac{sE_{kr}}{\sqrt{R_r^2 + (sX_{kr})^2}} \qquad (2.13)$$

Motor çalışırken rotorun faz akımı

Şeklinde ifade edilir. Pay ve payda (s) ile bölündüğünde rotor akımı

$$I_r = \frac{E_{kr}}{\sqrt{(R_r/s)^2 + (sX_{kr})^2}} \qquad (2.14)$$

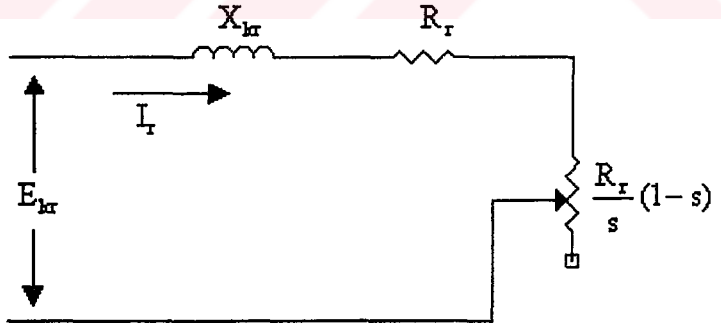
Şekline dönüşür. Rotor empedansı ise

$$Z_r = \sqrt{(R_r/s)^2 + X_{kr}^2} \qquad (2.15)$$

Şeklinde ifade edilir. Rotor eşdeğer devresindeki (R_r/s) direnci kayma ile değişen bir omik dirençtir. (R_r/s) direncine (R_r) rotor direncini ekleyip çıkaralım.

$$\frac{R_r}{s} = \frac{R_r}{s}(1-s) + R_r \qquad (2.16)$$

Rotor eşdeğer devresi Şekil 2.1'deki hale dönüşür.



Şekil 2.1 Rotor eşdeğer devresi

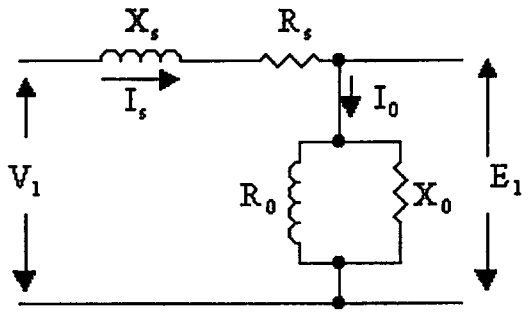
$$P_{cu} = I_r^2 . R_r \qquad (2.17)$$

Faz başına rotor bakır kayıplarını verir. Faz başına rotor milinden alınan mekanik gücü ise "Eş. 2.18" ile ifade edilebilir [1].

$$P_a = [I_r^2 . R_r (1-s)/s] \qquad (2.18)$$

2.6.2 Statorun eşdeğer devresi

Stator sargılarının eş değer devresi Şekil 2.2'deki gibi gösterilebilir.



Şekil 2.2 Stator eşdeğer devresi

Burada: R_0

R_s : :Üç fazlı staorda bir fazın etkin direnci

X_s: Bir fazın kaçak reaktansı

I_0: Motorun boş çalışmada çektiği akım

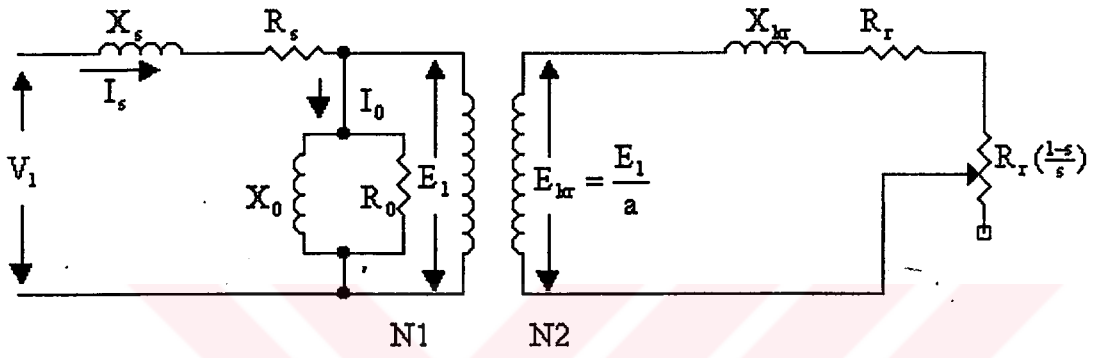
R_0:Motor sargılarının boş çalışmadaki etkin direnci

X_0: Motor sargılarının boş çalışmadaki reaktansı

Motor boşta çalışırken küçük bir I_0 akımı çeker. Bu akım Motorun statorda meydana gelen demir kayıplarını ve rotordaki sürtünme kayıplarını karşılar. Motor boş da çalışırken kayma (s) çok küçüktür (Yaklaşık s=%1) ve rotor yaklaşık olarak senkron devirle döner. Motor yükte çalışırken kayma artar ve statordan daha fazla bir akım (I_s) çeker. Statorun yüklü çalışmada çektiği I_s akımı I_0 akımından çok büyüktür [1].

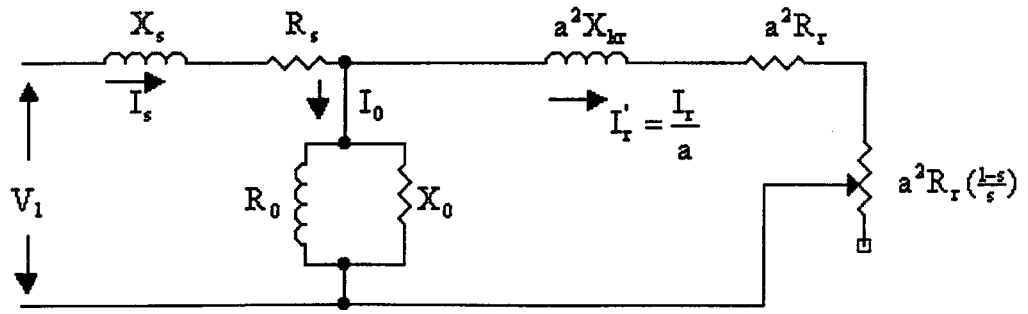
2.6.3 Motorun eşdeğer devresi

Motorun stator sarımlarındaki bir fazın etkin direnci R_s, kaçak reaktansı X_s ve dönüştürme oranı (a) olsun. Bu durumda rotor eşdeğer devresi stator tarafına göre dönüştürme yapılarak Şekil 2.3'deki gibi olur.

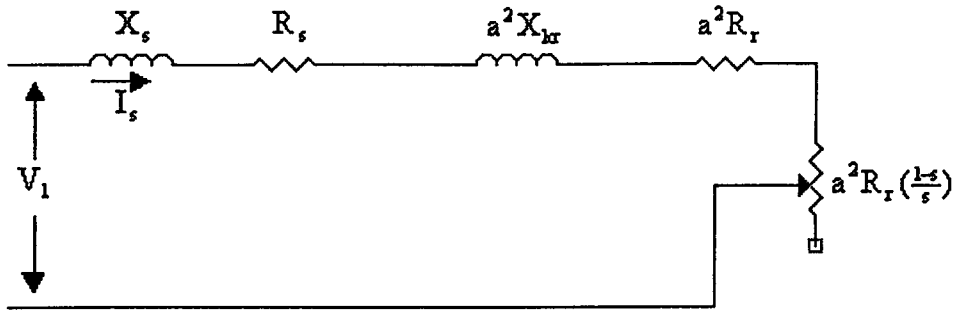


Şekil 2.3 Stator ve rotor birleşik eş değer devresi

Rotor eşdeğer devresi statora aktarılarak çizildiğinde motorun tüm eşdeğer devresi (Şekil 2.4) elde edilmiş olur. Hesaplamalarda boş çalışma akımı küçük bir değer olduğundan ihmal edilebilir. Bu durumda motorun eşdeğer devresi Şekil 2.5'deki gibi olur.



Şekil 2.4 Rotor devresinin statora aktarılmış hali



Şekil 2.5 I_0 akımı ihmal edildiğinde eşdeğer devre

2.7 Motor Milinden Alınan Güç ve Döndürme Momenti

Motorun eşdeğer devresine göre bir faz için motor milinden alınan güç

$$P_a = I_s^2.a^2.R_r(\frac{1-s}{s}) \tag{2.19}$$

Olarak hesaplanabilir. Toplam alınan güç ise

$$P_a = 3.I_s^2.a^2.R_r(\frac{1-s}{s}) \tag{2.20}$$

Olarak bulunur. Motorun eş değer devresinde toplam eşdeğer empedans

$$Z_e = \sqrt{\left[(R_s + a^2.R_r + a^2.R_r.(\frac{1-s}{s}))^2 + (X_s + a^2.X_{kr})^2\right]} \tag{2.21}$$

İle ifade edilir. Motorun I_s akımı ise

$$I_s = \frac{V_1}{Z_e} = \frac{V_1}{\sqrt{\left[(R_s + a^2.R_r + a^2.R_r.(\frac{1-s}{s}))^2 + (X_s + a^2.X_{kr})^2\right]}} \tag{2.22}$$

İle ifade edilir. Motordan alınan güç P_a(Watt) rotorun açısal hızına bölünürse Newton-metre olarak döndürme momentini verir.

$$M_d = \frac{P_a}{\omega_r} = \frac{P_a}{2.\pi.n_r} \quad (\text{Newton} - \text{m/faz}) \tag{2.23}$$

$$M_d = \frac{975.P_a(\text{Kw})}{n_r} = \frac{975.P_a(\text{Kw})}{.n_s(1-s)} \quad (\text{Kilogram} - \text{metre/faz}) \tag{2.24}$$

Kilogram metre olarak döndürme momenti şeklinde ifade edilir.

Motordan alınan güç

$$P_a = I_s^2.a^2.R_r(\frac{1-s}{s})$$

$$P_a = \frac{V_1^2}{\left[(R_s + a^2.R_r + a^2.R_r.(\frac{1-s}{s}))^2 + (X_s + a^2.X_{kr})^2\right]}.a^2.R_r.(\frac{1-s}{s}) \quad (2.25)$$

Olur. Formül yeniden düzenlenirse döndürme momenti

$$M_d = \frac{975.P_a(Kw)}{.n_s(1-s)}$$

$$M_d = \frac{975.a^2.V_1^2.R_r}{\left[(R_s + a^2.R_r + a^2.R_r.(\frac{1-s}{s}))^2 + (X_s + a^2.X_{kr})^2\right].s.n_s}.(Kg-m/Faz) \quad (2.26)$$

Olarak bulunabilir. Formülde n_s yerine frekansa bağlı değer yazılırsa

$$M_d = \frac{975.a^2.V_1^2.R_r.P}{\left[(R_s + a^2.R_r + a^2.R_r.(\frac{1-s}{s}))^2 + (X_s + a^2.X_{kr})^2\right].60.s.f}(Kg-m/Faz) \quad (2.27)$$

Bulunur [1].

3. İNDÜKSİYON MOTORU HIZ KONTROL YÖNTEMLERİ

İndüksiyon motorlarının devir sayıları doğru akım motorlarında olduğu gibi kolayca değiştirilemez. Doğru akım şönt motorun devir sayısı endüktör devresine seri bağlanan reosta ile çok geniş sınırlar içinde değiştirilebilir. İndüksiyon motorlarda devir sayısı ayarı kademeli ve küçük sınırlar altında yapılabilir. Endüstride devir sayısının geniş sınırlar içinde yapılması istenen yerlerde indüksiyon motor yerine doğru akım motorları tercih edilir. Buna karşılık özel bazı tasarımlarla bu sınırlama aşılabilir. Özellikle mikroişlemcili sistemler kullanılarak devir sayısını istenen sınırlar içinde çok kademeli olarak ayarlamak mümkündür.

Bir indüksiyon motorun hızı aşağıdaki yöntemlerle ayarlanabilir.

1. *Kutup sayısını değiştirerek*
2. *Statora uygulanan gerilimin genliğini değiştirerek*
3. *Rotor direncinin değiştirilmesi ile*
4. *Statora uygulanan gerilimin frekansını değiştirerek*

Bir indüksiyon motorunun temel hız formülü aşağıdaki şekildedir [3].

$$n_s = \frac{60.f}{p} \qquad (3.1)$$

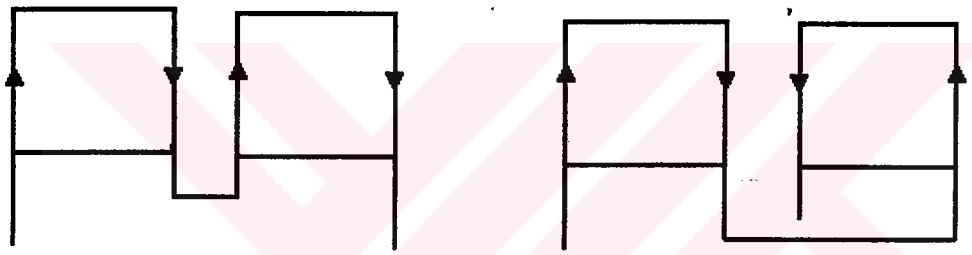
n: Devir sayısı

f: Statora uygulanan gerilimin frekansı

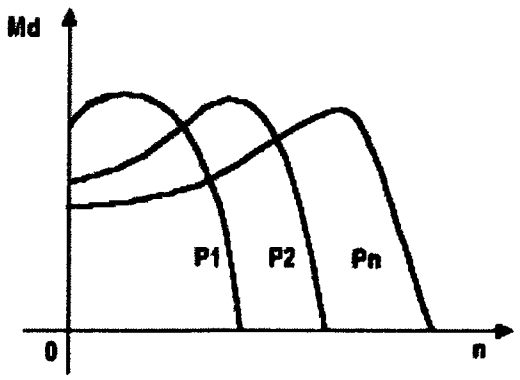
p: Çift kutup sayısı

3.1 Kutup Sayısının Değiştirilmesi İle Hız Ayarı

Kutup sayısını değiştirerek hız ayarı ancak birkaç çeşit devir ayarı yapma imkanı sağlar. Sabit frekans ve gerilimli şebekede kutup sayısını değiştirmekle hız ayarı yapılır. İki ayrı devir için bir sargı, üç ve dört ayrı devir için iki sargı statora yerleştirilir. Sargıların birbirlerine bağlanış şekilleri değiştirilerek kutup sayısı değiştirilir. Şekil 3.1'de sargılarda kutup sayısının nasıl değiştirildiği ve Şekil 3.2'de ise moment hız karakteristiği görülmektedir. Kutup sayısını değiştirmekle ancak birkaç kademe devir ayarı yapılabilir [4].



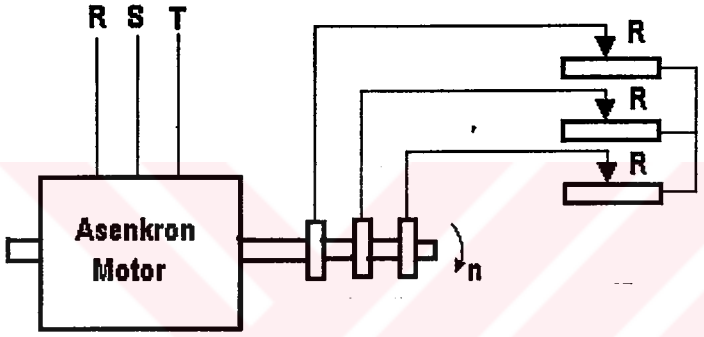
Şekil 3.1 Dört ve iki kutuplu sargılar



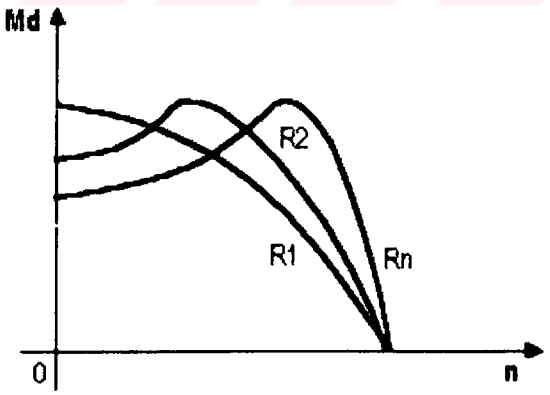
Şekil 3.2 Kutup sayısının değiştirilmesi ile elde edilen moment hız krakteristiği

3.2 Rotor Direncinin Değiştirilmesi İle Hız Ayarı

Rotoru sargılı indüksiyon motorlarda rotor sargılarına bilezikler yardımı ile direnler seri olarak bağlanır (Şekil 3.3). Bu dirençler kaymayı artırarak devir sayısının düşmesine neden olurlar . Bu metodla motora verilen enerjinin bir kısmı dirençler üzerinde kaybolacağından verim düşüktür. Büyük güçlü motorlarda kullanılmaz. Şekil 3.4 de bu metodla yapılan hız ayarının moment hız karakteristiği görülmektedir [5].



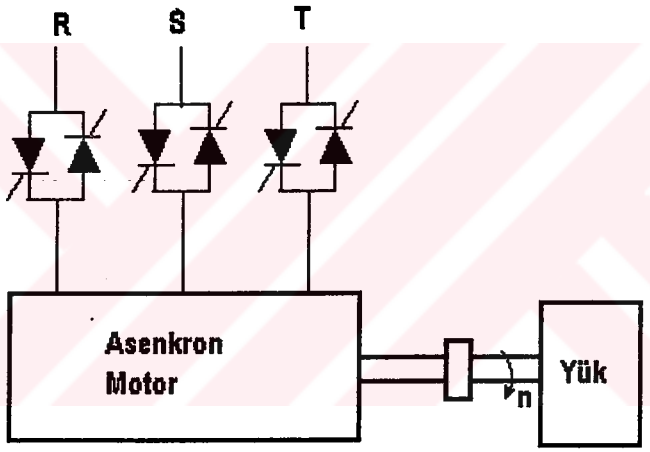
Şekil 3.3 Rotoruna değişken direnç bağlanarak hız ayarı yapılan bilezikli indüksiyon motor



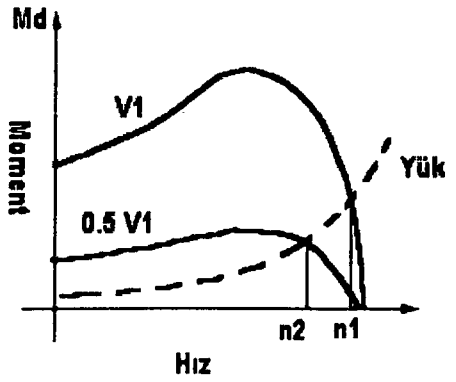
Şekil 3.4 Rotoruna değişken direnç bağlanarak hız ayarı yapılan bilezikli indüksiyon motor moment hız karakteristiği

3.3 Statora Uygulanan Gerilimin Genliği Değiştirilerek Hız Ayarı

İndüksiyon motorlarda moment gerilimin karesi ile doğru orantılıdır. Bunu daha önce ifade ettiğimiz moment formülünden "Eş. 2.26". görebiliriz.
Buradan da açıkça görüldüğü gibi uygulanan gerilimin etkin değeri değiştirilerek devir ayarı yapılır. Gerilimin etkin değerinin değiştirilmesi momentin değişmesine neden olur. Momentin değişmesi ise motor devrinin değişmesine neden olacaktır. Boş çalışmada bu yöntemle devir ayarı yapılmaz. Şekil 3.5'de bu yöntemle tristör kullanarak yapılan hız ayarının prensib şeması ve Şekil 3.6'da ise moment hız karakteristiği verilmiştir [5].



Şekil 3.5 Statora uygulanan gerilimin değiştirilmesi



Şekil 3.6 Stator voltajı değiştirilerek elde edilen moment hız karakteristiği

3.4 Uygulanan Gerilimin Frekansını Değiştirerek Hız Ayarı

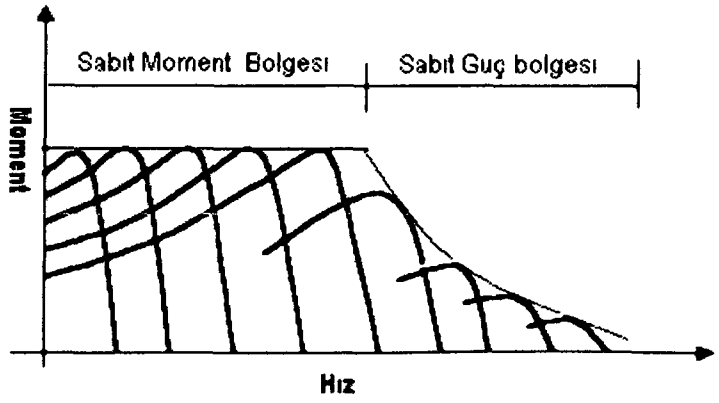
Stator voltajının frekansını değiştirmek çok fazlı indüksiyon motorların hız kontrolü için en iyi yöntemdir. Bu metotla statora uygulanan gerilimin frekansı değiştirilir. Devir sayısı da frekansla doğru orantılı olduğu için artar veya azalır. Devir sayısı artarken belirli bir frekanstan sonra motorun döndürme momentinde bir azalma olur. Düzgün bir devir ayarı için momentin de sabit kalması istenir. Sabit bir moment elde edebilmek için stator ile rotor arasında bulunan hava aralığındaki manyetik akı yoğunluğu sabit olmalıdır. Çünkü momentin rotor akımı ve manyetik akı yoğunluğuna göre formülü şu şekildedir.

$$M_d = K.\phi.I_r.\cos\theta \qquad (3.1)$$

Manyetik akının ve dolayısı ile momentin sabit kalması için Stator voltajının .Frekkansa oranı V/F sabit olmalıdır. Çünkü Rotor sargılarında indüklenen EMK değeri yaklaşık olarak statora uygulanan gerilime eşittir.

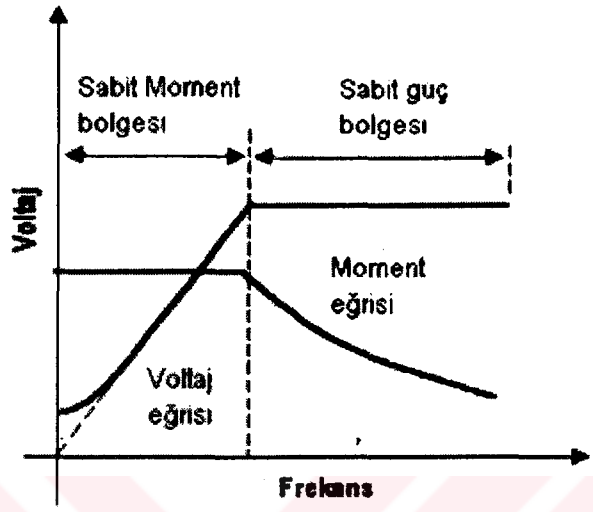
$$E_r = sE_1 = sV_1 = 4{,}44.f.\phi.N \qquad (3.2)$$

Manyetik akı değeri yalnız bırakılırsa, manyetik akının sabit kalması için V/F oranının sabit kalması gerektiği görülebilir. Bunun anlamı devir sayısını



Şekil 3.7 : Frekans değiştirilerek yapılan hız ayarı moment hız karakteristiği

değiştirirken momentin de sabit kalması için frekansla beraber stator voltajının da değiştirilmesi gerektiğidir. Şekil 3.7 Moment hız karakteristiği ve Şekil 3.8 de voltaj frekans eğrisi görülmektedir [6].



Şekil 3.8 İndüksiyon motorun voltaj frekans eğrisi

4. FREKANS ÇEVİRİCİLER

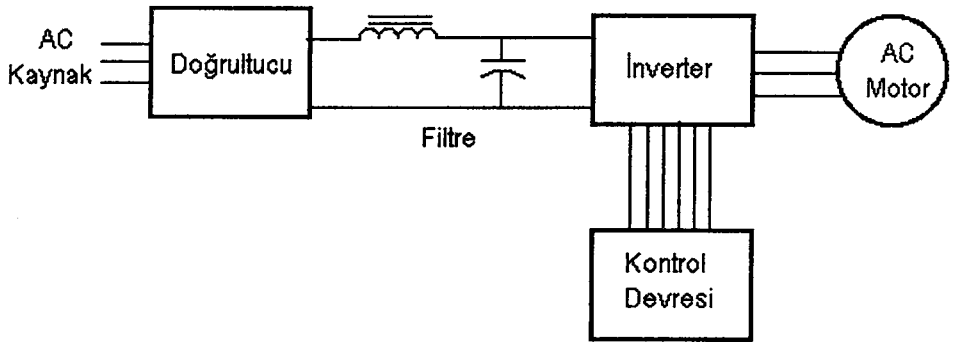
Mikroişlemci kullanarak üç fazlı indüksiyon motorun hız kontrolünü yapmak için frekans çeviricilerden yararlanılır. Frekans çeviriciler yardımı ile statora uygulanan voltajın genliği ve frekansı değiştirilerek devir ayarı yapılır. İndüksiyon motorun devir sayısı ayarında kullanılan frekans çevirici devre dört bölümden oluşur.

1. *Doğrultucu devre*
2. *Filtre devresi*
3. *Kontrol devresi*
4. *Evirici(İnverter) devresi*

Doğrultucu bölümü: Şebekenin üç fazlı AC sinyalini doğrultarak DC sinyale çevirir.

Filtre: Doğrultucu bölümden elde edilen sinyali filtre ederek şebeke frekansından soyutlar.

Kontrol devresi : Frekans çeviricinin çalışması için gerekli sinyallerin üretildiği ve kontrol işleminin yapıldığı bölümdür.



Şekil 4.1 Frekans çeviricinin blok şeması

Evirici devre : Ara devreden gelen DC sinyalden istenen frekans ve genlik değerinde AC sinyalin üretildiği kısımdır. Elde edilen AC sinyal kontrol devresinden alınan anahtarlama sinyallerine göre PWM veya kare şeklinde olabilir [4].

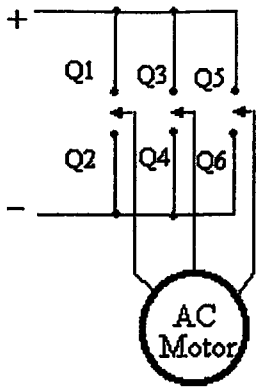
4.1 Altı Adımlı İnverter

Üç fazlı indüksiyon motora AC sinyal üreten inverter altı adımlı inverter olarak yapılır. Altı adımlı bir inverter kullanarak DC voltajdan AC voltaj nasıl elde edildiğini açıklamak için Şekil 4.2'yi ve ve Çizelge 4.1'i inceleyelim. Bilindiği gibi üç fazlı sinyalin her bir fazı bir birinden 120 derece faz farklıdır. Bu yüzden her bir anahtar 60 dereceklik açılarla ve bir birlerine göre 120 derece faz farklı olarak altı adım da açılıp kapatılarak bir peryot tamamlanır. Bunun için her bir zaman aralığında altı adet anahtarlama MOSFET lerine kontrol sinyalleri Çizelge 4.1'deki gibi gönderilmelidir.

Çizelge 4.1 Anahtarlama elemanlarının durumları

	Zaman aralıkları					
	T1	T2	T3	T4	T5	T6
Q1	1	1	1	0	0	0
Q2	0	0	0	1	1	1
Q3	0	0	1	1	1	0
Q4	1	1	0	0	0	1
Q5	1	0	0	0	1	1
Q6	0	0	0	0	1	1

Her bir zaman aralığında kontrol sinyalleri gönderilirken şu hususlara dikkat edilmelidir.



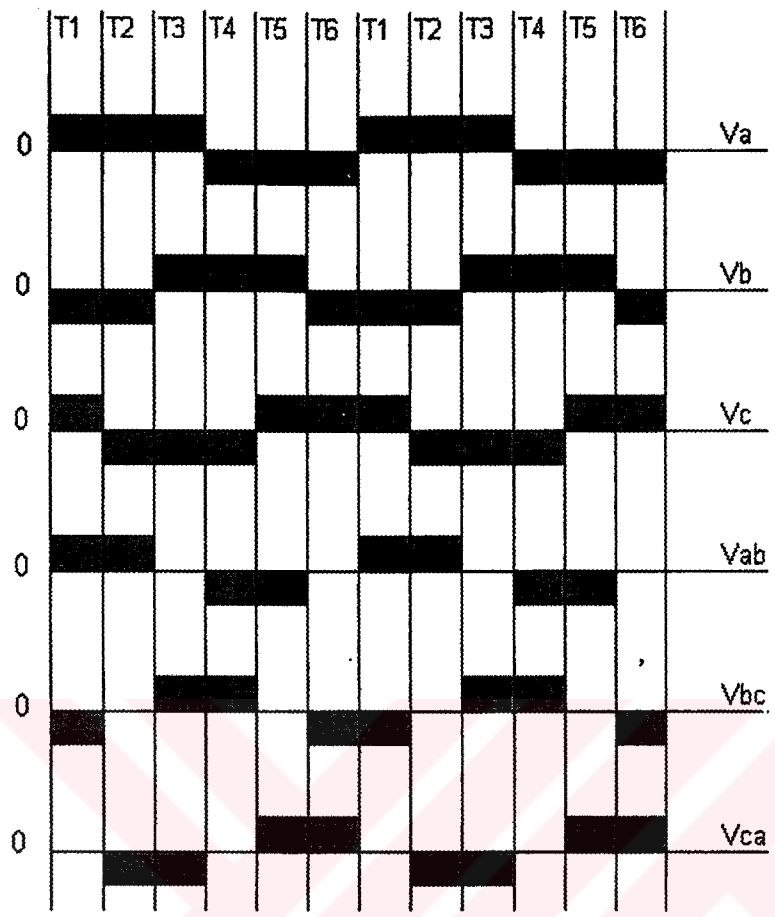
Şekil 4.2 Anahtarlama elemanlarının motora bağlanışı

1. Kol anahtarlama elemanlarından her ikisi aynı anda iletim veya kesimde olamaz. İki elemanın aynı anda kesimde olması bir fazın olmaması demektir. İki elemanın iletimde olması güç kaynağının kısa devre olmasına sebeb olur.

2. Aynı zaman adımın da her kolun bir elemanı iletimde olabilir.Bir peryotluk sürede elemanlardan her birisi yarım peryot iletimdedir.

3. Kollardan birinde bulunan bir anahtarlama elemanı aynı satırda bulunan ve kendisinden önce gelen elemanlarından 120 derece elektrikli açı sonra iletime geçer.Çıkış dalga sı Şekil 4.3'deki gibi oluşur [7].

İnverter çıkışında istenen frekansta bir kare dalga elde edilir. Devir sayısı ayarında evirici çıkışında elde edilen AC sinyalin(Kare dalga olarak) frekans ve gerilimin etkin değerinin ayar edilebilmesi motor kayıplarının azaltılması ve momentin sabit tutularak düzgün bir hız ayarı yapılabilmesi için gereklidir.

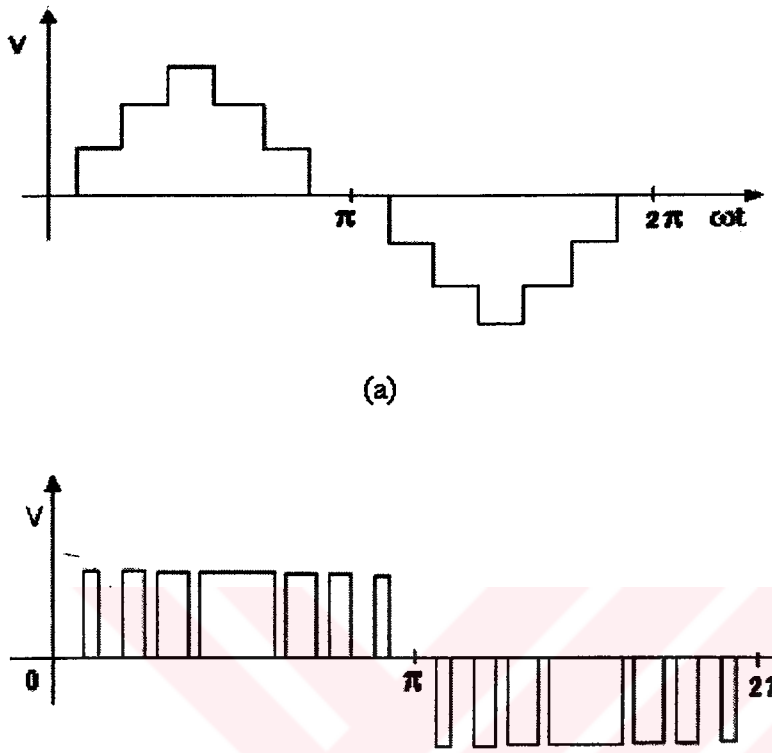


Şekil 4.3 Üç fazlı AC sinyalin oluşumunu gösteren dalga şekilleri

Evirici çıkışında gerilimin etkin değerini değiştirmek için

1. *Pals genişlik modülasyonu (PWM)kullanılabilir.*
2. *Pals genlik modülasyonu (PAM) kullanılabilir.*

Şekil 4.4'de bu iki sinyalin dalga şekilleri görülmektedir.



Şekil 4.4 Frekans modülasyonu çeşitleri

(a) Pals genişlik modülasyonu

(b) (b) Pals genlik modülasyonu

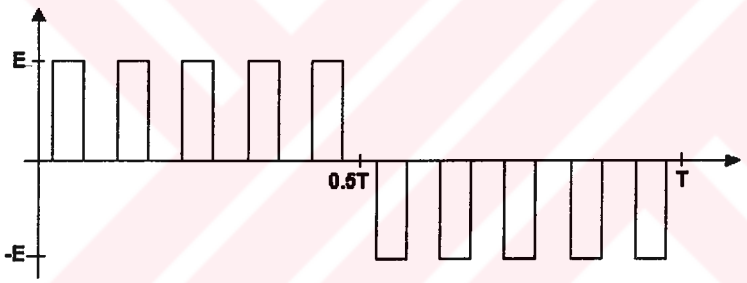
Pals genişlik modülasyonunda pals genişlikleri değiştirilerek frekans ve genlik ayarı yapılır. Pals genlik modülasyonunda ise basamak ayarı yapılır. Pals genişliklerini ayarlamak genlik ayarı yapmaktan daha kolaydır . Bu yüzden pals genişlik modülasyonu daha çok kullanılır [4].

4.2 Pals Genişlik Modülasyonlu İnverter.

Bir önceki konuda anlatılan altı adımlı inverter de çıkış sinyali kare dalga şeklindeydi. Kare dalga şeklindeki AC sinyalin Düşük frekanslı harmonik bileşenler fourier analizi yapıldığında yüksek olduğu görülür. Bu ise yük

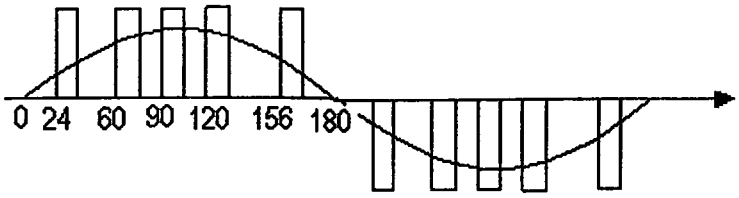
uçlarında ısı şeklinde kendini göstererek motor momentinde bir değişiklik olmamasına karşılık kayıplarda bir artma meydana getirir. Ayrıca harmonik distorsiyon alet ve araçlarda hatalara neden olur. Bu yüzden kare dalganın özellikle düşük frekanslı(3, 5, 7. harmonikler) harmonik bileşenlerinin yok edilmesi gerekir. Yüksek frekanslı büyük genlikli harmonik bileşenler ise filtre elemanları ile kolaylıkla elemine edilebilir. Harmonikleri azaltmanın bir yolu da PWM inverter kullanmaktır. Çeşitli PWM teknikleri vardır.

Her yarım periyotda belli sayıda ve sabit genişlikte (Şekil 4.5). palsler kullanılır. Bu teknikte sadece genlik kontrolü yapılır harmonik bileşenler dikkate alınmaz. Motor hızında titreşimler meydana gelir.



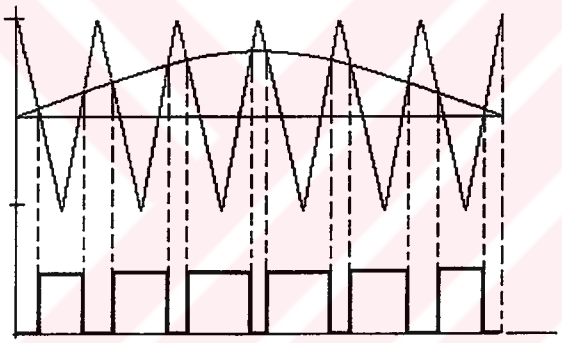
Şekil 4.5 Pals genişliği sabit PWM

Her yarım peryottaki pals genişlikleri ve pals pozisyonları öyle ayarlanırki bunların ağırlıklı ortalamaları (Şekil 4.6) sinüs dalgasına benzer. Bu yöntemde de belli sayıda pals kullanılır ve pals genişlikleri sabittir. Palslerin yerleşim açıları harmonik bileşenleri azaltacak şekilde hesaplanmıştır. Bu metod da harmonik bileşenler oldukça düşürülmüştür.



Şekil 4.6 Pals genişliği sabit pozisyonu ayarlı PWM

Bir üçgen dalga ile sinüs dalgası karşılaştırılarak sinüsoidal PWM elde edilir. Yani sinüs dalgası bir üçgen dalga ile modüle edilir(Şekil 4.7).. En iyi sonucu bu yöntem verir [8] .



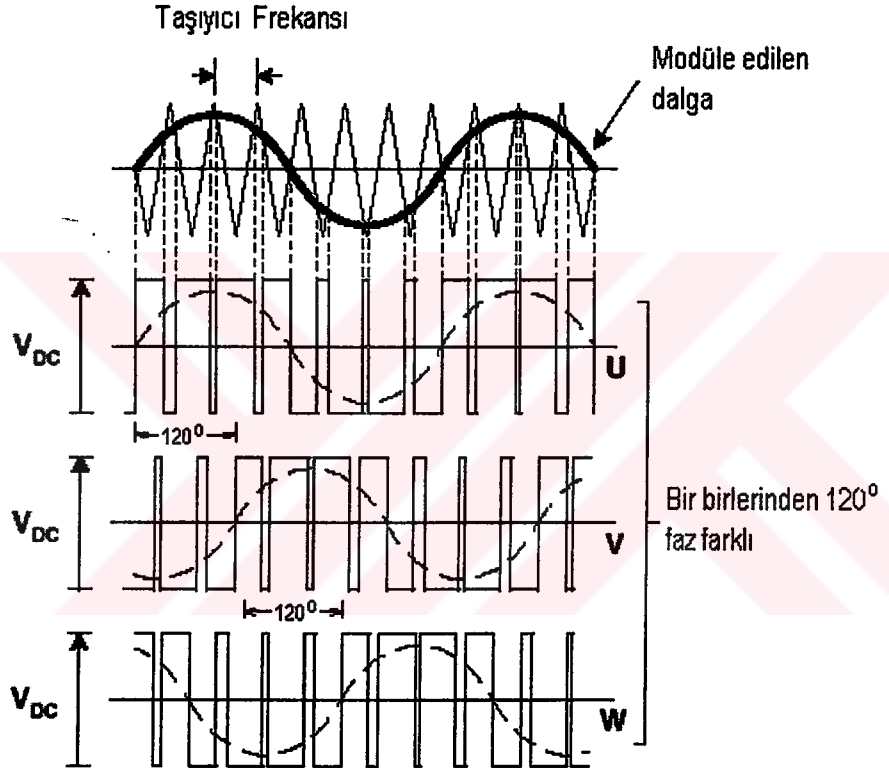
Şekil 4.7 Üçgen dalga ile sinüs dalgasını karşılaştırılması ile PWM elde edilmesi

4.3 Üç Fazlı PWM İnverter

Üç fazlı AC gerilim bir birinden 120 derece faz farklıdır. Gerilim dalga eşitlikleri şu şekilde verilir.

$$V_a = V_m \mathrm{Sin}\omega t$$
$$V_b = V_m \mathrm{Sin}(\omega t - \frac{2\pi}{3}) \qquad (4.1)$$
$$V_c = V_m \mathrm{Sin}(\omega t + \frac{2\pi}{3})$$

Bu eşitliklere göre PWM dalga şekilleri ise Şelkil 4.8'deki gibi olur.



Şekil 4.8 Üç faz PWM sinyalleri

5. MİKRODENETLEYİCİNİN YAPISI

Bir mikrodenetleyici içinde RAM, ROM, giriş/çıkış portları, zamalayıcı ve sayıcı, seri haberleşme arabirimi, kesme girişleri olan bir çeşit küçük bilgisayardır. Özellikle kontrol amaçlı uygulamalarda tercih edilirler. Bunda çok fonksiyonlu olmasının yanı sıra az yer kaplaması, gerçekleştirilen sistemin maliyetini azaltması gibi sebepler rol oynamıştır. Hız kontrol devresinde 8051 mikrodenetleyici yapısı ile yazılım ve donanım uyumluluğu olan AT89C52 ve AT89C55 mikrodenetleyicileri kullanılmıştır. Bu mikrodenetleyicilerden AT89C52 8KB flash program hafızasına sahiptir ve maksimum 24 Mhz saat frekansında çalışırken AT89C55 20 KB flash program hafızasına sahiptir ve maksimum 33 Mhz saat frekansında çalışabilir. Bu yüzden defalarca silinip programlanabilirler. Diğer özellikleri aynıdır. Çizelge 5.1'de bu iki işlemcinin genel özellikleri ve Çizelge 5.2'de ise pinlerin ne için kullanıldıkları görülmektedir [9].

Çizelge 5.1 Kullanılan mikrodenetleyicilerin özellikleri

	Flash program bellek	RAM bellek	Giriş/Çıkış hattı	Seri port	Zamanlayıcı	Saat frekansı	Kesme sayısı	Program Kilitleme
AT89C52	8 Kb	256 B	32	1	3	24Mhz	8	Var
AT89C55	20 Kb	256 B	32	1	3	33Mhz	8	Var

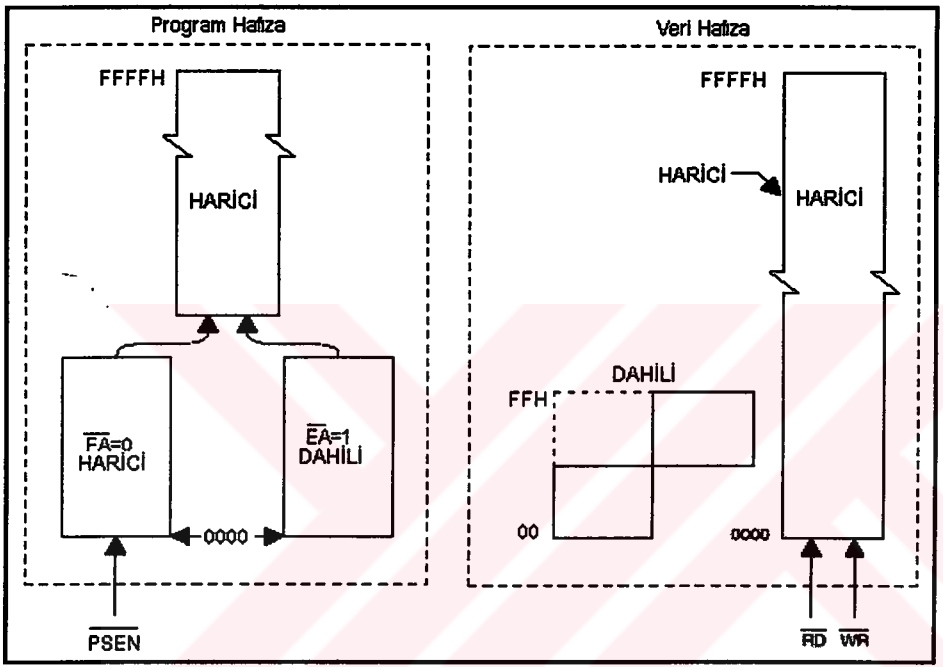
Çizelge 5.2 8051 Mikrodenetleyicisinin pin fonksiyonları

NO	SEMBOL	AÇIKLAMA
1-8	PORT 1	Giriş ve çıkış için kullanılabilir. Harici Pull-up direnci gerektirmez.
9	RST	Mikrodenetleyici RESETLEME ucu. Program 0000H adresinden çalışmaya başlar.
10-17	PORT 3	Giriş ve çıkış için kullanılabilirler. Harici Pull-up direnci gerektirmezler.
18	XTAL 2	Saat frekansı giriş ucu 2
19	XTAL 1	Saat freansı giriş ucu 1
20	VSS	Toprak
21-28	PORT 2	Giriş ve çıkış için Adres hattı olarak kullanılabilirler.
29	PSEN	Dış program belleğinden veri okumak için kullanılır.
30	ALE	Harici Program memory adresini seçmek için kullanılır.
31	EA	Mikrodenetleyici içindeki ROM un Program Belleği olarak kullanılıp kullanılmayacağına karar verir.
32-39	PORT 0	Giriş ve çıkış olarak kullanılabilir. Harici Pull-up direnci gerektirirler.
40	VCC	Mikrodenetleyicinin Pozitif besleme ucu(+5 V)

5.1 Hafıza Düzeneği

8051 mikrodenetleyicisi program hafıza ve veri hafıza olmak üzere iki ayrı hafıza düzeneğine sahiptir. Şekil 5.1'de 8051'in hafıza düzeneği görülmektedir.

Program hafıza yazılan programların komut kodlarının tutulduğu hafıza dır. Mikrodenetleyici dahili 2K, 4K, 8K, 10K,12K,16K, 20K,32K ROM hafızaya sahip olabilir. Eğer yazılan program bu dahili hafızaya sığmayacaksa harici ROM kullanılabilir. Bu durumda EA pini 0 yapılmalıdır. Harici ROM kullanılmayacak ise EA pini 1 (+5V) yapılmalıdır.

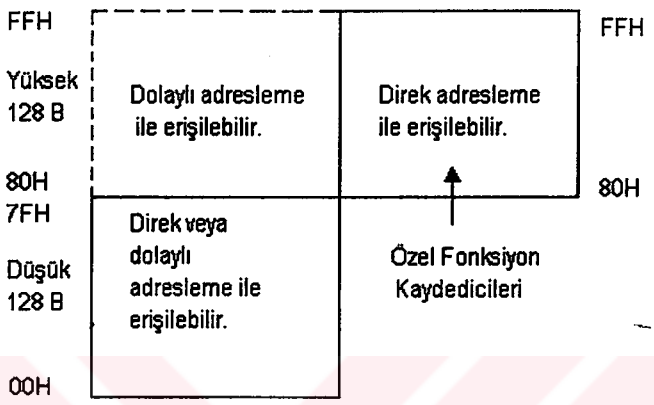


Şekil 5.1 Mikrodenetleyicinin hafıza düzeneği

Veri hafızası 128 bayt veya 256 bayt olabilir. Standart olarak 128 bayt tüm 8051 ailesinde vardır. AT89C52 ve AT89C55 256 bayt veri hafızasına sahiptir. Ayrıca 64 KB lik istenirse harici bir veri hafıza eklenebilir.

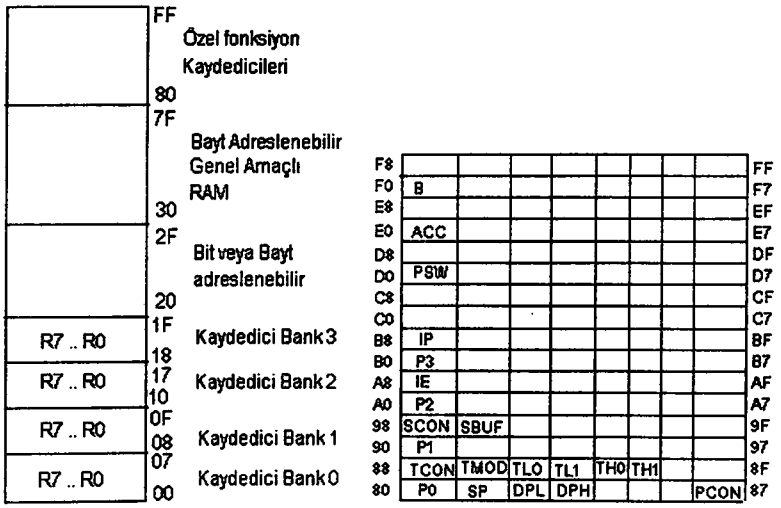
Dahili veri hafızası(RAM) 256 bayt ise ilk 128 bayt (00-7FH adresleri arası) direk veya dolaylı adresleme ile erişilebilir. Kalan 128 bayta (80-FFH adresleri) erişmek için dolaylı adreslemeden yararlanılır. Birde görünüşte aynı adres alanını kullanıyormuş gibi görünen(80-FFH) özel fonksiyon kaydedicilerinin kullandığı bir alan daha vardır ki bu adres alanı fiziksel

olarak farklı bir bölgededir ve direk adresleme ile erişilebilir. Yani Direk adresleme ile 80-FFH adreslerine erişilirse Özel fonksiyon kaydedicilerine erişilmiş olur. Aynı adres bölgesine dolaylı adresleme ile erişilmiş ise veri hafızaya erişilmiş olur. Bu durum Şekil 5.2'de görülmektedir.



Şekil 5.2 İç veri hafıza düzeneği

İlk 128 bayt RAM bölgesi ile 128-256 bayt arası özel fonksiyon kaydedicileri Şekil 5.3'de görülmektedir.0 ile 1F adresleri arasında 4 kaydedici bankı vardır.



Şekil 5.3 SFR hafıza haritası

Bu bankların her birinde 8 adet R0 dan R7 ye kadar 8 tane kaydedici vardır. Bu kaydedicilere ulaşmak için öncelikle PSW den bank seçimi yapılır daha sonra R0 dan R7 ye kadar olan kaydediciler direk adresleme ile kullanılabilir. 20-2F arası bit veya bayt adreslenerek kullanılabilir. 2F,7FH arası genel amaçlı olarak kullanılabilir [9].

5.2 Durum Kaydedicisi(Program Status Word):

Mikrodenetleyicinin yaptığı iş hakkında çeşitli bilgiler verir. PSW üzerindeki her bir bitin özel bir anlamı vardır. Bu bitler kontrol edilerek mikrodenetleyicinin yaptığı işler hakkında bilgi alınabilir ve istenirse bunlar değerlendirilerek program akışı değiştirilebilir. PSW kaydedicisi Şekil 5.4'de görülmektedir.

CY	AC	F0	RS1	RS0	OV	-	P

CY: Elde bayrağı
AC: yardımcı elde baytrağı
F0: Genel amaçlı durm bayrağı
RS1, RS0 Bank seçim bitleri

0	0	Bank0
0	1	Bank1
1	0	Bank2
1	1	Bank3

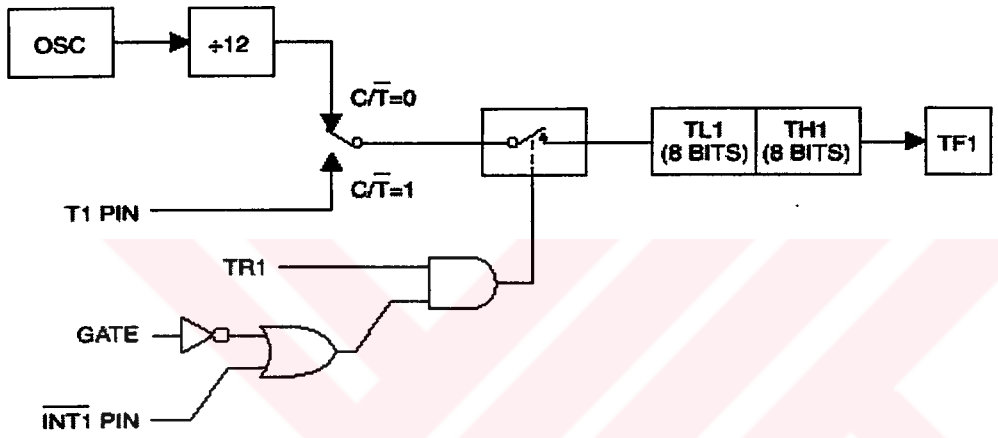
P: Eşlik(Parity) biti

Şekil 5.4 PSW kaydedicisi

5.3 Zamanlayıcı ve Sayıcı Yapıları

Zamanlayıcı/Sayıcılar iç veya dış kaynaklı bir zaman peryodunu ölçmek için veya olayları saydırmak için kullanılır. Örneğin girişe uygulanan bir darbenin genişliğinin veya frekansının ölçülmesi buna örnek olarak verilebilir.

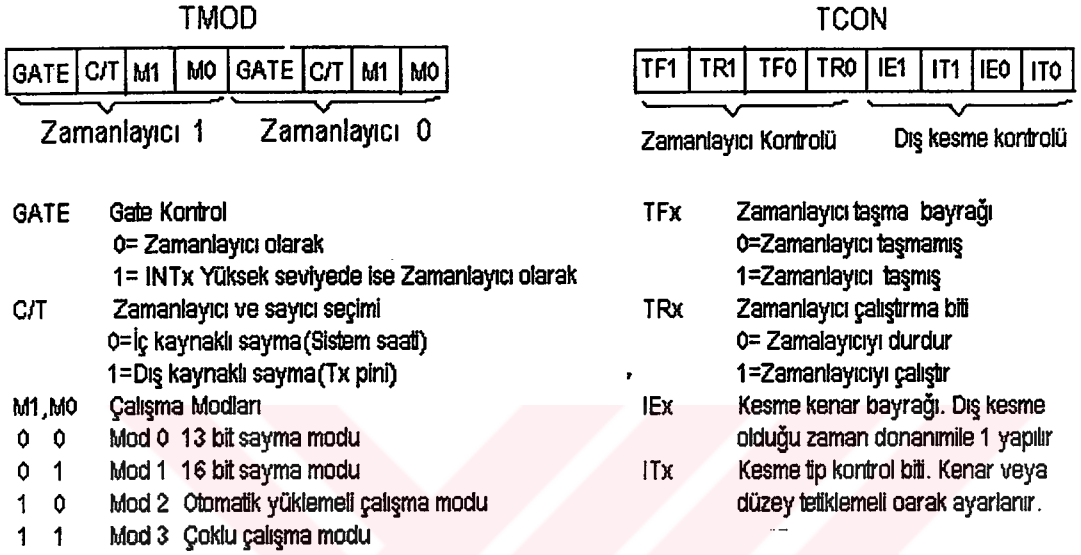
8051 ürünlerinin çoğunda iki adet 16 bitlik zamanlayıcı (T0, T1) ve her zamanlayıcı için iki tane özel fonksiyon saklayıcısı vardır. Bunlar T0 için TH0 ve TL0, T1 için TH1, TL1 dir. Bu sekiz bitlik zamanlayıcılar birleştiklerinde 16 bitlik zamanlayıcıyı oluştururlar. Çalışma esnasında her giriş darbesinde bir veya iki SFR deki sayma değeri bir artırılır. Şekil 5.5'de zamanlayıcının yapısı görülmektedir.



Şekil 5.5 Zamanlayıcı/sayıcının yapısı

Zamalayıcı/sayıcı zamanlayıcı durmunda iken makine periyodunu sayar. Makine periyodu saat frekansının 1/12 sidir. Sayıcı durumunda makine periyoduna bağlı olarak P3.4(T0) veya P3.5 (T1) dış giriş ucundan örnek alır. Giriş uçlarının 1 den 0'a geçişlerinde saklayıcının değeri 1 artar. Bir sonraki örnekleme için bir makine periyodunun geçmesi gerekir. Bu yüzden örnekleme frekansı osilatör frekansının 1/24 üne eşittrir. Daha yüksek frekanslarda hatalı sayımlar oluşur. Zamanlayıcı sayıcı ayrıca sayma değeri taştığında bir kesme üretir. Bunun için ilgili kesme bayrağı yetkilendirilmesi gerekir.

Zamanlayıcı/sayıcı TMOD ve TCON kaydedicileri programlanarak kullanılırlar. Şekil 5.6'de TMOD ve TCON Kaydedicileri ve işlev tanımlamaları görülmektedir.



Şekil 5.6 TMOD ve TCON kaydedicileri

TMOD ilk 4 biti zamanlayıcı 0 ve ikinci 4 biti zamanlayıcı 1 ayarlamak için kullanılır. GATE=1 olduğunda Zamanlayıcı x'in INTx dış kesmesinin 1 olması yeterlidir..

C/T zamanlayıcı veya sayıcı seçme bitidir. Zamalayıcı olarak çalışacaksa bu bit 0 yapılır. Bu durumda sistem saatini sayar. Eğer bu bit 1 yapılırsa harici Tx pininden gelen darbeleri sayar.

M1 ve M0 bitleri 4 çalışma modundan birini seçer. Mod 1 seçildiğinde 16 bitlik sayma yapar yani TH ve TL nin mevcut değerlerinden başlayarak 65535(FFFF) değerine kadar sayar. Bu değere ulaştığında kesme üreterek TFx bayrağını set eder. Zamanlayıcıyı çalıştırmak veya durdurmak için TCON kaydedicisindeki TRx bitlerinden yararlanılır [10].

5.4 Zamanlayıcı İle Zaman Gecikmesi Yapmak

Zamanlayıcı THx ve TLx kaydedicisinin birlikte oluşturduğu değerden itibaren 65535 değerine kadar sayar. Bundan yararlanarak istene zaman gecikmesi sağlanabilir. Örnek olarak 50 ms lik bir zaman geçikmesi isteyelim. Sistem saat frekansı 11.0592 Mhz olsun bunun için şu sıra izlenir.

Sayıcı saat frekansı=Sistem saat frekansı /12
=11.0592/12
=921.6 Khz

Her sayma için geçen
zaman =1/sayıcı saat frekansı
=1/921.6Khz
=1.085μs

Gecikme için sayma
Sayısı = Gecikme zamanı/Her sayım için geçen zaman
= 50ms/1.085μs
=46080

THx ve TLx kaydedicilerine yüklenecek değer
Ön yükleme sayısı = 10000.h-Gecikme için sayma sayısı
= 65536-40080
= 19456=4C00H

Dolayısı ile THx=4C ve TLx=00 değerleri hexadesimal olarak yüklenip zamanlayıcı çalıştırıldığında 50 ms lik geçikme sağlanacaktır [10].

5.5 8051 Adreslame Türleri

8051 Mikrodenetleyici sisteminde komutlar belli adresleme türlerine göre çalışırlar. Bu adresleme türleri şunlardır.

Kaydedici (Register) Adresleme : Sistemin iç RAM'ında 4 adet kaydedici bankı vardır. Bu banklarda yer alan(Bank0, Bank1, Bank2, Bank3) R0'dan R7'ye kadar olan kaydedicilerle yapılan işlemlerde bu adrsesleme modu kullanılır.

Örneğin; MOV R0,A

Bu komut A (accumulator)'da ki bilgiyi R0 kaydedicisine yükler.

Doğrudan (Direct) Adresleme : Bu adreslemede iç RAM'deki adreslere doğrudan erişim sağlanmış olur.

Örneğin; MOV A,20H

20H RAM adresinde ki bilgi doğrudan A'ya aktarılır.

Dolaylı Adresleme : Dolaylı adreslemede kaydediciler üzerinden, hedeflenen adreslerle işlemler yapılır. Bu türde sadece R0 ve R1 kaydedicileri kullanılır.

Örneğin; MOV @R0,A

"@" simgesi dolaylı adreslemenin simgesidir. R0'da kayıtlı olan bilgi 20H bilgisi olsun. Bu komut çalıştığında A'daki bilgiyi R0'ın gösterdiği adrese yani 20H adresine kaydeder.

Veri (Immediate) Adresleme : Bu türde işlemler doğrudan veri ile ilgilidir ve iki türlüdür. Bunlar 8 bit veri ve 16 bit veri adreslemedir.

Örneğin; MOV A,#02H

"#", veri adreslemenin simgesidir. Yukarıdaki komut 8 bitlik veri adreslemedir ve doğrudan 02H bilgisini A'ya kaydeder.

MOV DPTR,#1025H

Komutu ise 16 bit adreslemedir. DPTR (Data Pointer)'ye 16 bitlik 1025H bilgisini kaydeder.

Hedef (destination) Adresleme : Alt program çağırma ve koşulsuz dallanma komutları icra edilirken bu adresleme kullanılır, iki türü vardır.
16 bitlik destination adresleme; LCALL ve LJMP komutlarında kullanılır ve 16 bit dallanma yani belleğin her yerine dallanabilirler.
11 bitlik destination adresleme; ACALL ve AJMP komutlarında kullanılır ve komutun bulunduğu 2K'lık blok içinde dallanma gerçekleşir.

Göreceli (Relative) Adresleme : Dallanma komutları icra edilirken bu adresleme türü kullanılır. Eğer SJMP komutu kullanılmış ise dallanma aralığı +128 ile –127 byte arasında sınırlıdır. Yani komutun bulunduğu yerden geriye doğru 127 byte, ileriye doğru 128 byte dallanır. LJMP komutunda böyle bir sınırlanma yoktur.

Örneğin;
```
GİT: MOV A,#30H
     MOV R0,A
     SJMP GİT
```
Burada sırasıyla A'ya 30H bilgisi atanır. Sonra A'daki bu bilgi P0'a aktarılır. SJMP ile de koşulsuz olarak GİT etiketinin belirlediği adrese dallanır.

Bit Adresleme : Bu adresleme türü iç RAM üzerinde bit bit işlemler yapılırken kullanılır.
Örneğin; MOV 00H,C
Bu komut C (caryy elde) bitini 00 bit adresine yazar.

Program belleği adresleme
Eğer program belleğne bir takım bilgiler yazılmışsa bunları okumak için MOVC komutu ve DPTR, A kaydedicileri ile dolaylı adreslem kullanılır. DPTR kaydedicisi DPH ve DPL olarak iki 8 bitlik kaydediciden oluşur. Bunlar istenirse ayrı ayrı veya birleşik olarak kullanılabilirler. Genelde

Program belleğinde bir takım tablolara erişmek için kullanılır. Örnek program aşağıda verilmiştir.

```
MOV DPTR,#TAB1      ;DPTR ye TAB1 in başlangıc adresini yükle
BAS:
CLR A               ; A=0
MOVC A,@A+DPTR      ;Tablodaki ilk değeri A ya at
MOV P0,A            ;A daki değeri P0 portuna gönder
INC DPTR            ; Adresi 1 artır
SJMP BAS            ;BAS'a git
TAB1:
DB 079H,04AH,0,0FEH,04CH,1
```

Yukarıdaki programda TAB1 diye bir tablo program belleğinde tanımlanmıştır. Her seferinde TAB1 tablosundan bir değer alınıp porta gönderilmektedir.

5.6 Kesmelerin Kontrolü

Mikrodenetleyici özellikle giriş çıkış işlemlerinde bir porttan değer gelip gelmediğini sürekli kontrol etmek zorunda kalır. Bu ise sürekli olarak mikrodenetleyiciyi meşgul ederek diğer işleri aksatabilir. Bunu önlemek için kesmeler den yararlanılır.

Mikrodenetleyici bir işle meşgul iken kendi işini bırakarak çevre birimlerden gelen isteğe cevap vermesi ve daha sonra işine kaldığı yerden devam etmesidir.

Çevre birimler isteklerini mikrodenetleyiciye kesme girişlerine sinyal göndererek veya bazı kaydedicilerdeki bitleri set ederek iletirler.

Çizelge 5.3 8051 kesme adresleri

Kesme	Kaynağı	Kesme sebebi	Kesme vektör adresi
Harici kesme 0	INT0 pin	Düşük düzey veya düşen kenar	0003H
Zamanlayıcı 0	TF0	FFFF den taşma	000BH
Harici kesme 1	INT1 pin	Düşük düzey veya düşen kenar	0013H
Zamanlayıcı 1	TF1	FFFF den taşma	001BH
Seri port	RI veya TI	Veri gönderiminin veya alımının tamamlanması	0023H

8051 in beş adet kesme üreten kaynağı vardır. Bunlardan iki tanesi harici diğerleri dahilidir. Bir kesme geldiğinde işlemci derhal işini bırakıp gelen kesme kaynağına göre kesme vektör adresine dallanır. Burada kesme hizmeti için 8 baytlık bir yer ayrılmıştır. Eğer bu adres alanı hizmet görecek alt programın yazılımı için yeterli ise alt program buraya yazılır. Eğer yeterli değilse kesme alt programına bu adrese konulan bir LJMP komutu ile gidilip RETI komutu ile dönülür. Mikrodenetleyicili sisteme güç verildiğinde veya RESET kesmesi üretildiğinde mikrodenetleyici 0000H adresine dallanarak buradaki ilk komutu işlemeyeye başlar. Çizelge 5.3 de kesmeler ve kesme vektör adresleri görülmektedir [10].

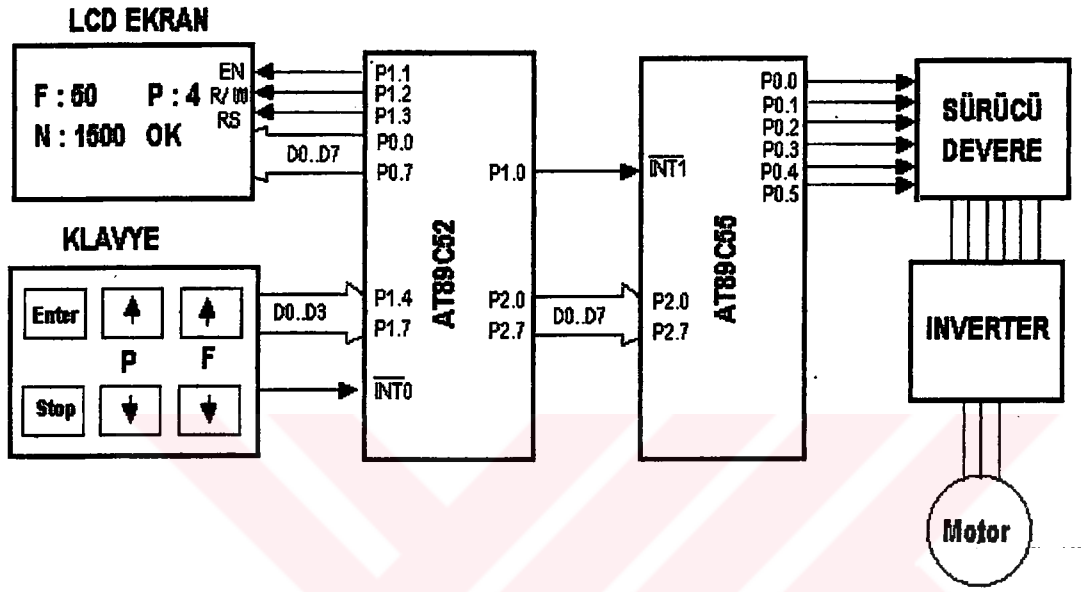
6. HIZ KONTROL DEVRESİNİN GERÇEKLEŞTİRİLMESİ

Üç fazlı sinüsoidal PWM inverter kontrol devresinin gerçekleştirilmesi için frekansı ayarlanabilen sinüsoidal dalga üreteci, taşıyıcı üçgen dalga üreteci, bu iki dalgayı kıyaslayıcı devre ve çeşitli lojik kapı devreleri gerekmektedir. Bunların büyük bir kısmı mikroişlemci kullanarak elemine edilebilir. İntel firmasınca özel olarak AC motor kontrolü için tasarlanmış 8XC196MC mikrodenetleyicili hız kontrol devresinin yapımı için iyi bir seçim olacaktır. Ancak bu mikrodenetleyicinin maliyetinin fazla olması nedeniyle ucuz başka bir mikrodenetleyicinin kullanılması zorunlu olmuştur. Kontrol devresinde kullanılan mikrodenetleyici ile bir çok çevre eleman elenirken, sistem tasarımı büyük ölçüde bu işlemcinin yazılımla kontrolüne kalmaktadır.

6.1 Kontrol Devresinin Yapımı

Kontrol devresinin genel blok şeması Şekil 6.1'de görülmektedir. Kontrol devresinde iki adet mikrodenetleyici kullanılmıştır. Bu mikrodenetleyicilerden birisi sistemi kullanan kişiye hız ayarı için gerekli olan frekans ve kutup sayısı bilgisini klavyeden girmesini sağlar. Girilen değerleri LCD ekranda direk olarak görmek mümkündür. Devre çalıştırıldığında varsayılan değerler olarak Frekans değeri 0, kutup sayısı 1 olarak ayarlanır. Bu değerleri ekranda görmekte mümkündür. Başlangıç anında motor durmaktadır. Kullanıcı klavyeden F ve P tuşlarını kullanarak frekans değerini 0-60 Hz olarak ve Çift kutup sayısı P değerini 1-5 arası değerler şeklinde artırma veya azaltma tuşları ile birinci mikrodenetleyiciye gönderir. Bu değerler birinci mikrodenetleyiciye INT0 ucu vasıtası ile bildirilir. İstenen değerler ayarlandığında frekans ve kutup sayısı değerine göre motorun senkron devir sayısı hesaplanarak LCD ekranda kullanıcıya gösterilir. Girilen değerler ENTER tuşuna basılmadıkça işlem görmez. Yani motor önceki ayarlanan değerinde konumunu korur. Yani

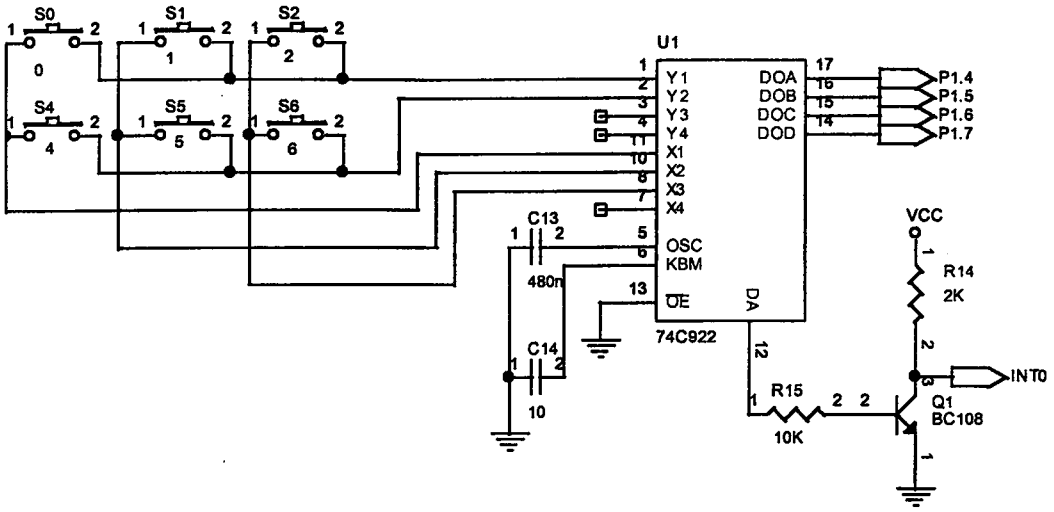
duruyorsa durur, çalışıyorsa o devrinde dönmeye devam eder. ENTER tuşuna basıldığı anda birinci mikroişlemci P1.0 ucu yardını ile ikinci işlemcinin INT1 ucuna bir kesme göndererek P2 portundan frekans bilgisini bildirir.



Şekil 6.1 Hız ayar devresinin blok şeması

İkinci mikrodenetleyici AT89C55 mikrodenetleyicisidir. Bu mikrodenetleyicinin 20Kb lik bir EEPROM belleği vardır. Bu sayede işlemcinin içine büyük miktarlarda tablo değerleri saklanabilir. Tasarlanan devrede girilen frekans değerlerine göre istenen sinüsoidal PWM dalga formu üreten değerler herbir frekans için hesaplanıp tablolar halinde bu işlemcinin içine yerleştirilmiştir. Bu tablodan yararlanarak ikinci işlemci inverter için gerekli anahtarlama sinyallerini bir yazılım vasıtası ile gönderir.

6.2 Klavye Devresi



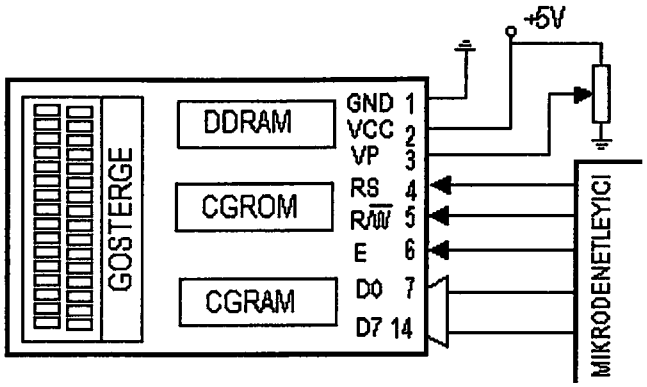
Şekil 6.2 Klavye devresi

Klavye devresinde (Şekil 6.2) 74C922 dekoder entegresi kullanılmıştır. Bu entegrenin klavye tuşlarının bağlandığı satır ve sutun girişleri ile bir interrupt çıkışı ve veri çıkış portları vardır. Herhangi bir tuşa basıldığında 74C922 entegresi basılan tuşu satır ve sutunları tarayarak bulur. Veri çıkış uçlarına tuşun hexadesimal sayı karşılığını koyar ve tus kodunun hazır olduğunu belirtmek için DA çıkış ucunu belli bir süre(yaklaşık 100ms) set eder ve daha sonra tekrar lojik 0 değerine çevirir. 74C922 dekoder entegresinin DA çıkışının kullanımı ile kesme üretilerek tuş kodlarının işlemciye gönderilmesi sağlanır. Mikroişlemci kesme girişleri lojik 0 da aktif olduğundan bu entegrenin çıkışına bir NAND(ve değil) kapısı konulmuştur [10].

6.3 LCD Gösterge

Devre tasarımında girilen değerleri(frekans, kutup sayısı) ve hesaplanan devir sayısını ekranda göstermek için İki satır 16 karakterlik (2x16) LCD gösterge kullanılmıştır. Bu seçimin yapılmasının nedeni LCD gösterge kontrolünün yazılımla daha kolay olmasıdır. LCD ler çeşitli firmalar tarafından üretilmesine rağmen kontrolleri standartlaşmıştır. Tüm LCD ekranlarda yetki

(Enable), OkuYaz(R/W), ve Kaydedici Seç(RS) uçları ile veri giriş hatları vardır (Şekil 6.3).



Şekil 6.3 LCD iç yapısı

LCD göstergeler 40 karekter ve 4 satıra kadar çeşitli seçenekler sunarlar. LCD göstergeler 80 adet karektere kadar kodu saklayabilmek için bir dahili RAM bulundururlar . Bu RAM'a Gösterge Veri RAM'ı (Display Data RAM) adı verilir. Bir satırında 16 karekteri olan 2 satırlık bir göstergeyi herbir satırında 40 karekteri olan 2 sanal satır olarak düşünebiliriz. Ancak bir anda 40 karekterden 16 sı görülebilmektedir. Kalan karekterler ancak kaydırma işleminden sonra görülebilirler. arzu edilen herhangi bir noktaya bir karekter yazdırılmak istenirse önce DDRAM adresi seçilmeli ve daha sonra karekter kodunun buraya yazılması gerekir.

6.3.1 Karekter kümesi

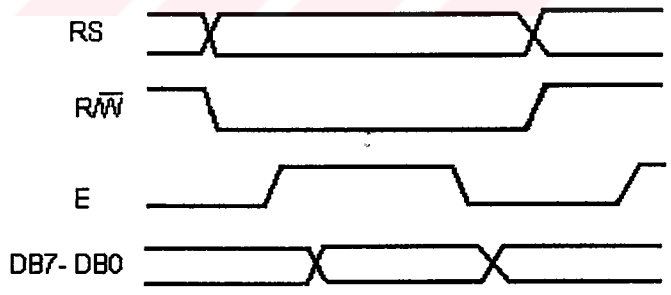
LCD gösterge önceden tanımlanmış veya kullanıcı tarafından tanımlanan karekterleri gösterebilmektedir. LCD gösterge 192 adet karekter içeren bir Karekter Üretici ROM'a (Character Generator ROM) sahiptir. Bu karekterlerden 96 tanesi ASCII karekterdir. Bu karekterler belirli kodlarla seçilirler. Bu kodlar LCD göstergelerin kullanım kitapcıklarından

öğrenilebilir. Ayrıca LCD göstege Karakter Üretici RAM (Character Genarator RAM) olarak isimlendirilen ve kullanıcı tarafından tanımlanabilen 8 karakterden oluşan bir hafızaya da sahiptir.

6.3.2 LCD kontrol işlemleri

LCD göstergeye gönderilen veri ya bir komut kodu veya bir karekterdir. LCD göstergelerde ekranın temzlenmesi, İmlecin başa alınması , DDRAM adresinin seçilmesi, Resetleme gibi komut işlemleri vardır. Bunun için LCD ye gönderilen verinin Komut mu yoksa ekranda gösterilecek bir karakter mi olduğu belirtilmelidir.

Şekil 6.4'de LCD göstergeye yazma işlemine ait zamanlama diyagramı görülmektedir. RS (Register Select) ucu düşük seviyeye çekilirse yapılacak işlem bir kontrol işlemidir. Eğer Yüksek seviyede tutulursa gönderilen bir komut değil karakterdir.

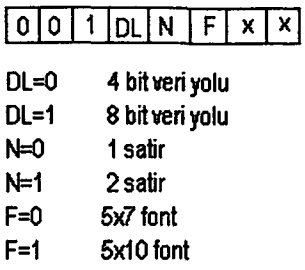


Şekil 6.4 LCD göstergeye yazma işlemine ait zamanlama diyagramı

6.3.3 LCD komutları

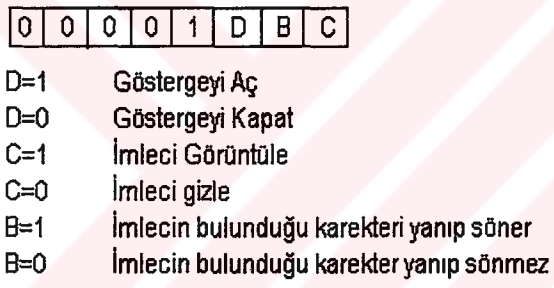
Burada ençok kullanılan LCD komutları anlatılacaktır.

Fonksiyon belirleme(Function Set) : Bu komut ile veri yolunun büyüklüğü göstergedeki satır sayısı ve font büyüklüğü belirlenir(Şekil 6.5).



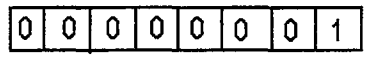
Şekil 6.5 Fonksiyon belirleme

Göstergeyi kapat/aç (Display off/on) : LCD göstergeyi açmak yada kapatmak için kullanılır. Göstege kapalı iken DDRAM daki veriler korunur. Bu komutun ikili gösterimi Şekil 6.6'daki gibidir.



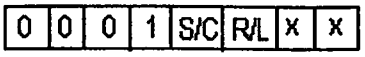
Şekil 6.6 Göstergeyi aç kapat

Göstergeyi temizle(Clear Display): Göstergenin temizlenmesi DDRAM daki her hücreye boşluk(20H) karakterinin yazılmasını ve imlecin sol baştan göstergenin ilk karakterine konumlanmasını sağlar (Şekil 6.7).



Şekil 6.7 Göstergeyi temizle

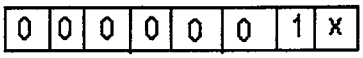
İmleci kaydır(Shift Display): İmleci veya göstergeyi sola veya sağa kaydırmak için kullanılır(Şekil 6.8).



S/C= 0 İmleci kaydır
S/C=1 Göstergeyi kaydır
R/L=0 Sola kaydır
R/L=1 Sağa kaydır

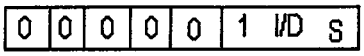
Şekil 6.8 İmleç kaydırma

İmleç Başa (Cursor Home) : İmleci en sola ve en üst satıra konumlandırır. (Şekil 6.9)



Şekil 6.9 İmleç başa

Giriş Kipi(Entry mode) : Giriş kipi her karakter okuma veya yazma işlemini takiben, imlecin veya göstergenin işlemini belirler. En çok kullanılan işlem modu, göstergenin o anki değerlerinin korunarak imlecin bir sağa kaydırılmasıdır. Bu düzenleme ile bir sonraki karakter o anki konumun bir sağına yazılır (Şekil 6.10).



I/D(İncrement/Decrement)
I/D=0 İmleci bir azalt (Bir sola kaydır)
I/D=1 İmleci bir artır (Bir sağa kaydır)
S (Shift)
S=1 Göstergenin tüm içeriğini kaydır
S=0 Göstergeyi kaydırma

Şekil 6.10 Giriş kipi

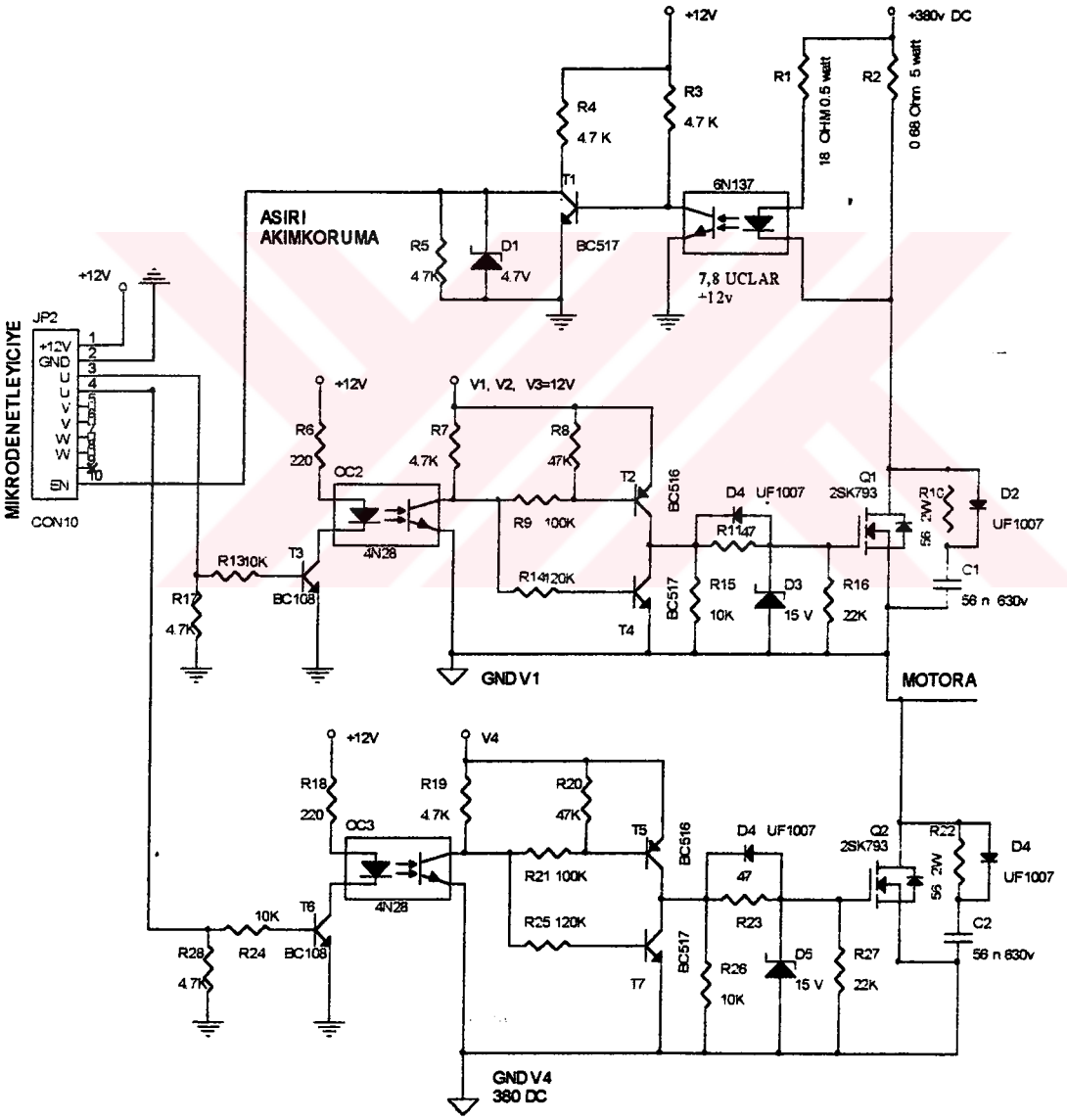
6.3.4 LCD ile 4 bit veri yolunu kullanma

LCD 4 veya 8 bit veri yolu kullanabilir. 4 bit kullanımda DB7-DB4 kullanılır. Her veri veya komut iki adımda işlenir. Önce verinin yüksek dörtlü ve daha sonra da düşük dörtlü kısmı gönderilir. LCD ilk çalıştığında varsayılan 8 bit veri yolu kullanımıdır. Dört bitlik veri yoluna geçiş için önce 20H komutu gönderilir. Daha sonra diğer başlangıç komutları (Sırası ile Fonksiyon set, Göstergeyi aç, Göstergeyi temizle, Giriş modu, İmleç başa) yapılır [9].

6.4 Sürücü Devresi

Sürücü devresindeki MOSFET'ler direkt olarak sürülmezler. Özellikle üstte bulunan MOSFET lerin topraklarını ayırmak gerekmektedir. Çünkü aynı fazın alt ve üst MOSFET'lerinin topraklarını birleştirmek bu fazı kısa devre etmek anlamına gelir. Altta bulunan MOSFET lerin toprakları birleştirilebilir. MOSFET ler +5Volt ile sürülmezler. MOSFET lerin sürülmesi için 5 volttan daha büyük bir voltaj Gate –Source uclarına uygulanmalıdır. Bu da tipik olarak VGS=10 Volt dur. Bu yüzden mikrodenetleyiciden gelen çıkış voltajının yükseltilmesi gerekir. MOSFET'leri kontrol devresinden yalıtmak ve üstteki MOSFET'lerin topraklarını ayırabilmek için optoizolatör kullanılmıştır. Optoizolatör bir transistörle sürülmektedir. Optoizolatör ise push-pull olarak çalışan bir darlington transistör çiftini sürmektedir. Bu darlington çifti ile MOSFET'ler sürülmektedir. Şekil 6.11'de bir fazı sürmek için gerekli devre şeması verilmiştir. Şekil 6.14'deki İnverter devresine bakılacak olursa sürücü devresi ile 6 adet MOSFET'in sürüldüğü görülebilir. Her bir fazı sürmek için altı adet MOSFET'e ihtiyaç duyulmaktadır. Şekil 6.11'deki devrede üstteki MOSFET'leri sürmek için üç ayrı güç kaynağı gereklidir. Bunlar V1, V2, V3 sembolü ile gösterilmiştir . V4 güç kaynağı ise alttaki MOSFET'leri süren devreyi beslemek için kullanılmıştır.

Ayrıca bir fazdan aşırı bir akım çekilmesi durumunda aşırı akım sezme devresi ile sistemin korunması sağlanmaktadır. Üst MOSFET drain ucuna yerleştirilen 0.68 ohm luk dirençlerden aşırı bir akım geçmesi durumunda bu direnç üzerinde oluşan voltaj 6N137 optoizolatörünü iletime geçirmekte ve Enable girişine yetki vererek MOSFET lerin yalıtıma geçmesini sağlamaktadır.



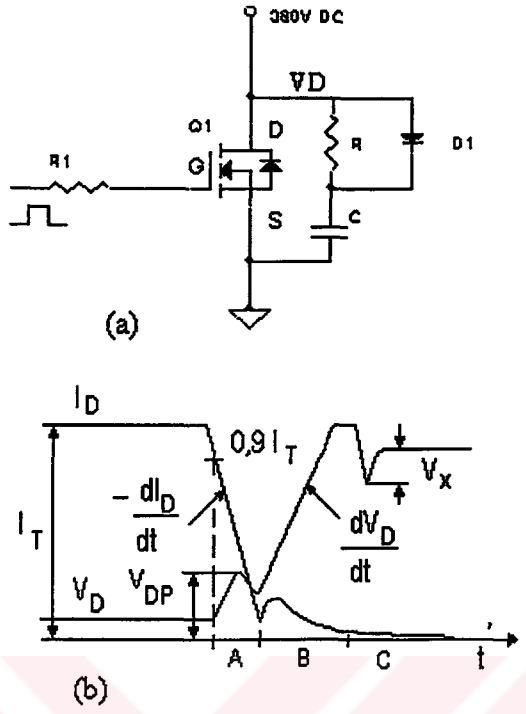
Şekil 6.11 Bir fazı süren sürücü devre

6.5 Snubber Devresi

Güç devrelerinde R ve C elemanlarından oluşan ve Tristör, IGBT, MOSFET gibi anahtarlama elemanlarına Şekil 6.12a'daki gibi parelel olarak bağlanan devreye snubber devresi adı verilir.

Snubber devresinin iki önemli fonksiyonu vardır. Birincisi MOSFET iletime geçtiğinde akımın yükselme oranını (di/dt) sınırlamaktır. Çünkü akımın çok hızlı yükselmesi aşırı ısınmaya sebep olacağından MOSFET'in yanması söz konusu olabilmektedir. Ancak bu uygulama için kullanılan sürücü devresinde MOSFET'ler motor faz sargısı ile seri bağlı olduklarından, MOSFET iletime geçtiğinde faz sargısının indüktansı tarafından akımın ani yükselişi sınırlanmaktadır. Bu yüzden MOSFET'nin iletime geçtiğinde di/dt'nin sınırlanması için snubber devresine ihtiyaç görülmemektedir.

Snubber devresinin ikinci fonksiyonu ise MOSFET kesime gittiğinde uçlarındaki gerilimin yükselme oranını (dv/dt) sınırlamaktır. Şekil-6.12a'da kullanılan snubber devresi görülmektedir. MOSFET kesime gittiğinde seri bağlı diyod ve kapasitör faz akımı için bir anlık bir yol oluşturmaktadırlar. MOSFET iletimde iken kapasitör uçlarında çok küçük bir gerilim mevcuttur. MOSFET kesime gittiğinde ise düz polarmalandırılmış hızlı toparlama diyodu (FRD) üzerinden kaynak gerilimine şarj olmaktadır. Kapasitör tam şarj olduğunda ise kaynaktan akım akışı kesilmektedir. Yani faz sargısındaki uyartım akımı MOSFET kesime gittiğinde ani olarak tamamen kesilmemekte, kapasitörün şarj süresince azalarak devam etmekte ve kapasitör kaynak gerilimine tam şarj olduğunda tamamen kesilmektedir[4].



Şekil 6.12 (a) Snubber devresinin bağlanışı

(b) Mosfetin kesime geçme zaman grafiği

Şekil 6.12a'daki snubber devresi şöyle çalışır. MOSFET kesime gittiğinde C kondansatörü D1 diyodu üzerinden şarj olur. Ve üzerinde (VD-Vdiyot) kadar bir voltaj şarj olur. MOSFET iletime geçtiğinde ise kondansatör R üzerinden deşarj olur. Böylece anlık olarak MOSFET üzerinde oluşan ani aşırı voltaj Snubber devresi tarafından emilir. Snubber devresinde kullanılan R değerinin seçimi çok önemlidir. Çünkü R'nin çok yüksek olması C'nin görevini yapamamasına, Çok düşük olması ise dI_D/dt akımının MOSFET'i bozmasına veya devrede voltaj salınımına neden olacaktır [11]. Devremizde anlık akım değişimi motor sargılarının endüktif etkisi nedeniyle önlendiğinden dikkate alınmamıştır. Fakat başka uygulamalar için bu durumun da dikkate alınması gerekir.

Şekil 6.12b'deki grafiğe bakılırsa A bölgesinde MOSFET kesim durumuna geçerken I_D MOSFET akımı çok hızlı bir şekilde dI_D/dt oranında düşmektedir. Bu anda ani bir voltaj yükselmesi (V_{DP}) oluşmaktadır. V_{DP} MOSFET anahtarlama çıkış voltajı üzerinde başlangıçta keskin bir tepe değeri oluşturur. Mümkün olduğunca bunun azaltılması gerekir. V_{DP} değerinin büyüklüğü:

a) Snubber devresinde kullanılan C kondansatörünün iç endüktansına (L_c) ve devre hatlarının oluşturacağı endüktans değerine (L_c)

b) Snubber devresinde kullanılan diyotun ileri polarma gerilim değerine (V_{FR}) bağlıdır. Bu iki değerin toplamı V_{DP} yi oluşturur.

$$V_{DP} = V_{FR} + \frac{dI_D}{dt}(L_s + L_c) \qquad (6.1)$$

$(L_s + L_c)$ nin maksimum değeri 0,2-0,4 μH den daha az olmalıdır.

Şekil 6.12b'de B bölgesine bakılırsa MOSFET kesime geçtiği anda snubber kondansatörü dV_D/dt oranında şarj olarak MOSFET'e anlık olarak aşırı voltaj düşmesini önlemektedir. C'nin değeri şu şekilde hesaplanır:

$$C = \frac{I_T}{dV_D/dt} \qquad (6.2)$$

C'nin değerini mümkün olduğunca büyük tutmak V_{DP} nin azalması dolayısı ile enerji kaybının azalması sağlanabilir.

MOSFET iletim durumuna geçtiği anda C kondansatörü R direnci üzerinden deşarj olduğundan. R direncinin değeri, MOSFET'in minimun iletim süresi içinde, C kondansatörü yükünü tamamen boşaltmaya izin verecek şekilde seçilmelidir.

Minimum iletim süresi en az $t_{ON} = 5\tau = 5RC$ olmalıdır. Buradan R değeri

$$R = \frac{1}{5C} x \text{Minimum MOSFET iletim suresi} \quad (6.3)$$

olarak hesaplanabilir. R'nin gücü ise

$$P_R = \frac{1}{2} C V_D^2 f \quad \text{olarak bulunur [11].} \quad (6.4)$$

Başka bir yaklaşımla snubber devresindeki C ve R değerleri şu şekilde hesaplanabilir. Bu yöntemde pratik olarak snubber devresinde yer alan C ve R değerlerini hesaplamak için MOSFET'in yükselme ve düşme (Şekil 6.13) zamanının bilinmesi gerekir. Yaklaşık yükselme(rise) ve düşme zamanının hesaplanması için için "Eş 6.1" kullanılır [12].

$$t_r \text{veya} \quad t_f = 2{,}2 R_1 C_{iss} \quad (6.5)$$

Burada :

t_r : MOSFET'in yükselme zamanı

t_f : MOSFET'in düşme zamanı

R_1 : Ön direnç

C_{iss} :MOSFET giriş kapasitesi

Şekil 6.12b'deki devre şöyle çalışır. MOSFET kesime geçtiği anda olduğu anda C kondansatörü D1 diyodu üzerinden şarj olur. Ve üzerinde (VD-Vdiyot) kadar bir voltaj şarj olur. MOSFET iletime geçtiği anda ise kondansatör R üzerinden deşarj olur. Bu arada MOSFET üzerinde kaybolması gereken güç Kondansatör üzerinde kaybolur.

Şekil 6.13'deki taralı enerji bölgesi kondansatör üzerinde açığa çıkan enerjiyi göstermektedir. Mosfetin kesime gitme süresi içinde (turn-off) kondansatör üzerindeki enerji "Eş 6.2" deki gibi yazılabilir.

$$E = \frac{C V_D^2}{2} = \frac{I_D V_D (t_r + t_{f)}}{2} \quad (6.6)$$

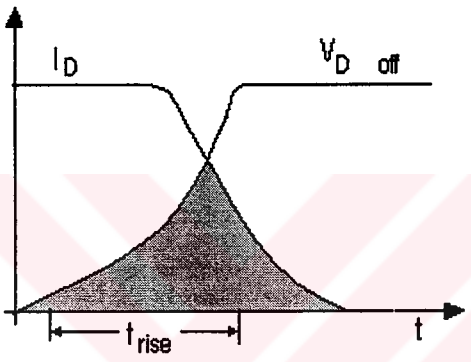
I_D : Maksimum drain akımı

V_D : Maksimum drain-source voltajı

"Eş 6.2" çözülürse snubber devresindeki kondansatörün değeri şöyle hesaplanır.

$$C = \frac{I_D(t_r + t_f)}{V_D} \qquad (6.7)$$

Temel devre teorisinden bilindiği üzere Kondansatör R üzerinden 5τ luk bir zaman içerisinde tamamen boşalır. MOSFET iletime geçtiğinde tekrar kesime



Şekil 6.13 MOSFET yaklaşık kesime geçme grafiği

geçmeden tamamen boşalması (mimimum iletim süresi) için geçen sürenin 3τ olduğu farz edilerek R değeri

$$R = \frac{t_{ON}}{3C} \qquad (6.8)$$

olarak hesaplanabilir. Boşalma akımı ise "Eş 6.5" ile hesaplanır.

$$I_{boş} = \frac{V_D}{R} \qquad (6.9)$$

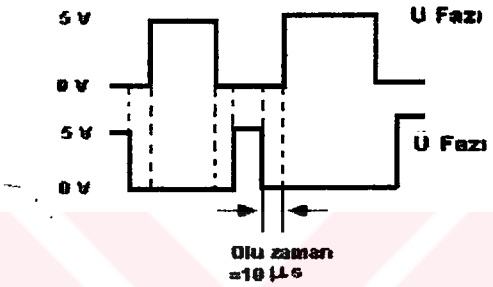
Eğer Direnç değeri çok küçük ve $I_{boş} > 0{,}25\ I_D$ ise R değeri daha büyük bir değer yeniden seçilerek hesaplamalar yeniden yapılmalıdır. Ve son olarak direncin gücü "Eş 6.4" deki gibidir.

Snubber devresinde en önemli nokta minimum iletim süresi t_{ON} içinde kondansatörün tamamen boşalmasıdır. Bunu etkileyen eleman ise R ve C

elemanlarıdır. R elemanı akımı sınırlarken C elemanı voltajın yükselme oranını sınırlamaktadır [12].

6.6 Ölü Zaman Durumu

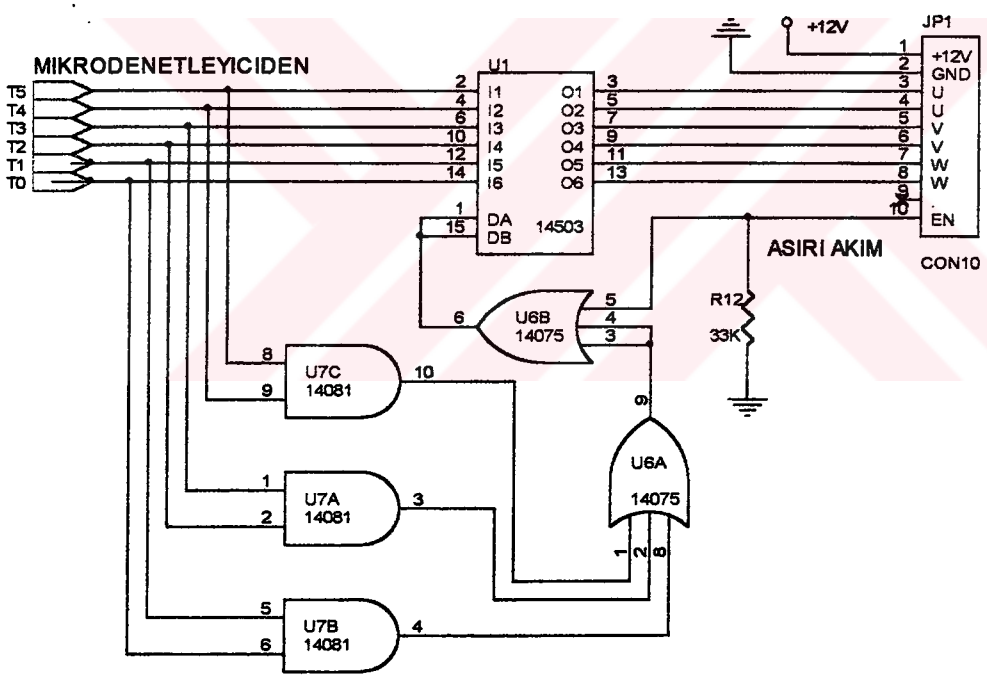
Eğer aynı kolun alt ve üst MOSFET'leri aynı anda iletime geçecek olursa bu kol kısa devre olur. Bu durum daha çok üst anahtarlama elemanı işlevini



Şekil 6.14 Ölü zamanın gösterimi

tamamlayıp alt anahtarlama elemanı iletime geçeceği zaman ortaya çıkar. geçiş anında anahtarlama elemanlarından birinde bir gecikme olursa çok kısa bir süre fazın kısa devre olmasına neden olabilir. Bu durumu önlemek için Geçiş anlarında her iki anahtarlama elemanının da kesimde olduğu kısa süreler koymak gerekir . Her iki anahtarlama elemanının kesimde olduğu anlara ölü zaman (dead time) denir. Bu durum Şekil 6.14'de görülmektedir. Devremizde ölü zamanın sağlanması simülatör programı tarafından yazılımla yapılmıştır. Yani tablo değerleri içinde ölü zaman olayı dikkate alınmıştır. Böylece ek bir donanıma gerek duyulmamıştır.

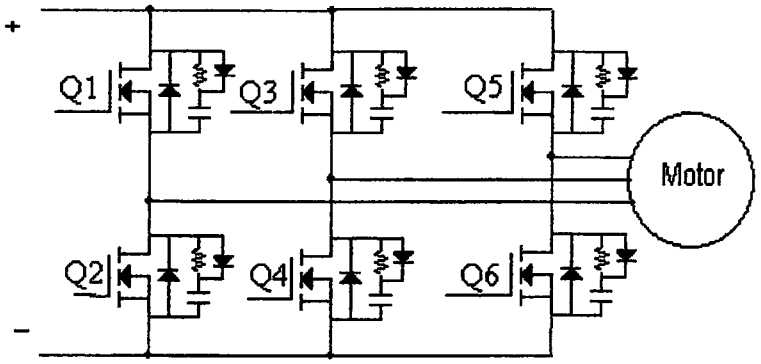
Mikrodenetleyici çıkışları ilk açılışta resetleme işlemi gerçekleşip program işleyinceye kadar çıkışları lojik 1 yapmaktadır. Bu durumda ilk güç verme anında MOSFET'ler sürülmekte ve fazların kısa devre olmasına neden olmaktadır. Bunu önlemek için Şekil 6.15'deki devre kullanılmıştır. Bu devre aynı zamanda çalışma anında kısa devre olma ihtimalinde de MOSFET'leri korumaktadır. Aynı koldaki MOSFET'leri sürecek olan mikrodenetleyici çıkışları aynı anda 1 olursa bu durum 14503 üç durumlu tamponun yetki girişini (DA) etkiler ve tampon çıkışları yüksek empedans durumuna geçer. Böylece MOSFET'ler devre dışı kalarak açık devre durumuna gelirler.



Şekil 6.15 Kısa devre önleme devresi

6.7 İnverter Devresi

İnverter devresi altı adet anahtarlama elemanından oluşur. Anahtarlama elemanı olarak MOSFET veya IGBT kullanılabilir.



Şekil 6.16 İnverter devresi

Bu elemanların tipik özelllikleri şu şekildedir.

Anahtarlama gate voltajı ile sağlanır.

Orta ve yüksek (1200 V-400A) seviye güç devrelerinde kullanılabilir.

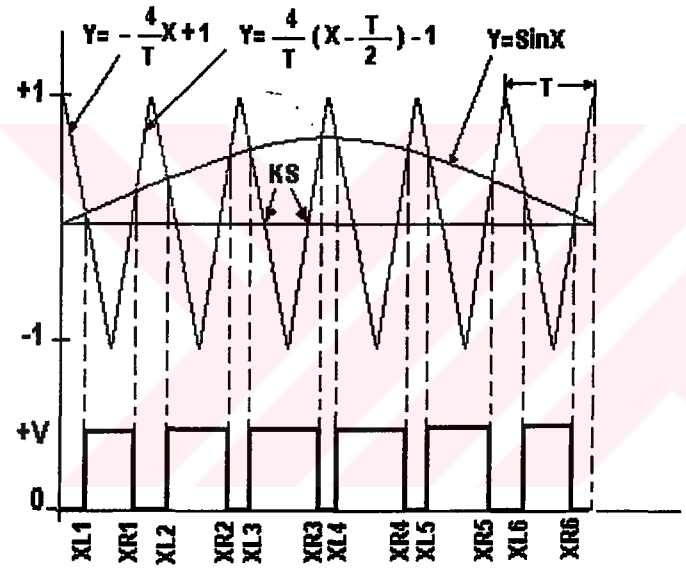
Yüksek frekanslarda çalışabilir(1 Mhz).

Mikroişlemcilere bağlantısı basittir.

İnverter devresi DC sinyali AC sinyale çevirir. 6 adımlı 3 fazlı invertere uygun devre şeması Şekil 6.16'de görülmektedir. İnverter çıkışında 3 fazlı AC sinyalin nasıl elde edildiği Bölüm 4'de anlatılmıştır.

7. SİNÜSOİDAL PWM ANAHTARLAMA SİNYALLERİNİN ÜRETİLMESİ

Sinüsoidal PWM sinyalleri Bir üçgen dalga ile sinüs dalgasının karşılaştırılması ile elde edilir. Bunun için bu iki sinyalin kesim noktaları bulunur. Daha sonra bir birinden çıkartılarak PWM dalganın yüksek ve düşük seviyede kaldığı açı değerleri hesaplanır. Bu açı değerleri zaman ortamına çevrilerek PWM dalganın süreleri hesaplanır. Şekil 7.1'den yararlanarak bu hesaplamalar yapılabilir.



Şekil 7.1 Sinüzoidal PWM'in elde edilmesi

Kesim noktalarının hesaplanması için Newton–Raphson yönteminden yararlanılabilir. Newton Raphson yönteminde yaklaşık kök şöyle bulunur. Kökü hesaplanacak fonksiyon f(x) ve türevi $f'(x)$ ise denklemin yaklaşık kökü

$$X_1 = X_0 + \frac{f(X_0)}{f'(X_0)} \qquad (7.1)$$

ile ifade edilir. Daha sonra

$$|X_1 - X_0| \leq \varepsilon \qquad (7.2)$$

Hesaplanarak bu değer kabul edilen hata (epsilon) değerinden küçük ise X_1 in verilen fonksiyonun kökü olduğu kabul edilir. Newton raphson yöntemi ile üçgen dalganın kesim noktaları ise şu şekilde bulunur.

Önce ücgen dalganın X'e bağlı denklemi ifade edilir. Üçgen dalga iki doğrunun birleşmesinden meydana gelmiştir. Üçgen dalganın maksimum tepe değerleri +1 ve -1 olduğuna göre Üçgen dalganın sola yatık doğrusunun denklemi

$$Y = -\frac{4}{T}(X - C) + 1 \qquad (7.3)$$

ile ifade edilebilir. Sağa yatık doğrunun denklemi ise

$$Y = \frac{4}{T}(X - \frac{T}{2} - C) - 1 \qquad (7.4)$$

ile ifade edilir. Sinüs dalgası ise

$$Y = M.SinX \qquad (7.5)$$

ile gösterilir.

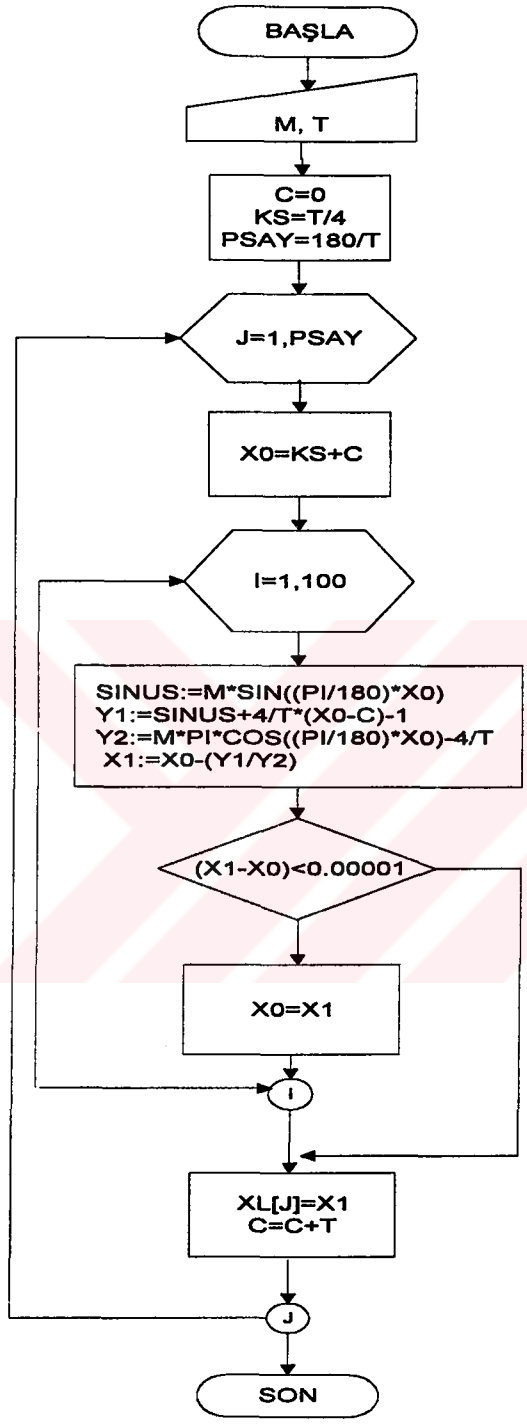
XL değerlerini hesaplamak için "Eş 7.5" ve "Eş 7.3" denklemleri birbirinden çıkarılarak yeni bir denklem elde edilir. Bu denklemde bulunan kökler kesim noktalarının XL değerlerini oluşturur.

$$Y = M.SinX + \frac{4}{T}(X - C) - 1 \qquad (7.6)$$

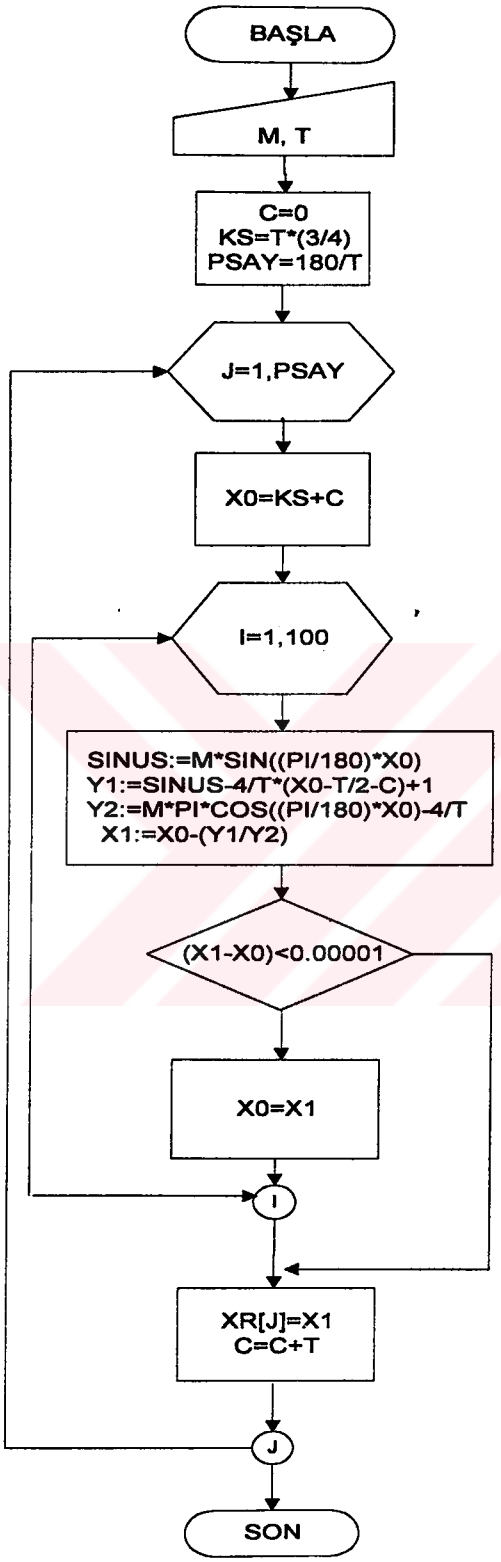
XR değerlerini hesaplamak için "Eş 7.5" ve "Eş 7.4" denklemleri birbirinden çıkarılarak yeni bir denklem elde edilir. Bu denklemde bulunan kökler ise kesim noktalarının XR değerlerini oluşturur.

$$Y = M.SinX - \frac{4}{T}(X - \frac{T}{2} - C) + 1 \qquad (7.7)$$

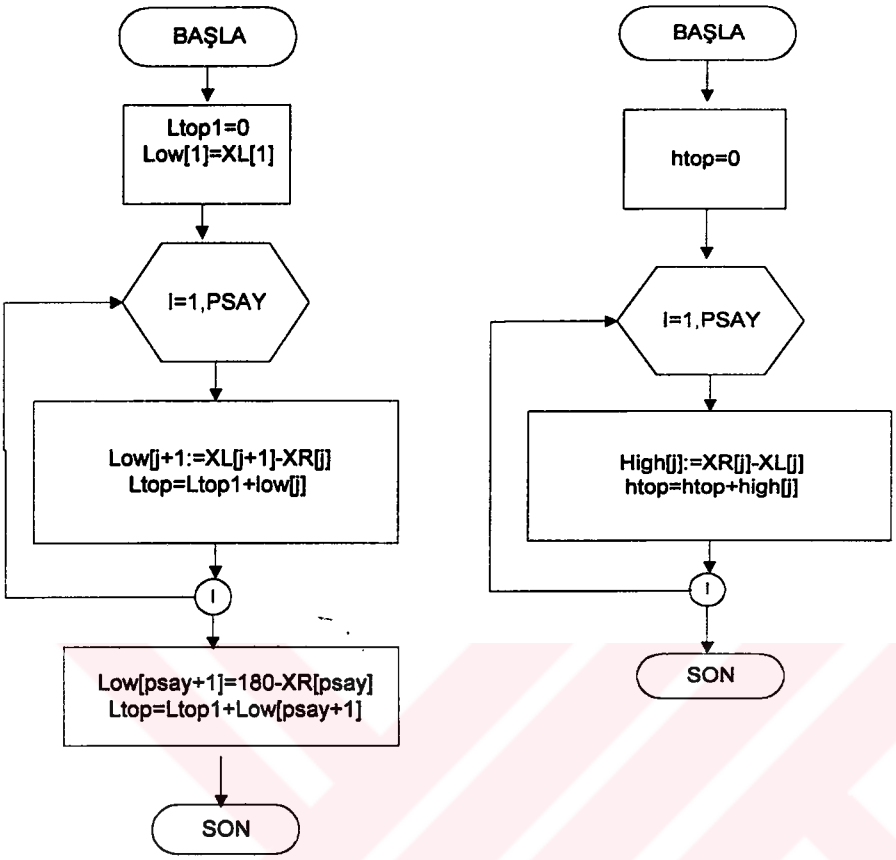
Denklemlerdeki C değeri bir sonraki doğru denklemini yazmak için öteleme değeridir. C değeri, üçgen dalganın peryodu (T) kadar ötelendiğinde, bir sonraki parelel doğrunun denklemi bulunur



Şekil 7.2 Sol kesim noktalarının bulunması



Şekil 7.3 Sağ kesim noktalarının bulunması



Şekil 7.4 Sinüs dalgasının düşük ve yüksekte kaldığı sürelerin hesaplanması

Sinüs dalgasının denkleminde yer alan M değeri ise sinüs dalgasının maksimum değerinin üçgen dalganın maksimum değerine oranını ifade etmektedir. Bu değer ayarlanarak PWM dalgadaki Yüksek ve Düşük seviye süreleri değiştirilebilir. PWM dalganın sol ve sağ kesim noktalarını bulmak için yazılan programın akış şeması Şekil 7.2 ve Şekil 7.3'de verilmiştir.

Burada kullanılan sembollerin anlamları ise şöyledir:

M: Modülasyon indeksi. sinüs dalganın maksimum değerinin üçgen dalganın maksimum değerine oranı.

T: Üçgen dalganın periyodu

C: Üçgen dalganın ötelenme açısı

KS: Üçgen dalganın x eksenini kestiği nokta

PSAY : 180 derece içindeki pals sayısı

X0 : Kökün varsayılan başlangıç değeri

X1: Bulunan kök

Y1: Fonksiyon

Y2:Fonksiyonun türevi

XL: Sol kesim noktası

XR: Sağ kesim noktası

Low : PWM dalganın düşük de kaldığı açısal değerler.

High : PWM dalganın yüksek de kaldığı açısal değerler

Ltop=Düşük seviyelerin toplamı

Htop=Yüksek seviyelerin toplamı

Sinüs dalgasının düşük ve yüksek te kaldığı süreler ise şu şekilde hesaplanır (Şekil 7.4). Yüksek ve düşük seviye açısal süreler bulunduktan sonra PWM dalganın etkin değeri hesaplanır. Herhangi bir dalganın ortalama ve etkin değerleri şu şekilde hesaplanır [13].

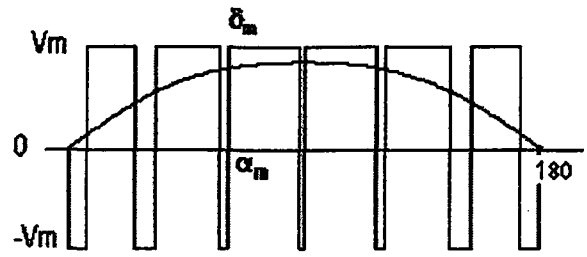
$$V_{ort} = \frac{1}{T}\int V(t).dt \qquad (7.8)$$

$$V_{et} = \sqrt{\frac{1}{T}\int V(t)^2.dt} \qquad (7.9)$$

Sinüs dalgasının etkin ve ortalama değerleri değeri hesaplanırsa

$$V_{et} = 0.707\ V_{max} \qquad V_{ort} = 0.636 V_{max} \qquad (7.10)$$

sonucu elde edilir. PWM dalganın etkin değeri, maksimum değer 1 volt alındığından, yüksek seviyelerin toplamının peryoda bölümünden elde edilen sonucun kare kökü alınarak bulunur.



Şekil 7.5 Sinüsoidal PWM dalga

Şekil 7.5'e göre PWM dalganın etkin ve ortalama değerleri

$$V_{et} = [\sum_{m=1}^{Psay} (-1)^m \frac{V_m^2 \delta_m}{T}]^{1/2} \qquad V_{ort} = \sum_{m=1}^{Psay} (-1)^m \frac{V_m \delta_m}{T} \qquad (7.11)$$

ile ifade edilir. Hesaplama kolaylığı için $V_m = 1$ ve T=180 derece için etkin ve ortalama değerler

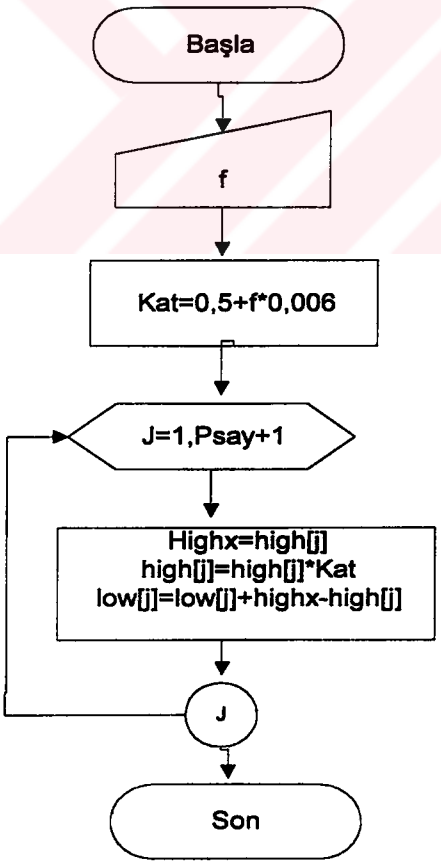
$$V_{et} = [\sum_{m=1}^{Psay} (-1)^m \frac{\delta_m}{180}]^{1/2} \qquad V_{ort} = \sum_{m=1}^{Psay} (-1)^m \frac{\delta_m}{180}$$ yazılabilir [14]. (7.12)

V/F oranın sabit kalması için frekans bilgisinin de hesaplara katılması gerekir. F=50 Hz, V=380v normal çalışma koşulları olarak kabul edildiğinde Frekansın F1=40 Hz'e düşürüldüğü varsayılırsa V/F in sabit kalması için yüksek seviyelerin F1/F katsayısı ile çarpılması ve eski değerinden çıkarılarak farkın düşük seviyeye eklenmesi gerekir. Aynı işlem Bir fazı oluşturan alt ve üst anahtarlama elemanları göz önüne alınarak yapılmalıdır. Teorik olarak F1/F katsayısı kadar yüksek seviyelerin düşürülmesi pratikte uygulanamamaktadır. Çünkü Şekil 3.8'e bakıldığında motor ilk kalkınma anında belli bir eşik gerilimi istemektedir.Bunun yerine Frekansın düşmesi durumunda voltajın düşmesi veya frekansın yükselmesi durumunda voltajın yükselmesi gerektiği(Momentin her frekansta sabit kalması için) düşünülerek frekansa bağlı bir katsayı ile orantılı olarak V/F nin hesaba katılması gereklidir. Bunun için yapılan deneylerde etkin değerin en azından 0,707 Vm

olması gerektiği görüldüğünden V/F oranı için yüksek seviyelere uygulanacak katsayı,

$Kat = 0{,}5 + f.0{,}006$ olarak kabul edilmiştir (Şekil 7.6).

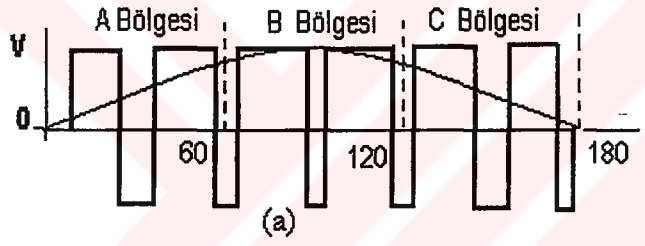
Bu katsayı ile tüm yüksek seviyeler çarpılır ve bir periyot içindeki her pals için uygulanırsa V/F'e uygun yeni yüksek ve düşük değerler bulunmuş olur. Böylece frekans düştükçe etkin değerin düşmesi, yükseldikçe ise etkin değerin yükselmesi sağlanmıştır. Burada 0.006 katsayısı gerilim /frekans oranı sabitesidir.



Şekil 7.6 V/F oranı için düşük ve yüksek seviye değerlerinin belirlenmesi

Bu işlemler yapıldıktan sonra Yeni kesim noktaları XL ve XR değerleri oluşturulur. XL ve XR değerlerinden yararlanılarak Low[j] ve High[j] ler hesaplanır. Low[j] ve High[j] 1000 değeri ile çarpılarak büyütme işlemi yapılır ve içleri durumlarına göre 0 veya 1 ile doldururlur. Bu durumlardan yararlanılarak portlara gönderilmesi gereken değerler için değer tablosu oluşturulur.

Şekil 7.7a'ye bakılacak olursa bir sinus dalgası 60'ar derecelik üç bölüme ayrılmıştır. Buna göre PWM dalgalar da üç bölüme ayrılmıştır. Bu şekle göre Üç faz için portlara gönderilmesi gereken dalga biçimleri Şekil 7.7b' de görülmektedir.



(a)

Zaman aralıkları

	T1	T2	T3	T4	T5	T6
Q1	A	B	C	$\overline{A}$	$\overline{B}$	$\overline{C}$
Q2	$\overline{A}$	$\overline{B}$	$\overline{C}$	A	B	C
Q3	$\overline{B}$	$\overline{C}$	A	B	C	$\overline{A}$
Q4	B	C	$\overline{A}$	$\overline{B}$	$\overline{C}$	A
Q5	C	$\overline{A}$	$\overline{B}$	$\overline{C}$	A	B
Q6	$\overline{C}$	A	B	C	$\overline{A}$	$\overline{B}$

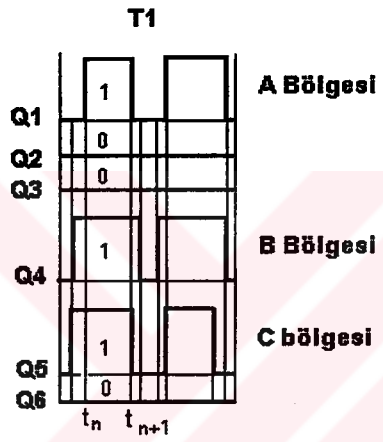
(b)

Şekil 7.7 Üç fazlı PWM dalganın üretilmesi

(a) PWM dalganın bölümlenmesi

(b) Anahtarlama elemanlarının durumları

Örneğin T1 zaman aralığında Q1 elemanına A bölgesi Q2 Elemanına 0(Çalışmıyor), Q3 elemanına 0, Q4 elemanına B , Q5 elemanına C bölgesi, Q6 elemanına 0 dalgaları gönderilir. Dikkat edilirse bir zaman diliminde üç farklı bölge çeşitli anahtarlama elemanlarına gönderilmektedir. Aynı zaman diliminde portlara gönderilmesi gereken farklı zaman aralıkları vardır. Bu zaman aralığını tek değere düşürmek için bu üç bölgenin eşlenikleri de düşünülerek alt alta kesim noktaları bulunur. Şekil 7.8 de eşlenikler düşünülmeksizin bu durum gösterilmiştir.



Şekil 7.8 Anahtarlama elemanlarına gönderilecek süreler ve durum değerlerinin hesaplanması

Kesim noktalarındaki değerine bakılır. Eğer kesim noktasındaki değeri yüksek ise 1 değilse 0 alınır. $t_{n+1} - t_n$ süresi kadar sırası ile Q1Q2Q3Q4Q5Q6 anahtarlama elemanlarına 100110 değeri gönderilir. Bu şekilde 180 derece tamamlanıncaya kadar süreler ve anahtarlama elemanlarının durum değerlerinden bir tablo oluşturulur. Süre değerleri hexadesimal olarak zamanlayıcı kaydedicisine konulmalıdır. Bu yüzden öncelikle hesaplanan değerler bir program yardımı ile hexadesimale çevrilir. Daha sonra ilk iki bayt süre, üçüncü bayt durum değeri olarak bir tablo oluştururlur. Bu tablo her frekans için farklıdır.

7.1 Mikrodenetleyici Kontrol Yazılımının Gerçekleştirilmesi

Mikrodenetleyici kontrol yazılımı, birinci mikrodenetleyici kontrol yazılımı ve ikinci mikrodenetleyici kontrol yazılımı olmak üzere iki kısımdan meydana gelmektedir. Birinci mikrodenetleyiciye bir tane tuştakımı ile, girilen değerleri ekranda göstermek için kullanılan bir LCD display bağlıdır. Şekil 7.9'da verilen akış şemasında birinci mikrodenetleyici kontrol yazılımı akış şemaları verilmiştir. Önce yazılımda kullanılan değişkenler tanımlanmaktadır. Daha sonra bu değişkenlere ilk değerler verilmekte ve ekranda gösterilmektedir. Daha sonra Main programı ile sonsuz bir döngüye girmektedir. Main programı içinde sonsuz döngüde mikrodenetleyici işlerken tuş takımından herhangi bir tuşa basıldığında bu KEYIN kesme altprogramını çalıştırmaktadır. Tuş takımında görev verilen tuşların fonksiyonları Çizelge 7.2'de açıklanmıştır.

KEYIN kesme altprogramı (Şekil 7.10) ise hangi tuşa basıldığını tesbit edip ona göre frekans veya kutup bilgisini oluşturmakta ve ENTER tuşuna basıldığında ise ayarlanan frekans değeri ikinci mikrodenetleyicinin INT1 ucuna kesme üretilerek P2 portundan gönderilmektedir.

Çizelge 7.2 Devrede kullanılan tuşların fonksiyonları

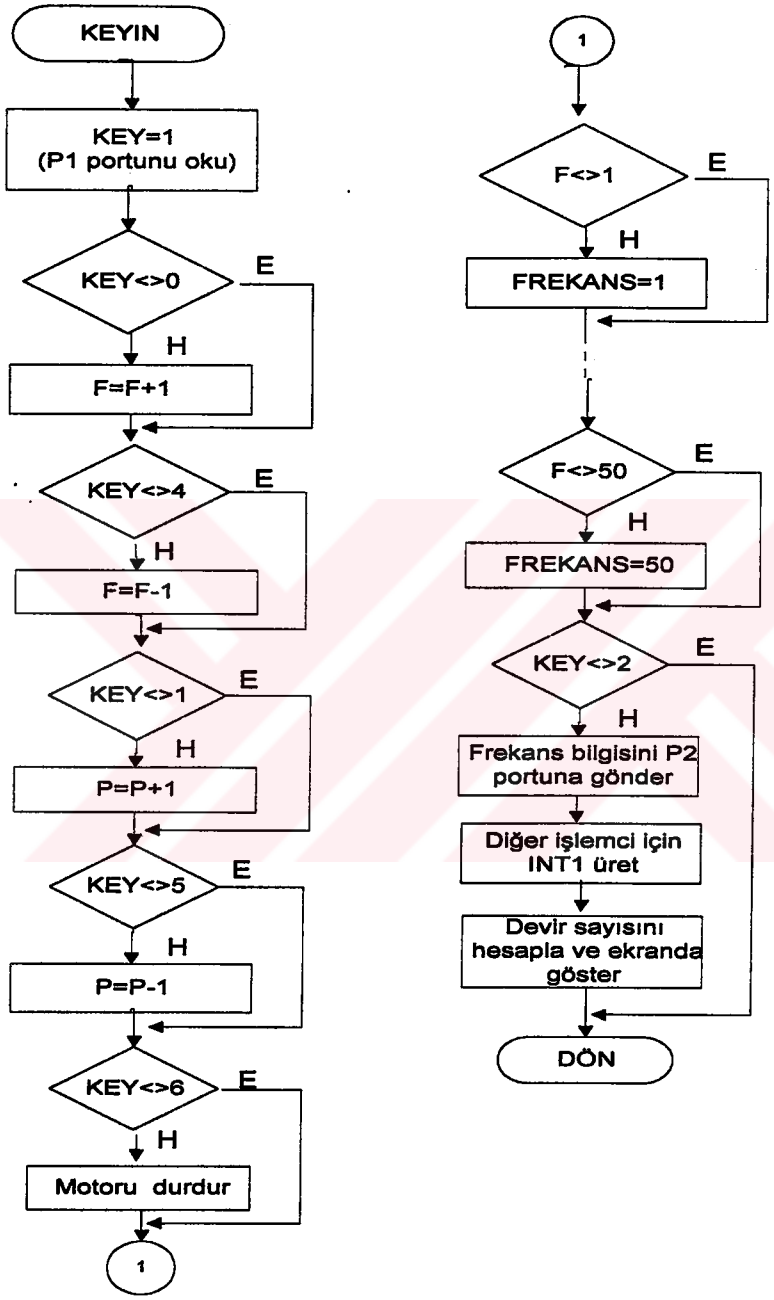
0	Frekans değerini bir artırır.
4	Frekans değerini bir azaltır.
1	Kutup sayısını bir artırır.
5	Kutup sayısını bir azaltır
2	Enter
6	Motoru durdur.

Daha sonra ayarlanan frekans değerine göre senkron devir sayısı hesaplanarak ekranda gösterilmektedir. ENTER tuşuna basıldıktan sonra ancak ayarlanan değerler işlem görmektedir. Bu durumda ekranda OK yazısı görülebilir.

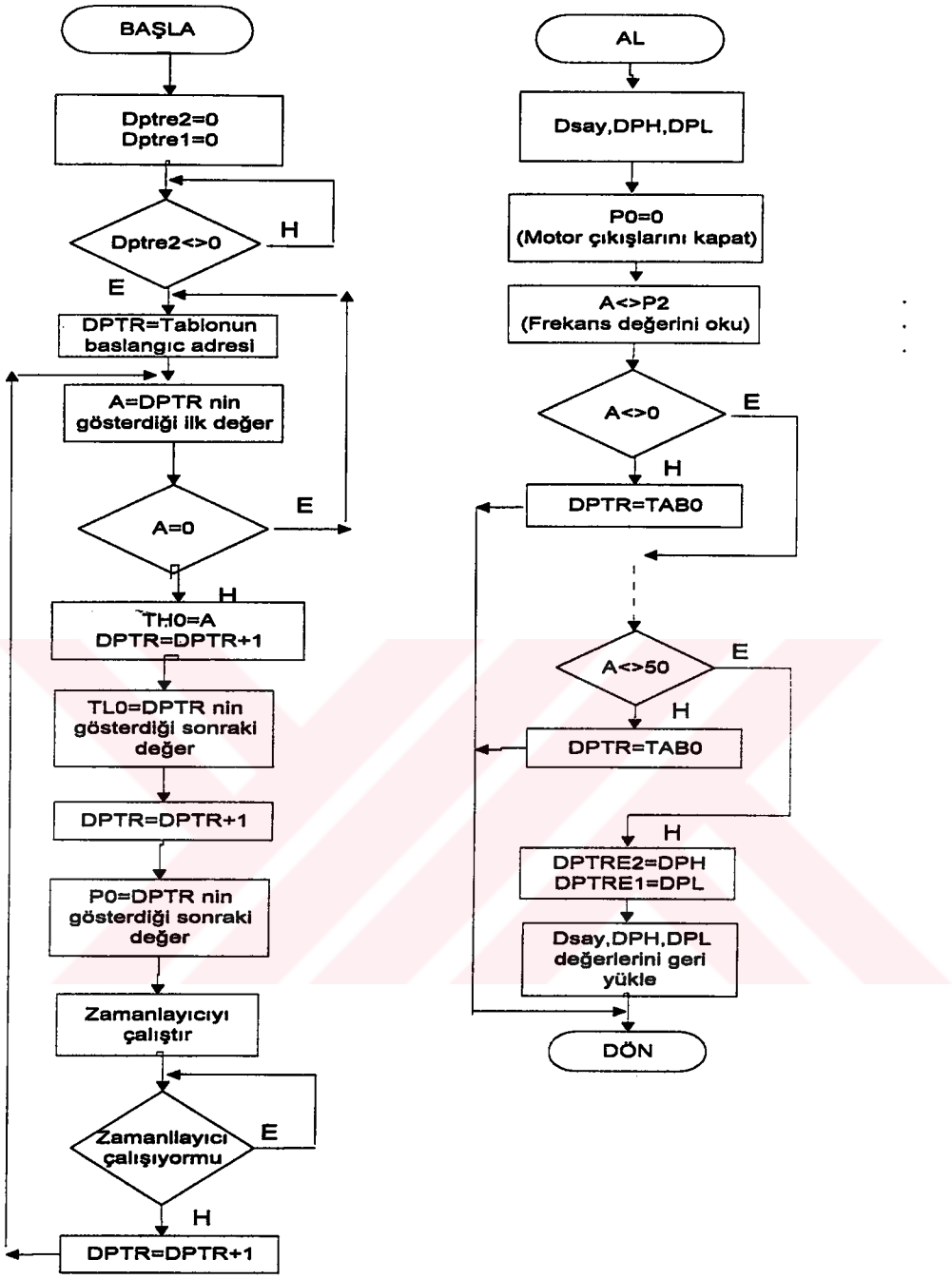
Şekil 7.9 Birinci mikrodenetleyici ana program akış şeması

İkinci mikrodenetleyici yazılımı iki kısımdan oluşmaktadır. İkinci mikrodenetleyici de 1-50 arası frekanslar için oluşturulmuş değer tablosu vardır. Her bir tabloda farklı sayıda değer yer alabilir. Çünkü mfarklı frekanslarda farklı sayıda anahtarlama sinyali kullanılmıştır. Tabloda yer alan her üç değer birlikte bir bütün oluşturur. İlk iki bayt zamanlayıcı kaydedicilerini set etmek için zaman bilgisini oluştururken üçüncü bayt anahtarlama elemanlarının durumunu belirtir. Tablonun başlangıç adresi DPTR kaydedicisi ile alınır. Daha sonra değeri 1 artırılarak sırası ile veriler

okunur. Başlangıç anında motor durma pozisyonundadır. Bir frekans bilgisi geldiğinde AL kesme



Şekil 7.10 Tuş takımı ile frekans ve Kutup sayısı değerlerinin girilmesi ve ikinci işlemciye gönderilmesi



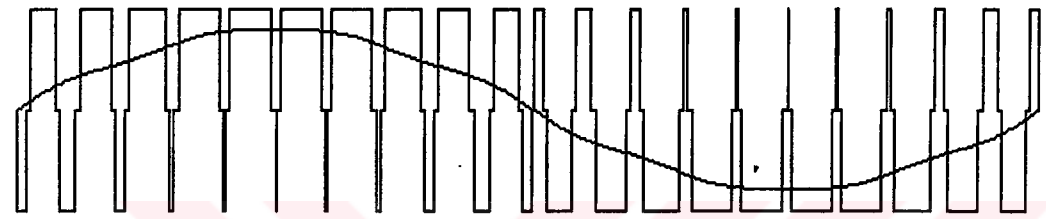
Şekil 7.11 İkinci mikrodenetleyici kontrol yazılımı akış şeması

hizmet programı (Şekil 7.11) vasıtası ile bu bilgi alınır. Frekans değerine göre tablonun başlangıç adresi DPTR kaydedicisine yüklenir. Daha sonra DPTR nin DPH kısmı DPTRE2 değişkenine ve DPL kısmı DPTRE1 Değişkenine

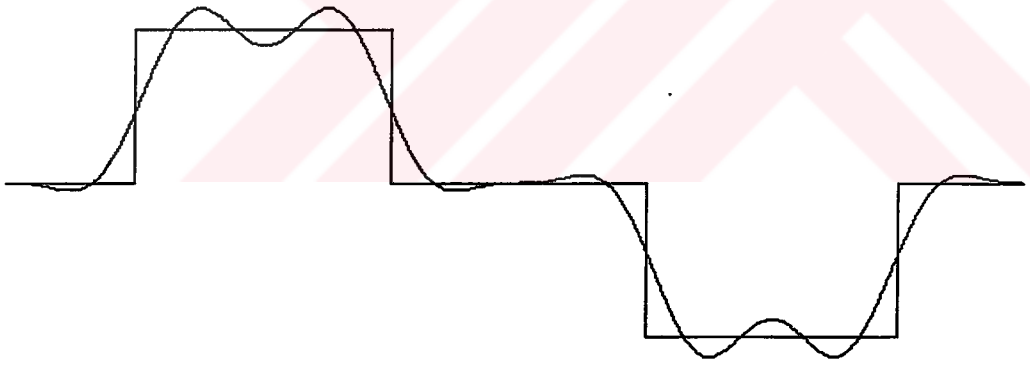
atanır. Ana programda (Şekil 7.11) program ilk çalışmaya başladığında DPTRE2 sürekli kontrol edilir. Eğer DPTRE2 nin içeriği değişmişse yeni bir frekans bilgisi girilmiş demektir. Bu durumda DPH Kaydedicisine DPTRE2 ve DPL kaydedicisine ise DPTRE1 aktarılır. Böylece tablodan ilk iki değer okunur ve sırası ile TH0,TL0 set edilir. Üçüncü bayt okunur P0 portuna gönderilererek motor anahtarlama elemanları çalıştırılır. Zamanlayıcı durduğu anda bir sonraki değerler okunarak aynı işlemler tekrarlanır. Tablonun son değeri 000h tablo sonunu göstermektedir. 000h değeri ile karşılaşıldığında döngü sona erer. Tablonun başlangıç adresi tekrar yüklenir ve işlemler tekrar edilir.

8. SİMÜLASYON VE DENEY SONUÇLARI

Sinüsoidal PWM ve kare dalganın harmonik bileşenleri hesaplanarak Çizelge 8.1'de verilmiştir. Şekil 8.1a'daki grafikte f=50Hz ve pals sayısı 20 için PWM vePWM dalganın ters fourier dönüşümü çizdirilmiştir. Şekil 8.1b'de simetrik kare dalga ve ters fourier dönüşümü ilk 7 katsayı için çizdirilmiştir. Buradan harmonik bileşenlerin etkisi daha iyi görülmektedir.



(a)



(b)

Şekil 8.1 (a) PWM dalganın harmonik bileşenleri dalga şekli

(b) Kare dalganın harmonik bileşenleri dalga şekli

Çizelge 8.1 PWM ve kare dalganın harmonik bileşenleri

	PWM Harmonik	Kare Harmonik
1	0,8043	0,9003
3	0,0093	-0,3001
5	0,0174	-0,1801
7	0,0266	0,1286
9	0,0385	0,1000
11	0,0557	-0,0818
13	0,0833	-0,0693

Çizelge 8.1'de f=50 .Hz ve pals sayısı (m=20) için PWM ve kare dalganın harmonik bileşenlerinin değerleri görülmektedir. PWM dalga hep pozitif değerler alarak yumuşak bir yükselme ve inme eğilimi gösterirken kare dalga da negatif ve pozitif bölgelerde salınımlar yapmaktadır. Özellikle düşük frekanslarda harmonik bileşenlerin etkisi aşırı artmakta ve bozucu etkilerin daha da büyüdüğü görülmektedir.

PWM yöntemi ile inverter girişindeki DC gerilim değeri değiştirilmeksizin, inverter çıkış gerilimi, bir yarım periyot içinde uygun açılarda açılıp kapatılarak ayarlanır. Ayrıca bu yöntemle anahtarlama yapıldığından çıkış gerilimindeki yüksek frekanslı harmonik bileşenler(5,7,...) bastırılabilir.

PWM yöntemi ile inverter çıkış geriliminin ayarlanması modülasyon indeksi ile ilgilidir. Modülasyon indeksi referans sinyalin genliğinin , taşıyıcı sinyalin genliğine oranıdır.

$$M = \frac{V_r}{V_c} \qquad \text{Burada,} \qquad (8.1)$$

V_r: Referans dalga(sinüs) genliğini

V_c: Taşıyıcı dalga (üçgen) genliğini ifade etmektedir.

M'nin değeri ayarlanarak PWM dalganın yüksek ve düşük seviye süreleri ayarlanır. İnverter çıkışında düzgün bir sinüs akım şekli elde edebilmek için taşıyıcı dalganın frekansının, üçgen dalganın frekansına oranı 3 ve 3'ün katları olmalıdır. Taşıyıcı oranı,

$m = \frac{f_c}{f_r}$ olarak gösterilebilir. (8.2)

Burada,

f_c :Taşıyıcı dalga frekansını

f_r :Referans dalga frekansını ifade etmektedir.

Sinüsoidal PWM modülasyonu ile kare dalga gerilimde oluşan harmoniklerin neden olduğu ısınma ve moment salınımlarından bir ölçüde sakınılabilir. Taşıyıcı oranının yüksek tutulması halinde çıkış geriliminde yüksek mertebeden salınımlar oluşur. Çıkış akımı sinüse yakın olduğundan düşük hızlarda salınımlar oluşmaz ve düzgün bir dönüş sağlanır.

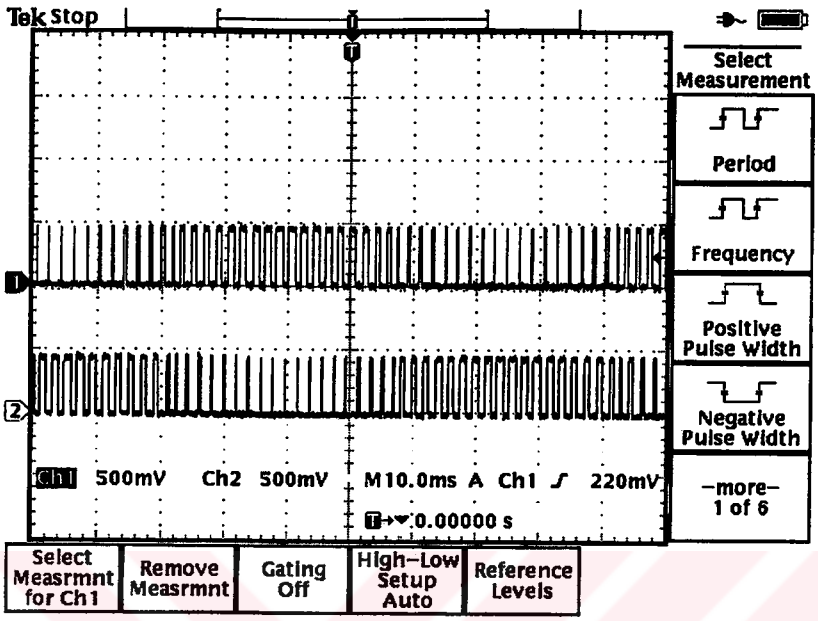
Tasarımı yapılan devre etiket değerleri çizelge 8.2'de verilen üçfazlı asenkron motorda denenmiştir.

Çizelge 8.2 Deney yapılan asenkron motor etiket değerleri

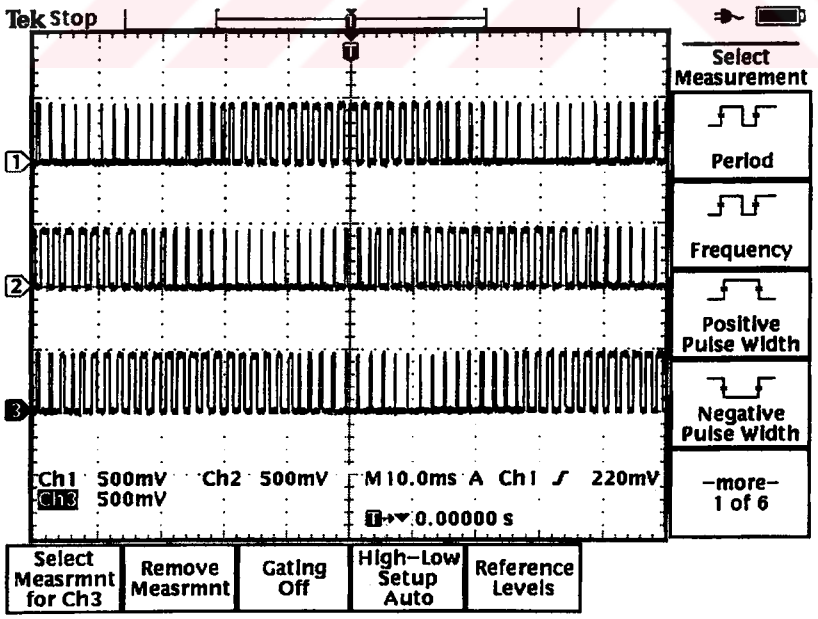
Çalışma gerilimi V	380Volt
Çalışma frekansı	50Hz
Çalışma akımı	0.27A
Devir sayısı	3000d/dk
Cosφ	0,78
Güç	0.09KW 1/8Hp
Bağlantı şekli	Yıldız

8.1 Deneyde Yapılan Gözlemler ve Alınan Değerler

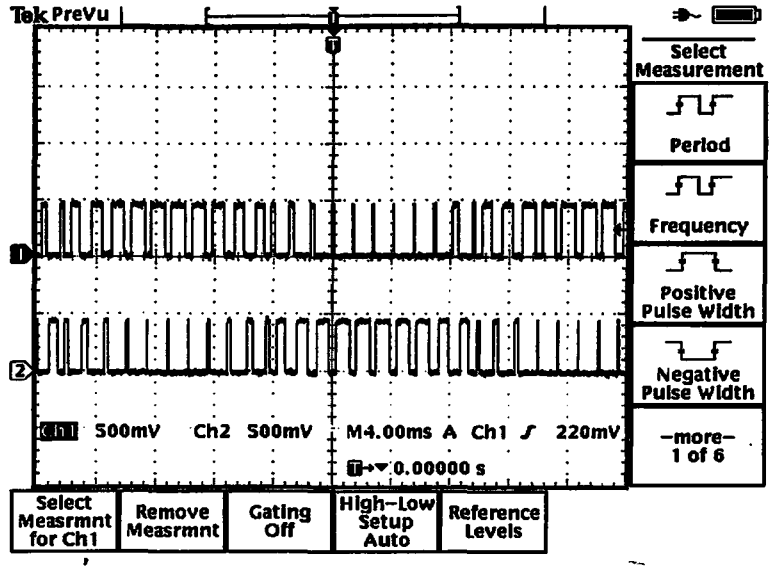
Devre tasarımında 1-60 Hz arası frekans değerleri kullanılmıştır. Buna göre 14, 35, 48, 52Hz frekans değerleri için alınan MOSFET'lerin gate sinyalleri ve çıkış akım dalga formları Şekil 8.2'den Şekil 8.11'e kadar gösterilmiştir. Şekillere dikkat edilirse frekans düştükçe 360 derece içindeki pals sayısı artırılmıştır. Aynı şekilde frekans yükseldikçe pals sayısı düşürülmüştür. Düzgün bir sinüs akım şekli elde edebilmek için pals sayısını anahtarlama elemanlarının izin verdiği sınırlar içinde, mümkün olduğunca artırmakta fayda vardır. Bir periyot içindeki frekans-pals sayısı çarpanının değerinin artırılması iyi bir sinüs dalgası elde etmek için gereklidir. Devrede kullanılan MOSFET'lerin küçük ve gücünün yetersiz olması nedeni ile 1Khz sınırı aşılamamıştır. Buna karşılık Simülatör yazılımı ve diğer devre tasarımları istenilen frekansta anahtarlama sinyallerinin üretilmesine izin vermektedir. Bu yüzden daha yüksek anahtarlama sinyallerine ulaşılabilmesi için hızlı MOSFET veya IGBT'lerin kullanılması gereklidir.



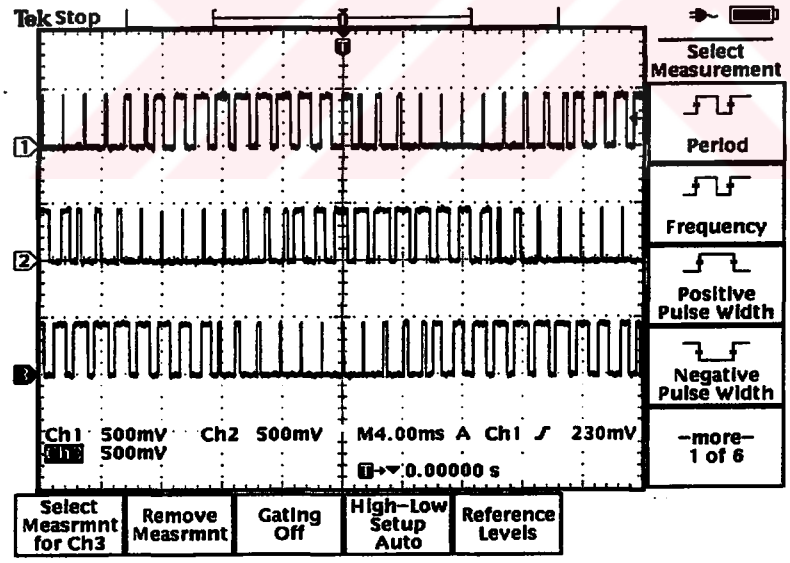
Şekil 8.2 F=14 Hz, m=36 iken bir fazın alt ve üst MOSFET'lerine gönderilen anahtarlama sinyalleri.



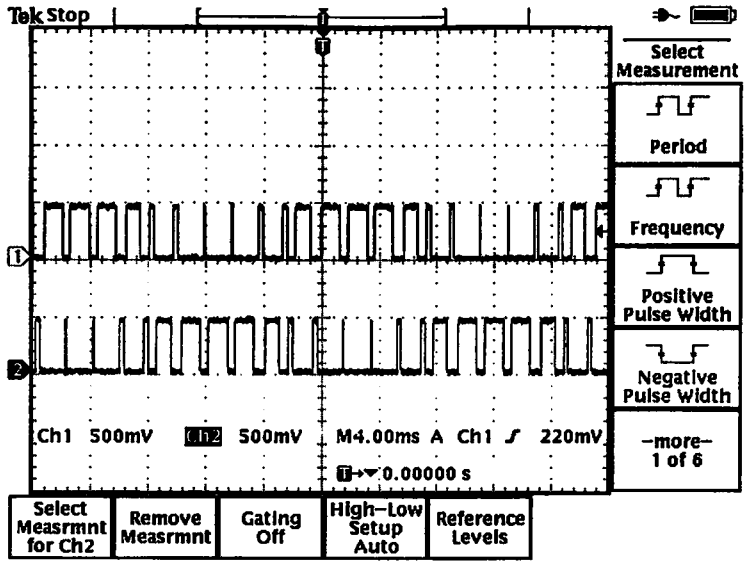
Şekil 8.3 F=14 Hz, m=36 iken üç fazın üst MOSFET'lerine gönderilen birbirlerinden 120 derece faz farklı anahtarlama sinyalleri.



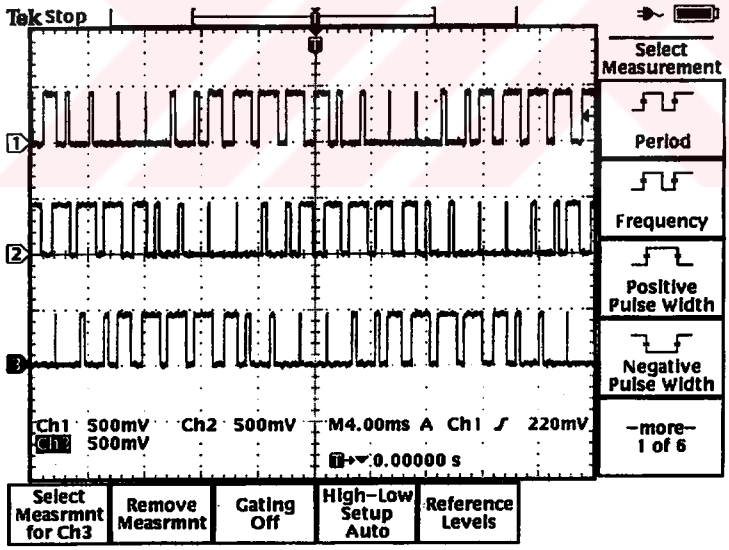
Şekil 8.4 F=35Hz, m=20 iken bir fazın alt ve üst MOSFET'lerine gönderilen anahtarlama sinyalleri.



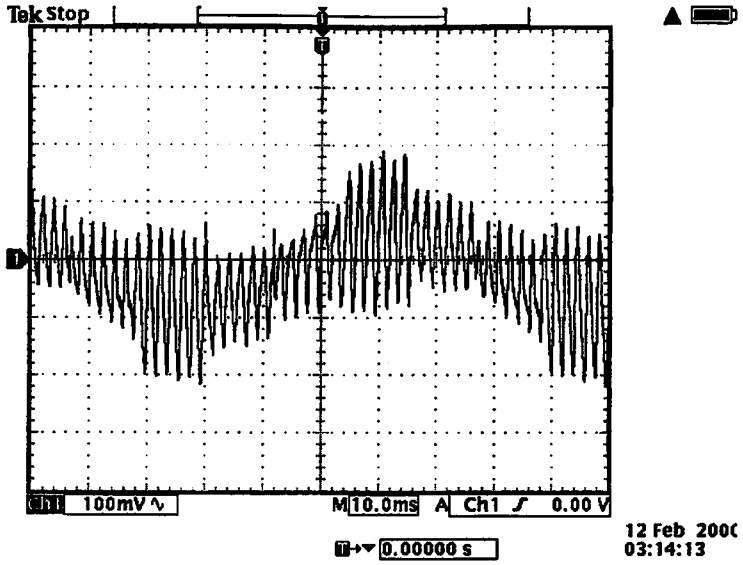
Şekil 8.5 F=35 Hz, m=20 iken üç fazın üst MOSFET'lerine gönderilen birbirlerinden 120 derece faz farklı anahtarlama sinyalleri.



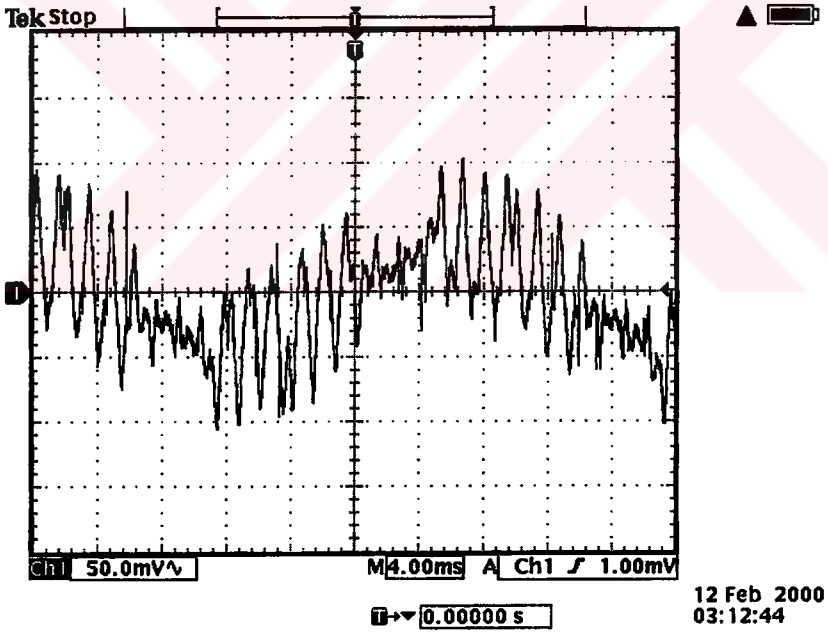
Şekil 8.6 F=52 Hz, m=10 iken bir fazın alt ve üst MOSFET'lerine gönderilen anahtarlama sinyalleri.



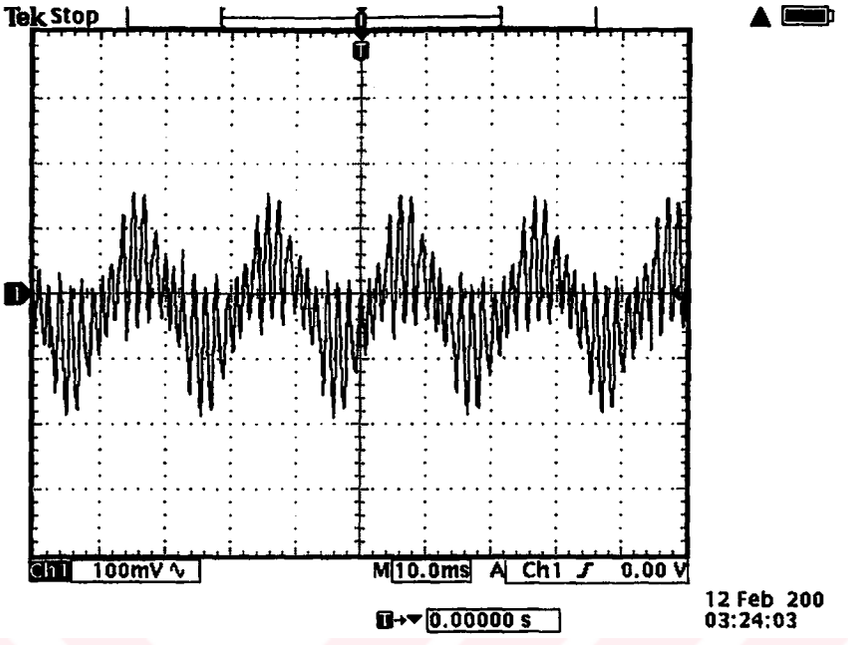
Şekil 8.7 F=52 Hz,.m=10 iken üç fazın üst MOSFET'lerine gönderilen birbirlerinden 120 derece faz farklı anahtarlama sinyalleri.



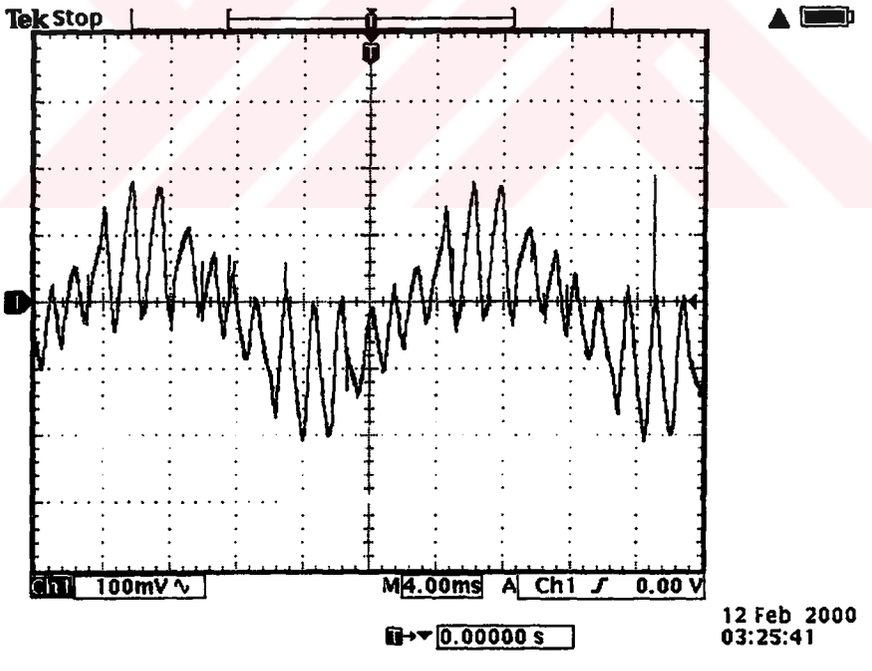
Şekil 8.8 F=14 Hz, m=36 ve Time/div=10ms iken bir fazın akım dalga formu



Şekil 8.9 F=35 Hz, m=20 ve Time/div=4ms iken bir fazın akım dalga formu



Şekil 8.10 F=48 Hz, m=12 ve Time/div=10ms iken bir fazın akım dalga formu



Şekil 8.11 F=48 Hz, m=12 ve Time/div=4ms iken bir fazın akım dalga formu

9. SONUÇ VE ÖNERİLER

Çok çeşitli AC motor hız kontrol teknikleri vardır. Burada en verimli hız kontrol yöntemlerinden biri olan Sinüsoidal PWM inverter kullanılarak AC motorun hız ayarının nasıl yapılabileceği üzerinde durulmuştur. Sinüsoidal PWM kullanımı ile motor kayıplarının azaltılması amaçlanmaktadır

Devre gerçekleştirilirken öncelikle üçgen ve sinüs dalgasının karşılaştırılarak sinüsoidal PWM dalganın elde edilmesi için gerekli hesaplamalar Newton-Raphson yöntemi ile Delphi programlama dili kullanılarak yapılmıştır. Ayrıca PWM dalganın fourier analizi ve mikrodenetleyicinin kullanacağı değerler tablosu da Delphi ile oluşturulmuş, elde edilen PWM dalga bilgisayar ekranında çizdirilmiştir. Böylece elde edilen PWM dalga formunun doğruluğu direk olarak test edilebilmiştir. Ayrıca indüksiyon motorda sabit moment elde edebilmek için gerekli olan V/f oranı dikkate alınmıştır

Hız kontrol devresi iki adet mikrodenetleyici kullanılarak gerçekleştirilmiştir. Bunun sebebi hız kontrolünün gerçek zaman saati içinde gerçekleştirilmesidir. Bir mikrodenetleyici ile LCD , tuş takımı gibi çevre cihazlar kontrol edilirken diğer mikrodenetleyici gerçek zaman içinde kontrol sinyallerini motora gönderir. Mikrodenetleyici kullanımı ile gerekli olan elemanlar oldukça azaltılmış, güvenli ve ucuz bir hız kontrol devresi gerçekleştirilmiştir.

Devre, aşırı akım kesme devresi ile korunmuş ve ölü zaman oluşması yazılımla gerçekleştirilmiştir. İnverter devresinde kullanılan MOSFET'leri aşırı voltajlardan korumak için snubber devresi kullanılmıştır.

Kontrol yazılımı hız ve 16 bit işlem gerektirmektedir. Bu yüzden hızlı ve 16 bitlik mikrodenetleyicilerin kullanılması yazılımı kolaylaştıracaktır. Motor

kontrolü için özel olarak tasarlanmış 16 bitlik intel firmasının ürettiği 8XC196 MC kontroleri kullanılabilir. Bu mikrokontroler içinde dalga formu üretici birim bulunmaktadır. Sinüsoidal PWM daha basit bir yazılımla hızlı bir şekilde üretilebilir. Bu mikrodenetleyiciyicinin pahalı oluşu bir dezavantaj oluşturmaktadır.

Hız kontrol devresi mikrodenetleyici ROM belleği içine yerleştirilen değer tablosunu kullanarak PWM dalgayı ürettiğinden biraz büyük ROM bellek gerektirmektedir. Devrede 60 farklı frekans için hız kontrolü yapılmaktadır. Daha fazla kademede hız kontrolü yapmak için daha büyük bellek gereklidir. Bu ise devreye harici ROM bellek takılarak veya dahili ROM belleği daha fazla olan bir mikrodenetleyici kullanarak sağlanabilir.

Sonuç olarak oldukça verimli ve ucuz maliyetli bir asenkron motor hız kontrol devresi gerçekleştirilmiştir.

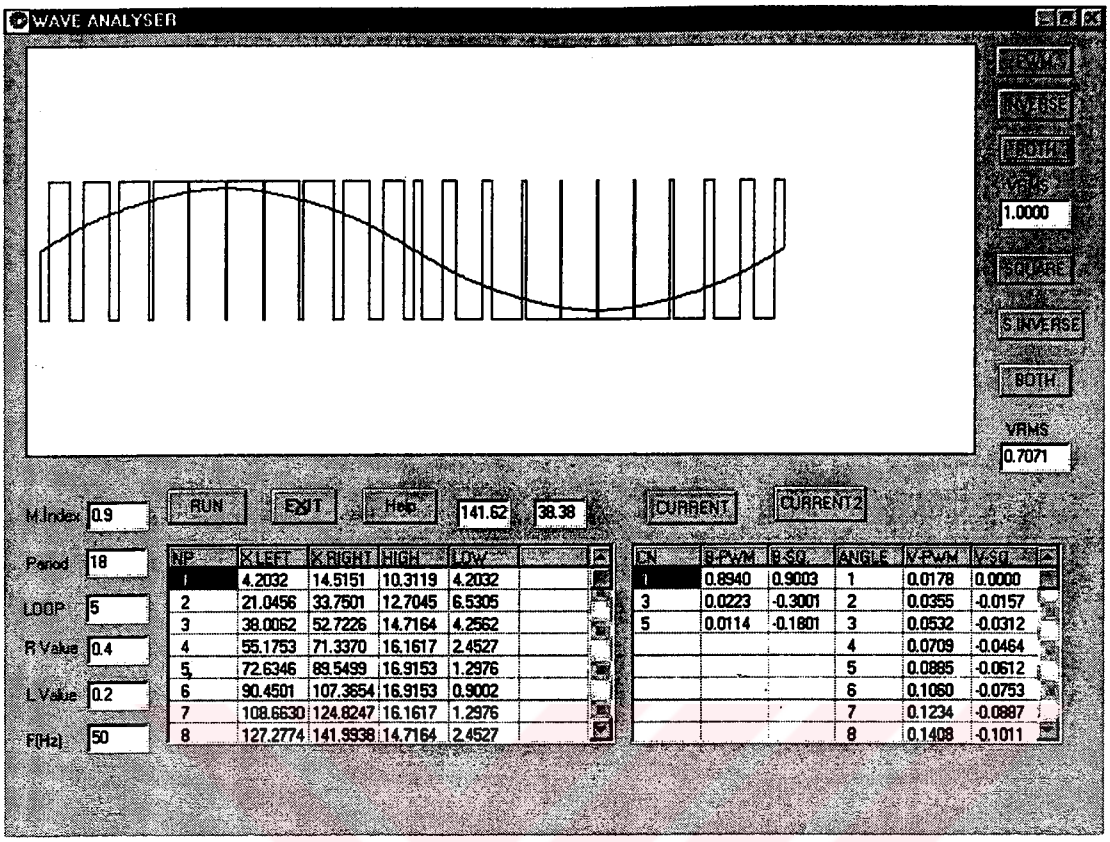
KAYNAKLAR

1. Saçkan, A. H., 1994, Asenkron Motorlar, **Birsen Yayınevi**, İstanbul.
2. Alexander, S.L., 1955, Theory of Alternating Current Machinary, **Mcgraw-Hill Book Company.**
3. Sarma,M. S.,1985, Electric Machines, Second Ed., **West Publishing Com.**,USA.
4. Bay,Ö. F., 1996, Anahtarlamalı Relüktans Motorun Bulanık Mantık Tabanlı Modellenmesi ve Kontrolu, Doktora tezi, Erciyes Üniversitesi
5. Sarıoğlu,M.K.,1983, asenkron Makinalar, **Çağlayan Kitabevi**, İstanbul.
6. Humphries,J.T., 1998, Motors and Controls, **Merrill Publishing Company**,USA.
7. Wildi,T., 1991, Electrical Machines Drives and Power Systems, **Prentice Hall**, New Jersey.
8. Gürdal,O., 1997, Güç Elektroniği, Ankara.
9. Gümüşkaya,H.,1998, Mikroişlemciler ve 8051 Ailesi, **Alfa yayınevi**, İstanbul.
10. Barnett, R.H., 1995, The Family of Microcontrollers, Prentice Hall, **New Jersey**.
11. Snubber diodes & other GTO Fast Diode Applications, 1988, **Marconi Electronic Devices Ltd.**
12. Chryssis, G.,1989, High-Frequency Switching Power Supply, **McGraww-Hill**, USA.
13. Edminister, J.A., 1972, Electric Circuits, **McGraww-Hill İnc.**
14. Rashid, M.H., 1993, Power Electronics, **Prentice Hall**, New Jersey.
15. Johnson,D.E., Johnson, J.R., Hilburn, J. L., 1992, Electric Circuit Analysis, **Prentice Hall**
16. Sinha,N.K., 1986, Microprocessor- Based Control Systems, **D. Reidell Publishing Company**, Holland.

17. Fitzgerald, A. E.,1983, Electric Machinary, **McGraw-Hill**
18. Kenjo,T., 1991, Electric Motor And Their Controls, **Oxfort University Press**, New York.
19. 8XC196 MC/MD/MH Microcontroler Users Manuel, 1996
20. AP-483 Application Example Using The 8XC196 MC/MD , 1993
21. İnverter Motor Controler Using 8Xc196 MC Microcontroler Design Guide Application Note , 1998

EK-1

SİNÜSOİDAL PWM TABLO DEĞERLERİNİ ÜRETEN VE DELPHİ PROGRAMLAMA DİLİ İLE YAPILAN SİMÜLATÖR PROGRAMI



Delphide yapılan Similatör programının ekran çıktısı

DELPHİ İLE YAPILAN SİMÜLATÖR PROGRAMI

```
unit pwm;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ComCtrls, ExtCtrls, Grids,math;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Edit1: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Edit3: TEdit;
    Button4: TButton;
    Edit2: TEdit;
    StringGrid1: TStringGrid;
    StringGrid2: TStringGrid;
    Button5: TButton;
    Edit4: TEdit;
    Edit5: TEdit;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Edit6: TEdit;
    Button6: TButton;
    Edit7: TEdit;
    Label7: TLabel;
    Shape1: TShape;
    Button7: TButton;
    Button8: TButton;
    BOTH: TButton;
    Label8: TLabel;
    edit8: TEdit;
    edit9: TEdit;
    Edit10: TEdit;
    CURRENT: TButton;
    CURRENT2: TButton;

    procedure Button4Click(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure Button6Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button7Click(Sender: TObject);
    procedure Button8Click(Sender: TObject);
    procedure BOTHClick(Sender: TObject);
    procedure CURRENTClick(Sender: TObject);
```

```
 procedure CURRENT2Click(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
end;


var
 Form1: TForm1;
 AKIM,AKIM2,KV,KB,High,Low,HighX,LowX,LowE,HighE,LOWE1,HIGHE1,LowEX,HighEX,
 KSL,KSR,v,b,XL,XR,XLE,XRE,XBL,XBR:ARRAY [1..1000] OF REAL;
 DiziT,Low1,High1,Low2,High2,XRR,XLL,XLLE,XRRE:Array[1..1000] of integer;
 Dizi,DIZIE:Array[1..1800000] of 0..1;
 DiziA,DiziB,DiziC,DiziAE,DiziBE,DiziCE,sure:Array[1..600000] of integer;
 DegerA,DegerAx,DegerB,DegerBx,DegerC,DegerCx,PSAY:Integer;
 M,Beta,GK:Real;
 Dosya:Textfile;

 implementation

 {$R *.DFM}

 Function Hex_cevir(Tamsayi:word):String;
 Const
 Hex_digits:String[16]=('0123456789ABCDEF');
 VAR
 Say:byte;
 Hex:string[4];
 begin
 Hex:='';
 For Say:=1 to 4 do
    Begin
    Hex:=Copy(Hex_digits,((Tamsayi mod 16)+1),1)+hex;
    Tamsayi:=Tamsayi div 16;
    end;
Hex_cevir:=hex;
End;

procedure TForm1.Button4Click(Sender: TObject);

LABEL Son1,Son2,ss1,ss2,ss3,ss4,ss5,ss6;

   VAR
   T1,J1,I1,J,I,top2:longint;
   T2,Deger1,code:INTEGER;
   F,FT,F1,n2,n1,NN,k,N,P:integer;
   Kat,Tx,s1,s2,KS1,KS2,Delta,bp,T,KS,C,X0,X1,Y1,Y2,SINUS:REAL;
   {v,b,XR,XL:ARRAY [1..360] OF REAL;}
   n1yazi,n2yazi,vyazi,nyazi,byazi,rmsyazi,fyazi,XLyazi,jyazi,xryazi:String;
   Krmsyazi,KVyazi,KByazi,Ltop1yazi,htop1yazi,kslyazi,highyazi,lowyazi:string;
   HlFark,MM,XRL,L,R,teta,Krms,Ltop1,htop1,Top:real;
   Top1,DT,DiziTsay:Integer;
   onceki:integer;
   yazi,Hexsayi,Hexsayi1,Hexsayi2,Hexsayi3:string;
   begin
{ Y=Sinus(X) ile Y=4/T(X-T/2-C)-1 fonksiyonlarının X eksenini
```

```
kestiği noktalar Newton Raphson yöntemi ile bulur. Bunun için
Y değerleri eşitlenirse Fonksiyon:
Y=Sinus(X-T/2-C)-4/T(X-T/2-C)+1 haline dönüşür. }


Val(Edit1.Text,M, Code); {M Modülasyon indeksi}
Val(Edit2.Text,T, Code); {T üçgen Dalganın peryodu}
Val(Edit6.Text,F, Code);
C:=0;                {Üçgen Dalganın ötelenmesi}
KS:=T*3/4;           {üçgen dalganın X eksenini ilk kestiği nokta}
PSAY:=TRUNC(180/T);  {180 derecedeki pals sayısı}
FOR J:=1 TO PSAY DO
  BEGIN
   X0:=KS+C;         {ilk hesaplanacak kökün baslangıç değri}
   KS2:=X0;
   FOR I:=1 TO 100 DO
     BEGIN
      SINUS:=M*SIN((3.141592/180)*X0);
      Y1:=SINUS-4/T*(X0-T/2-C)+1;              {Fonksiyon}
      Y2:=M*0.017453292*COS((3.141592/180)*X0)-4/T;{Fonksiyonun Türevi}
      X1:=X0-(Y1/Y2);
      IF ABS(X1-X0) < 0.0000001  THEN GOTO Son1;

      X0:=X1;

     END;
 Son1:XR[J]:=X1;      {Üçgen dalganın Sağa yatık(/)Kesim noktası bir diziye atılıyor}
 KSR[J]:=KS2;
 C:=C+T;          {Peryot Kadar ötelemeyi artır}
   END;

{ Üçgen dalganın sola yatık bölümü için Kesim değerleri hesaplanıyor}
 KS:=T/4;
 C:=0;
 PSAY:=TRUNC(180/T);
 FOR J:=1 TO PSAY DO BEGIN
  X0:=KS+C;
  KS1:=X0;
  FOR I:=1 TO 100 DO BEGIN
  SINUS:=M*SIN((3.141592/180)*X0);
  Y1:=SINUS+(4/T)*(X0-C)-1;
  Y2:=M*0.017453292*COS((3.141592/180)*X0)+4/T;
  X1:=X0-(Y1/Y2);
  IF ABS(X1-X0) < 0.00001  THEN GOTO Son2;

  X0:=X1;
  END;
Son2: XL[J]:=X1;
  KSL[J]:=KS1;
  C:=C+T;
  END;

{Hücrelerin içi temizleniyor}

  For i:=0 to 5 do begin
  For j:=1 to 1000 do Begin
  Stringgrid1.Cells[i,j]:=' ';
```

```
  Stringgrid2.Cells[i,j]:=' ';
  End;End;
{Sinüs Dalgasının low da kaldığı sürelerin toplamı}
  Ltop1:=0;
  Low[1]:=XL[1];
  LowX[1]:=XL[1];
  For j:=1 to psay do Begin
  Low[j+1]:=(XL[j+1]-XR[j]);
   if Low[j+1]<0.001 then Low[j+1]:=0.001;
  LowX[j+1]:=Low[j+1];
  Ltop1:=Ltop1+low[j];
  End;
  Low[psay+1]:=low[1];
  LowX[psay+1]:=Low[1];
  //Low[psay+1]:=180-XR[psay];
  //LowX[psay+1]:=180-XR[psay];
  Ltop1:=Ltop1+Low[psay+1];
  Str(Ltop1:3:2,yazi);
  Edit9.text:=yazi;  {Low değeri kutucuk içine yazdırılıyor}

{Sinüs Dalgasının High  da kaldığı sürelerin toplamı}
  htop1:=0;
  For j:=1 to psay do Begin
  High[j]:=(XR[j]-XL[j]);
  HighX[j]:=(XR[j]-XL[j]);
  htop1:=htop1+high[j];
  End;
  Str(htop1:3:2,htop1yazi);
  Edit8.text:=htop1yazi;


{Eşlenik high süreleri hesaplanıyor}

  For j:=1 to psay+1 do Begin
  HighE[j]:=Low[j];
  HighEX[j]:=Low[j];
  End;

{Eşlenik Low süreleri hesaplanıyor}

  For j:=1 to psay do Begin
  LowE[j]:=High[j];
  LowEX[j]:=High[j];
  End;


{----------Yeni değerler hesaplanıyor          ----------}
  F1:=F;
  //IF (F>50) and( F<=60) Then FT:=70;
  //IF (F>=40) and( F<=50) Then FT:=60;
  //IF (F<40) Then FT:=50;

  //IF (F>=44) and( F<=50) Then F:=50;
  //IF (F>=36) and( F<=42) Then F:=42;
  //IF (F>=30) and( F<=34) Then F:=34;
  //IF (F>=10) and( F<=28) Then F:=28;
  //IF (F>=2) and( F<=8) Then F:=26;
```

```
IF (F>50) Then F:=50;
Kat:=(0.5+f*0.0065);

For j:=1 to psay do Begin
High[j]:=HighX[j]*Kat;
Low[j]:=LowX[j]+(HighX[j]-(HighX[j]*kat))/2;
LowX[j+1]:=LowX[j+1]+(HighX[j]-(HighX[j]*Kat))/2;
End;
Low[Psay+1]:=Low[1];

Tx:=0;
For J :=1 to psay do Begin
Tx:=Tx+Low[j]+High[j];
End;
Tx:=tx+Low[psay+1];


//Eşlenik için


For j:=2 to psay do Begin
HighE[j]:=HighEx[j]*Kat;
LowE[j-1]:=LowE[j-1]+(HighEX[j]-(HighEX[j]*Kat))/2;
LowE[j]:=LowE[j]+(HighEX[j]-(HighEX[j]*Kat))/2;
End;

HighE[1]:=HighEx[1]*Kat;
HighE[psay+1]:=HighEx[psay+1]*Kat;
LowE[1]:=LowE[1]+(HighEX[1]-(HighEX[1]*Kat))/2;
LowE[psay]:=LowE[psay]+(HighEX[psay+1]-(HighEX[psay+1]*Kat))/2;

For j:=1 to psay do Begin
LowEX[j]:=LowE[j];
End;
For j:=1 to psay do Begin
LowE[j+1]:=LowEx[j];
End;
LowE[psay+2]:=(HighEX[psay+1]-(HighEX[psay+1]*Kat))/2;
LowE[1]:=(HighEX[1]-(HighEX[1]*Kat))/2;;

Tx:=0;
For J :=1 to psay+1 do Begin
Tx:=Tx+LowE[j]+HighE[j];
End;
Tx:=tx+LowE[psay+2];


F:=F1;

{Yeni XL değeri hesaplanıyor}

Top:=Low[1];
XL[1]:=Low[1];
For j:=2 to psay do Begin
Top:=Top+High[j-1]+low[j];
XL[j]:=Top;
```

```
XLL[j]:=Trunc(1000*Top);
End;
XLL[1]:=Trunc(1000*Low[1]);

{Yeni XR değeri hesaplanıyor}
Top:=0;
For J :=1 to psay do Begin
Top:=Top+Low[j]+High[j];
XR[j]:=Top;
XRR[j]:=Trunc(1000*Top);

End;

//eşlenik için

{Yeni XL değeri hesaplanıyor}

PSAY:=PSAY+1;
Top:=LowE[1];
XLE[1]:=LowE[1];
For j:=2 to psay do Begin
Top:=Top+HighE[j-1]+lowE[j];
XLE[j]:=Top;
XLLE[j]:=Trunc(1000*Top);
End;
XLLE[1]:=Trunc(1000*LowE[1]);

{Yeni XR değeri hesaplanıyor}
Top:=0;
For J :=1 to psay do Begin
Top:=Top+LowE[j]+HighE[j];
XRE[j]:=Top;
XRRE[j]:=Trunc(1000*Top);

End;

{------ low kesimler 0 ile dolduruluyor----------}

psay:=psay-1;
For j1:=1 to psay-1 do Begin
For I1:=XRR[j1] to XLL[J1+1] do Begin
Dizi[I1]:= 0;
End;
End;
For I1:=1 to XLL[1] do Begin
Dizi[I1]:= 0;
End;
For I1:=XRR[psay] to 180000 do Begin
Dizi[I1]:= 0;
End;

{----------High kesimler 1 ile dolduruluyor---------}
For J1:=1 to psay do Begin
For I1:=XLL[J1] to XRR[J1] do Begin
Dizi[I1]:= 1;
End;
END;
```

```
For J1:=1 to 60000 do Begin
DiziA[J1]:=Dizi[J1]
End;
For J1:=60001 to 120000 do Begin
DiziB[J1-60000]:=Dizi[J1]
End;
For J1:=120001 to 180000 do Begin
DiziC[J1-120000]:=Dizi[J1]
End;

//Eşlenik için
{------ low kesimler 0 ile dolduruluyor----------}

psay:=psay+1;
For j1:=1 to psay-1 do Begin
For I1:=XRRE[j1] to XLLE[J1+1] do Begin
DiziE[I1]:= 0;
End;
End;
For I1:=1 to XLLE[1] do Begin
DiziE[I1]:= 0;
End;
For I1:=XRRE[psay] to 180000 do Begin
DiziE[I1]:= 0;
End;

{-----------High kesimler 1 ile dolduruluyor---------}
For J1:=1 to psay do Begin
For I1:=XLLE[J1] to XRRE[J1] do Begin
DiziE[I1]:= 1;
End;
END;

For J1:=1 to 60000 do Begin
DiziAE[J1]:=DiziE[J1]
End;
For J1:=60001 to 120000 do Begin
DiziBE[J1-60000]:=DiziE[J1]
End;
For J1:=120001 to 180000 do Begin
DiziCE[J1-120000]:=DiziE[J1]
End;

psay:=psay-1;

I1:=0;
Onceki:=DiziA[1]+DiziB[1]+DiziC[1]+DiziAE[1]+DiziBE[1]+DiziCE[1];
For J1:=1 to 60000 do Begin
DT:=DiziA[J1]+DiziB[J1]+DiziC[J1]+DiziAE[j1]+DiziBE[j1]+DiziCE[j1];
IF DT<> onceki Then Begin I1:=I1+1;onceki:=DT;DiziT[I1]:=J1;End;
DiziTsay:=I1;   {kesim noktaları sayısı}
End;
DiziTsay:=I1+1;
DiziT[DiziTsay]:=60000; { Kesim noktalarını bulundurur}

Sure[1]:=DiziT[1];     {aralık olarak kesim noktalarının farkı}
```

```
  For J1:=2 to DiziTsay do begin
  Sure[J1]:=DiziT[J1]-DiziT[J1-1];
  End;

  top2:=0;
  For J1:=1 to DiziTsay do begin
  Top2:=Top2+sure[j1];
  End;


  ASSIGNFILE(DOSYA,'VERI.TXT');
  {$I-}
  APPEND(DOSYA);
  IF IORESULT<>0 THEN REWRITE(DOSYA);
  {$I+}
  WRITELN(DOSYA,' ');
  WRITELN(DOSYA,'TAB',F,':');
  For J1:=1 to DiziTsay do begin
  if j1=1 then Write(Dosya,'DB ');
  I1:=DiziT[j1];
  DegerA:=0;DegerB:=0;DegerC:=0;

  If DiziA[I1-1]=1 then DegerA:=32 else degerA:=0;
  If DiziAE[I1-1]=1 then DegerAx:=16 else degerAx:=0;
  If DiziB[I1-1]=1 then DegerB:=4 else degerB:=0;
  If DiziBE[I1-1]=1 then DegerBx:=8 else degerBx:=0;
  If DiziC[I1-1]=1 then DegerC:=2 else degerC:=0;
  If DiziCE[I1-1]=1 then DegerCx:=1 else degerCx:=0;

  T1:=Trunc(sure[J1]*1000*2/(3*360*F));
  If (T1>0)and (T1<=26) then T1:=1;
  If T1>26 then T1:=T1-26;
  If T1=0 then T1:=1;
  T1:=65535-T1;
  If T1=65535 then goto ss1;
  hexsayi:=Hex_cevir(T1);
  Hexsayi1:=copy(Hexsayi,1,2);
  Hexsayi2:=Copy(Hexsayi,3,2);
  deger1:=degerA OR degerB OR DegerC OR degerAX OR degerBX OR DegerCX;
  If (J1=DiziTsay) then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' ');goto ss1;
  end;
  If ((J1 mod 5)=0)then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' ');Write(Dosya,'DB ');
  end
  else
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1,',');
ss1:
  End;
  Writeln(Dosya,' ');

 //2
```

```
  For J1:=1 to DiziTsay do begin
  if j1=1 then Write(Dosya,'DB ');
  I1:=DiziT[J1];
  DegerA:=0;DegerB:=0;DegerC:=0;

  If DiziB[I1-1]=1 then DegerA:=32 else degerA:=0;
  If DiziBE[I1-1]=1 then DegerAx:=16 else degerAx:=0;
  If DiziC[I1-1]=1 then DegerB:=4 else degerB:=0;
  If DiziCE[I1-1]=1 then DegerBx:=8 else degerBx:=0;
  If DiziA[I1-1]=1 then DegerC:=1 else degerC:=0;
  If DiziAE[I1-1]=1 then DegerCx:=2 else degerCx:=0;


  T1:=Trunc(sure[J1]*1000*2/(3*360*F));
  If (T1>0)and (T1<=26) then T1:=1;
  If T1>26 then T1:=T1-26;
  If T1=0 then T1:=1;
  T1:=65535-T1;
  If T1=65535 then goto ss2;
  hexsayi:=Hex_cevir(T1);
  Hexsayi1:=copy(Hexsayi,1,2);
  Hexsayi2:=Copy(Hexsayi,3,2);
  deger1:=degerA OR degerB OR DegerC OR degerAX OR degerBX OR DegerCX;
  If J1=DiziTsay then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' '); goto ss2;
  end;
  If ((J1 mod 5)=0)then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' ');Write(Dosya,'DB ');
  end
  else
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1,',');
ss2:

  End;

  Writeln(Dosya,' ');

//3
  For J1:=1 to DiziTsay do begin
  if j1=1 then Write(Dosya,'DB ');
  I1:=DiziT[J1];
  DegerA:=0;DegerB:=0;DegerC:=0;
  If DiziC[I1-1]=1 then DegerA:=32 else degerA:=0;
  If DiziCE[I1-1]=1 then DegerAx:=16 else degerAx:=0;
  If DiziA[I1-1]=1 then DegerB:=8 else degerB:=0;
  If DiziAE[I1-1]=1 then DegerBx:=4 else degerBx:=0;
  If DiziB[I1-1]=1 then DegerC:=1 else degerC:=0;
  If DiziBE[I1-1]=1 then DegerCx:=2 else degerCx:=0;


  T1:=Trunc(sure[J1]*1000*2/(3*360*F));
  If (T1>0)and (T1<=26) then T1:=1;
  If T1>26 then T1:=T1-26;
```

```
  If T1=0 then T1:=1;
  T1:=65535-T1;
  If T1=65535 then goto ss3;

  hexsayi:=Hex_cevir(T1);
  Hexsayi1:=copy(Hexsayi,1,2);
  Hexsayi2:=Copy(Hexsayi,3,2);
  deger1:=0;
  deger1:=degerA OR degerB OR DegerC OR degerAX OR degerBX OR DegerCX;
  If (J1=DiziTsay)then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' ');goto ss3;
  end;
  If ((J1 mod 5)=0) then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' ');Write(Dosya,'DB ');
  end
  else
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1,',');
ss3:

  End;

  Writeln(Dosya,' ');

  //4

  For J1:=1 to DiziTsay do begin
  if j1=1 then Write(Dosya,'DB ');
  I1:=DiziT[J1];
  DegerA:=0;DegerB:=0;DegerC:=0;

  If DiziA[I1-1]=1 then DegerA:=16 else degerA:=0;
  If DiziAE[I1-1]=1 then DegerAx:=32 else degerAx:=0;
  If DiziB[I1-1]=1 then DegerB:=8 else degerB:=0;
  If DiziBE[I1-1]=1 then DegerBx:=4 else degerBx:=0;
  If DiziC[I1-1]=1 then DegerC:=1 else degerC:=0;
  If DiziCE[I1-1]=1 then DegerCx:=2 else degerCx:=0;


  T1:=Trunc(sure[J1]*1000*2/(3*360*F));
  If (T1>0)and (T1<=26) then T1:=1;
  If T1>26 then T1:=T1-26;
  If T1=0 then T1:=1;
  T1:=65535-T1;
  If T1=65535 then goto ss4;
  hexsayi:=Hex_cevir(T1);
  Hexsayi1:=copy(Hexsayi,1,2);
  Hexsayi2:=Copy(Hexsayi,3,2);
  deger1:=degerA OR degerB OR DegerC OR degerAX OR degerBX OR DegerCX;
  If J1=DiziTsay then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' ');goto ss4;
  end;
```

```
  If ((J1 mod 5)=0) then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' ');Write(Dosya,'DB ');
  end
  else
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1,',');
ss4:
  End;

  Writeln(Dosya,' ');

  //5

  For J1:=1 to DiziTsay do begin
  if j1=1 then Write(Dosya,'DB ');
  I1:=DiziT[J1];
  DegerA:=0;DegerB:=0;DegerC:=0;

  If DiziB[I1-1]=1 then DegerA:=16 else degerA:=0;
  If DiziBE[I1-1]=1 then DegerAx:=32 else degerAx:=0;
  If DiziC[I1-1]=1 then DegerB:=8 else degerB:=0;
  If DiziCE[I1-1]=1 then DegerBx:=4 else degerBx:=0;
  If DiziA[I1-1]=1 then DegerC:=2 else degerC:=0;
  If DiziAE[I1-1]=1 then DegerCx:=1 else degerCx:=0;



  T1:=Trunc(sure[J1]*1000*2/(3*360*F));
  If (T1>0)and (T1<=26) then T1:=1;
  If T1>26 then T1:=T1-26;
  If T1=0 then T1:=1;
  T1:=65535-T1;
  If T1=65535 then goto ss5;

  hexsayi:=Hex_cevir(T1);
  Hexsayi1:=copy(Hexsayi,1,2);
  Hexsayi2:=Copy(Hexsayi,3,2);
  deger1:=0;
  deger1:=degerA OR degerB OR DegerC OR degerAX OR degerBX OR DegerCX;
  If (J1=DiziTsay)then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' ');goto ss5;
  end;

  If ((J1 mod 5)=0)then
  Begin
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
  Writeln(Dosya,' ');Write(Dosya,'DB ');
  end
  else
  Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1,',');
ss5:
  End;
  Writeln(Dosya,' ');
```

```
//6

For J1:=1 to DiziTsay do begin
if j1=1 then Write(Dosya,'DB ');
I1:=DiziT[J1];
DegerA:=0;DegerB:=0;DegerC:=0;

If DiziC[I1-1]=1 then DegerA:=16 else degerA:=0;
If DiziCE[I1-1]=1 then DegerAx:=32 else degerAx:=0;
If DiziA[I1-1]=1 then DegerB:=4 else degerB:=0;
If DiziAE[I1-1]=1 then DegerBx:=8 else degerBx:=0;
If DiziB[I1-1]=1 then DegerC:=2 else degerC:=0;
If DiziBE[I1-1]=1 then DegerCx:=1 else degerCx:=0;


T1:=Trunc(sure[J1]*1000*2/(3*360*F));{12 mhz için 2 carpanı olmayacak}
If (T1>0)and (T1<=26) then T1:=1;
If T1>26 then T1:=T1-26;
If T1=0 then T1:=1;
T1:=65535-T1;
If T1=65535 then goto ss6;

hexsayi:=Hex_cevir(T1);
Hexsayi1:=copy(Hexsayi,1,2);
Hexsayi2:=Copy(Hexsayi,3,2);
deger1:=degerA OR degerB OR DegerC OR degerAX OR degerBX OR DegerCX;

If (J1=DiziTsay)then
Begin
Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
Write(Dosya,',','000H');
Writeln(Dosya,' '); goto ss6;
end;
If ((J1 mod 5)=0)then
Begin
Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1);
Writeln(Dosya,' ');Write(Dosya,'DB ');
end

else
Write(Dosya,'0',Hexsayi1,'H,','0',hexsayi2,'H',',',deger1,',');
ss6:

End;

Closefile(Dosya);

{----------------------------------------}

For i:=0 to 5 do begin
For j:=1 to 1000 do Begin
Stringgrid1.Cells[i,j]:=' ' ;
Stringgrid2.Cells[i,j]:=' ';
End;End;

{Sinüs Dalgasının High da kaldığı sürelerin toplamı}
TOP:=0;
```

```
FOR J:=1 TO PSAY DO BEGIN
TOP:=TOP+High[J];
END;
Str(top:3:2,yazi);
Edit8.text:=yazi;  {Low değeri kutucuk içine yazdırılıyor}

{Sinüs Dalgasının Low da kaldığı sürelerin toplamı}
Top:=0;
FOR J:=1 TO PSAY+1 DO BEGIN
TOP:=TOP+HighE[J];
END;
Str(top:3:2,yazi);
Edit9.text:=yazi;  {Low değeri kutucuk içine yazdırılıyor}


for j:=1 to psay do Begin
str(j:3,jyazi);
str(XL[j]:3:4,xlyazi);
str(XR[j]:3:4,xRyazi);
str(High[j]:3:4,highyazi);
//str(Low[j]:3:4,lowyazi);
str(HighE[j]:3:4,lowyazi);

stringgrid1.Cells[0,j]:=jyazi;
stringgrid1.Cells[1,j]:=xlyazi;
stringgrid1.Cells[2,j]:=xryazi;
stringgrid1.Cells[3,j]:=highyazi;
stringgrid1.Cells[4,j]:=lowyazi;
end;
str(HighE[psay+1]:3:4,lowyazi);
stringgrid1.Cells[4,psay+1]:=lowyazi;

{ Pwm dalganın high  olan kesimlerinin toplamı}
TOP:=0;
FOR J:=1 TO PSAY DO BEGIN
TOP:=TOP+(XR[J]-XL[J]);
END;

TOP:=0;
FOR J:=1 TO PSAY DO BEGIN
TOP:=TOP+High[J];
END;
FOR J:=1 TO PSAY+1 DO BEGIN
TOP:=TOP+HighE[J];
END;


{ pwm İÇİN rms DEGERİ  }

TOP:=sqrt(TOP/180);
STR(TOP:3:4,Rmsyazi);
edit3.text:=rmsyazi;
//-----------------------------------

FOR J:=1 TO PSAY DO BEGIN
XBL[2*j]:=XL[J];
XBR[2*j]:=XR[J];
```

```
XBL[2*j-1]:=XLE[J];
XBR[2*j-1]:=XRE[J];
END;
XBL[2*(psay+1)-1]:=XLE[psay+1];
XBR[2*(psay+1)-1]:=XRE[psay+1];


{PWM İÇİN FOURİER KATSAYILARI}

V[1]:=0;
BP:=0;
n:=1;
Val(edit7.text,nn,code);
While(n<=nn) Do begin
FOR p:=1 TO Psay+1 DO BEGIN
Delta:=Pi/180*(XBR[p]-XBL[p]);
{ Delta:=Pi/180*High[p];}
S1:=Cos(n*((pi/180)*XBL[p]));
S2:=Cos(n*((pi/180)*XBL[p]+Delta));
BP:=BP+power((-1),p)*2*2/(n*Pi)*(s1-s2);
END;
B[n]:=BP;  {B[n] Fourier katsayıları değerleri}
BP:=0;
N:=N+2;
end;

N2:=1;
N1:=1; {Fourier Katsayıları numarası}
While n1<=nn  do begin
Str(n1:2,n2yazi);
Str(b[n1]:3:4,BYazi);
stringgrid2.Cells[0,N2]:=n2YAZI;{ Pwm için Fourier katsayı numarası yazdırılıyor}
stringgrid2.Cells[1,N2]:=bYAZI; { Fourier katsayıları}
N1:=N1+2;
N2:=n2+1;
end;

{ PWM için ters fourier dönüşümü hesaplanıyor.}
FOR K:=0 TO 360 DO BEGIN
top:=0;
n:=1;
while (n<=nn) Do begin
top:= top+B[N]*Sin(N*(Pi/180*K));
n:=n+2;
End;
v[K]:=top; {Anlık PWM değerleri toplanarak V[k] ya atılıyor]}
END;

{PWM için Açı değerleri ve o açı değerinde Fonksiyonun aldığı değer}
FOR N1:=1 TO 360 DO BEGIN
Str(n1:3,n1yazi);
Str(v[n1]:3:4,vYazi);
stringgrid2.Cells[3,N1]:=n1YAZI;
stringgrid2.Cells[4,N1]:=vyazi;
end;
```

```
{ PWM İÇİN AKIM  hesaplanıyor.}
// L:=0.5;
// R:=0.1;
 Val(edit5.text,L,code);
 Val(edit4.text,R,code);

 FOR K:=0 TO 360 DO BEGIN
 top:=0;
 n:=1;
 while (n<=nn) Do begin
 TETA:=ARCTAN(N*L/R);
 top:= top+B[N]*(1/SQRT(SQR(R*R+(N*L)))*Sin(N*(Pi/180*K)));
 n:=n+2;
 End;
 AKIM[K]:=top; {Anlık PWM değerleri toplanarak V[k] ya atılıyor]}
 END;

 {Kare dalga için Fourier katsayıları}

 Beta:=Pi/2;{Kare dalganın genişliği Radyan olarak}
 N:=1;
 Val(edit7.text,nn,code);
 While n<=nn Do begin
 KB[N]:=2/(n*Pi)*(Cos(n*(pi-Beta)/2)-Cos(n*(pi+Beta)/2));
 N:=N+2;
 end;

 {Kare dalga için Katsayılar ekrana yazdırılıyor}
 n:=1;
 n1:=1;
 While n<=nn Do begin
 Str(KB[n]:3:4,KBYazi);
 stringgrid2.Cells[2,N1]:=KBYAZI;
 n1:=n1+1;
 n:=n+2;
 End;

 {Kare dalga için ters dönüşüm yapılıyor}
 FOR K:=0 TO 360 DO BEGIN
 top:=0;
 n:=1;
 while (n<=nn) Do begin
 top:= top+KB[N]*Sin(N*(Pi/180*K));
 n:=n+2;
 End;
 KV[K]:=top;
 Str(KV[K]:3:4,KVYazi);
 stringgrid2.Cells[5,K+1]:=KVYAZI;
END;

{Kare dalganın rms değeri }

 KRMS:=Sqrt(Beta/Pi);
 Str(Krms:3:4,Krmsyazi);
 Edit10.text:=Krmsyazi;

{Kare dalga için AKIM dönüşüm yapılıyor}
```

```
    FOR K:=0 TO 360 DO BEGIN
    top:=0;
    n:=1;
    while (n<=nn) Do begin
    TETA:=ARCTAN(N*L/R);
    top:= top+KB[N]*(1/SQRT(SQR(R*R+(N*L)))*Sin(N*(Pi/180*K)));
    n:=n+2;
    End;
    AKIM2[K]:=top;
   END;

end;
procedure TForm1.Button5Click(Sender: TObject);
begin
HALT(1);
end;

procedure TForm1.Button6Click(Sender: TObject);
begin
//form2.showmodal;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
stringgrid1.Cells[0,0]:='NP';
stringgrid1.Cells[1,0]:='X LEFT';
stringgrid1.Cells[2,0]:='X RIGHT';
stringgrid1.Cells[3,0]:='HIGH';
stringgrid1.Cells[4,0]:='LOW';

stringgrid2.Cells[0,0]:='CN';
stringgrid2.Cells[1,0]:='B-PWM';
stringgrid2.Cells[2,0]:='B-SQ.';
stringgrid2.Cells[3,0]:='ANGLE';
stringgrid2.Cells[4,0]:='V-PWM';
stringgrid2.Cells[5,0]:='V-SQ.';

end;

procedure TForm1.Button1Click(Sender: TObject);
var
i:integer;
sut,sat:integer;
begin
shape1.repaint;
sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);

{Pwm dalga ekrana çizdiriliyor}
Gk:=1.5;{ Görüntü zoom Katsayısı}
for i:=1 to (2*Psay)+1 do begin
if (i mod 2)=1 then begin
canvas.lineto(sut+TRUNC(GK*XBL[I]),sat);
canvas.lineto(sut+TRUNC(GK*XBL[I]),Trunc(1*(sat+50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),Trunc(1*(sat+50)));
```

```
canvas.lineto(sut+TRUNC(GK*XBR[I]),sat);
end;
if (i mod 2)=0 then begin
canvas.lineto(sut+TRUNC(GK*XBL[I]),sat);
canvas.lineto(sut+TRUNC(GK*XBL[I]),Trunc(1*(sat-50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),Trunc(1*(sat-50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),sat);
end;
end;

Sut:=Sut+Trunc(GK*180);
canvas.moveto(sut,sat);
for i:=1 to (2*Psay)+1 do begin
if (i mod 2)=1 then begin
canvas.lineto(sut+TRUNC(GK*XBL[I]),sat);
canvas.lineto(sut+TRUNC(GK*XBL[I]),Trunc(1*(sat-50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),Trunc(1*(sat-50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),sat);
end;
if (i mod 2)=0 then begin
canvas.lineto(sut+TRUNC(GK*XBL[I]),sat);
canvas.lineto(sut+TRUNC(GK*XBL[I]),Trunc(1*(sat+50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),Trunc(1*(sat+50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),sat);
end;
End;
End;

procedure TForm1.Button2Click(Sender: TObject);
var
sut,sat,i:integer;
begin
shape1.repaint;
sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);
{Pwm ters dönüşüm ekrana çizdiriliyor}
for i:=1 to 360 do begin
canvas.lineto(sut+TRUNC(GK*i),trunc(Sat-50*v[i]));
end;
end;

procedure TForm1.Button3Click(Sender: TObject);
var
i:integer;
sut,sat:integer;
{Pwm için her iki deger de çizdiriliyor}
begin
shape1.repaint;
sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);

{Pwm dalga ekrana çizdiriliyor}
Gk:=1.5;{ Görüntü zoom Katsayısı}
//canvas.lineto(sut,Trunc(1*(sat+50)));
for i:=1 to (2*Psay)+1 do begin
```

```
if (i mod 2)=1 then begin
canvas.lineto(sut+TRUNC(GK*XBL[I]),sat);
canvas.lineto(sut+TRUNC(GK*XBL[I]),Trunc(1*(sat+50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),Trunc(1*(sat+50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),sat);
end;
if (i mod 2)=0 then begin
canvas.lineto(sut+TRUNC(GK*XBL[I]),sat);
canvas.lineto(sut+TRUNC(GK*XBL[I]),Trunc(1*(sat-50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),Trunc(1*(sat-50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),sat);
end;
end;

Sut:=Sut+Trunc(GK*180);
canvas.moveto(sut,sat);
for i:=1 to (2*Psay)+1 do begin
if (i mod 2)=1 then begin
canvas.lineto(sut+TRUNC(GK*XBL[I]),sat);
canvas.lineto(sut+TRUNC(GK*XBL[I]),Trunc(1*(sat-50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),Trunc(1*(sat-50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),sat);
end;
if (i mod 2)=0 then begin
canvas.lineto(sut+TRUNC(GK*XBL[I]),sat);
canvas.lineto(sut+TRUNC(GK*XBL[I]),Trunc(1*(sat+50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),Trunc(1*(sat+50)));
canvas.lineto(sut+TRUNC(GK*XBR[I]),sat);
end;
end;

sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);
for i:=1 to 360 do begin
canvas.lineto(sut+TRUNC(GK*i),trunc(Sat-50*v[i]));
end;
End;

procedure TForm1.Button7Click(Sender: TObject);
var
i:integer;
sut,sat:integer;
{Kare dalga çizdiriliyor}
begin
shape1.repaint;
sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);
canvas.lineto(sut+TRUNC(((Pi-Beta)/2)*180/pi*GK),sat);
canvas.lineto(sut+TRUNC(((Pi-Beta)/2)*180/pi*GK),sat-100);
canvas.lineto(sut+TRUNC((((Pi-Beta)/2)+Beta)*180/pi*GK),sat-100);
canvas.lineto(sut+TRUNC((((Pi-Beta)/2)+Beta)*180/pi*GK),sat);
canvas.lineto(sut+tRUNC(180*GK),sat);

Sut:=Sut+TRUNC(GK*180);
canvas.moveto(sut,sat);
```

```
 canvas.lineto(sut+TRUNC(((Pi-Beta)/2)*180/pi*GK),sat);
 canvas.lineto(sut+TRUNC(((Pi-Beta)/2)*180/pi*GK),sat+100);
 canvas.lineto(sut+TRUNC((((Pi-Beta)/2)+Beta)*180*GK/pi),sat+100);
 canvas.lineto(sut+TRUNC((((Pi-Beta)/2)+Beta)*180/pi*GK),sat);
 canvas.lineto(sut+TRUNC(180*GK),sat);
end;

procedure TForm1.Button8Click(Sender: TObject);
var
sut,sat,i:integer;
{Kare dalganın ters dönüşümü çizdiriliyor}
begin
shape1.repaint;
sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);
for i:=1 to 360 do begin
canvas.lineto(sut+TRUNC(GK*i),trunc(Sat-100*KV[i]));
end;
end;

procedure TForm1.BOTHClick(Sender: TObject);
var
i:integer;
sut,sat:integer;
{Kare dalganın her ikiside çizdiriliyor}
begin
shape1.repaint;
sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);
canvas.lineto(sut+TRUNC(((Pi-Beta)/2)*180/pi*GK),sat);
canvas.lineto(sut+TRUNC(((Pi-Beta)/2)*180/pi*GK),sat-100);
canvas.lineto(sut+TRUNC((((Pi-Beta)/2)+Beta)*180/pi*GK),sat-100);
canvas.lineto(sut+TRUNC((((Pi-Beta)/2)+Beta)*180/pi*GK),sat);
canvas.lineto(sut+TRUNC(180*GK),sat);

Sut:=Sut+TRUNC(GK*180);
canvas.moveto(sut,sat);
canvas.lineto(sut+TRUNC(((Pi-Beta)/2)*180/pi*GK),sat);
canvas.lineto(sut+TRUNC(((Pi-Beta)/2)*180/pi*GK),sat+100);
canvas.lineto(sut+TRUNC((((Pi-Beta)/2)+Beta)*180/pi*GK),sat+100);
canvas.lineto(sut+TRUNC((((Pi-Beta)/2)+Beta)*180/pi*GK),sat);
canvas.lineto(sut+TRUNC(GK*180),sat);

sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);
for i:=1 to 360 do begin
canvas.lineto(sut+TRUNC(GK*i),trunc(Sat-100*KV[i]));
end;
end;

procedure TForm1.CURRENTClick(Sender: TObject);
var
sut,sat,i:integer;
{AKIM dalgansı çizdiriliyor}
```

```
begin
shape1.repaint;
sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);
for i:=1 to 360 do begin
canvas.lineto(sut+TRUNC(GK*i),trunc(Sat-40*AKIM[i]));
end;
end;

procedure TForm1.CURRENT2Click(Sender: TObject);
VAR
sut,sat,i:integer;
{AKIM dalgansı çizdiriliyor}
begin
shape1.repaint;
sut:=shape1.left+10;
sat:=trunc(shape1.height/2)+shape1.top;
canvas.moveto(sut,sat);
for i:=1 to 360 do begin
canvas.lineto(sut+TRUNC(GK*i),trunc(Sat-40*AKIM2[i]));
END;
end;

end.
```

EK-2

MİKRODENETLEYİCİ KONTROL YAZILIMLARI

```
;BİRİNCİ MİKRODENETLEYİCİ KONTROL YAZILIMI
;TUŞ TAKIMINDA FREKANS VE KUTUP SAYISI BİLGİLERİNİ OKUR
;DEVİR SAYISINI HESAPLAR LCD EKRANDA GÖSTERİR.
;FREKAND DEĞERİNİ KESME ÜRETEREK İKİNCİ MİKRODENETLEYİCİYE GÖNDERİR

FREKANS  EQU 40H    ;Geçici frekans değeri
FREKANS1 EQU 41H    ;Gerçek frekans değeri
KUTUP   EQU 42H     ;Kutup sayısı
DIJITF1 EQU 43H     ;LCD ekran için frekansın ilk iki hanesi
DIJITF2 EQU 44H     ;LCD ekran için frekansın ikinci iki hanesi
DIJITP1 EQU 45H     ;LCD ekran kutup sayısı için
DIJITP2 EQU 46H     ;LCD ekran Kutup sayısı için
DIJITN1 EQU 47H     ;LCD ekran DEVİR sayısı gösterimi için
DIJITN2 EQU 48H     ;LCD ekran DEVİR sayısı gösterimi için
DIJITN3 EQU 49H     ;LCD ekran DEVİR sayısı gösterimi için
DIJITN4 EQU 50H     ;LCD ekran DEVİR sayısı gösterimi için
DIJITN5 EQU 51H     ;LCD ekran DEVİR sayısı gösterimi için

DIJIT4 EQU 52H
DIJIT3 EQU 53H
DIJIT2 EQU 54H      ;LCD ekran geçici değerleri için
DIJIT1 EQU 55H

BOL4 EQU 56H
BOL3 EQU 57H
BOL2 EQU 58H ;hesaplamalarda bölme sonucu için
BOL1 EQU 59H

KEY    EQU 60H      ;Tuş kodunu saklamak için
SONUC1  EQU 61H
SONUC2  EQU 62H


LCD_DATA EQU P0     ;LCD ekran veri gönderilen port
LCD_RS EQU P1.3     ;LCD ekran RS ucu
LCD_RW EQU P1.2     ;LCD ekran Read/Write ucu
LCD_E EQU P1.1      ;LCD ekran Enable ucu
KEYB_DATA EQU P1    ;Tuştakımı veri giriş portu

ORG 0000H           ;Program başlangıç adresi
LJMP INIT

ORG 0003H
LJMP KEYIN          ;Tuştakımı giriş kesme altprogramı


ORG 0030H           ;Program başlangıç adresi
INIT:
LCALL RESETLEME     ;LCD resetleme alt programını çağır
SETB IT0            ;Zamanlayıcı 0 kesmesini aktifle
SETB IT1
MOV FREKANS,#00H    ;Frekans değerini sıfırla
MOV KUTUP,#01H      ;Kutup değerini sıfırla
MOV FREKANS1,#00H   ;İkini işlemciye gönderilen gerçek frekans değeri
MOV KEY,#00H        ;Tuş bilgisini sıfırla
MOV DIJITF1,#00H
```

```
MOV DIJITF2,#00H
MOV DIJITP1,#01H
MOV DIJITP2,#00H
MOV DIJITN1,#00H
MOV DIJITN2,#00H
MOV DIJITN3,#00H
MOV DIJITN4,#00H
MOV DIJITN5,#00H

MOV IE,#85H             ;Kemeleri yetkile

LCALL DISPLAY           ;İlk başlangıç değerlerini ekranda göster

MAIN:
NOP
NOP
NOP
NOP                     ;Sonsuz döngüde çalış
NOP
NOP
NOP
NOP
LJMP MAIN

;----TUŞTAKIMINDAN DEĞER GİRME KESME ALT PROGRAMI----

KEYIN:
MOV IE,#00H             ;Diğer kesmeleri disable et
MOV KEY,#00H

MOV A,P1                ;Tuş takımını oku
ANL A,#0F0H
SWAP A                  ;Tuş kodunu ilk dört bit'e al

;Tuş takımındaki
;0 Frekansı artırmak için
;4 Frekansı azaltmak için
;1 Kutup sayısın artırmak için
;5 Kutup sayısını azaltmak için
;2 Enter
;6 Stop amacı ile kullanılmıştır.

MOV KEY,A               ;Tuş kodunu KEY değişkenine al
CJNE A,#0,AZALTF        ;Tuş 0 değilse AZALTF'e git
INC FREKANS             ;0 ise frekansı 1 artır
MOV A,FREKANS
CJNE A,#16,AZALTF       ;Frekans 60 den büyük mü?
MOV FREKANS,#1          ;Evet Frekansı 1 le


AZALTF:
MOV A,KEY
CJNE A,#4,ARTIRP        ;Tuş 4 değilse ARTIRP'e git
DEC FREKANS             ;4 ise frekansı 1 azalt
MOV A,FREKANS
CJNE A,#0,ARTIRP        ;Frekans 0'ın altında mı
MOV FREKANS,#15         ;Evet Frekansı 50 yap
```

```
ARTIRP:
MOV A,KEY
CJNE A,#1,AZALTP      ;Tuş 1 değilse Azaltp'ye git
INC KUTUP             ;1 ise Kutup sayısını 1 artır
MOV A,KUTUP
CJNE A,#6,AZALTP
MOV KUTUP,#1

AZALTP:
MOV A,KEY
CJNE A,#5,STOP        ;Tuş 5 değilse STOP'e git
DEC KUTUP             ;0 ise frekansı 1 artır
MOV A,KUTUP
CJNE A,#0,STOP
MOV KUTUP,#5


STOP:
MOV A,KEY
CJNE A,#6,S1          ;6 tuşuna basıldı.Motoru durdur
MOV FREKANS,#0
MOV FREKANS1,#0
MOV P2,FREKANS1       ;Frekans bilgisini gönder.
SETB P1.0             ;Kesme sinyali üret
LCALL DELAY
LCALL DELAY
CLR P1.0
LCALL DELAY
LCALL DELAY
SETB P1.0             ;Kesme sinyali üretme işlemi tamam

S1:

;------gerçek frekans değerlerini bul-----

MOV A,FREKANS         ;Frekans değerini a'ya al
CJNE A,#1,G1
MOV FREKANS1,#5       ;Gerçek frekans değeri 5
G1:
CJNE A,#2,G2
MOV FREKANS1,#10
G2:
CJNE A,#3,G3
MOV FREKANS1,#15
G3:
CJNE A,#4,G4
MOV FREKANS1,#20      ;Gerçek frekans değeri 20
G4:
CJNE A,#5,G5
MOV FREKANS1,#25
G5:
CJNE A,#6,G6
MOV FREKANS1,#30
G6:
CJNE A,#7,G7
MOV FREKANS1,#35
```

```
G7:
CJNE A,#8,G8
MOV FREKANS1,#40
G8:
CJNE A,#9,G9
MOV FREKANS1,#42
G9:
CJNE A,#10,G10
MOV FREKANS1,#44    ;Gerçek frekans değeri 44
G10:
CJNE A,#11,G11
MOV FREKANS1,#46
G11:
CJNE A,#12,G12
MOV FREKANS1,#48
G12:
CJNE A,#13,G13
MOV FREKANS1,#50
G13:
CJNE A,#14,G14
MOV FREKANS1,#55
G14:
CJNE A,#15,G15
MOV FREKANS1,#60    ;Gerçek frekans değeri 60
G15:

CLR A
MOV A,KEY
CJNE A,#2,S2        ;Enter tuşuna basımadı ise S2'e git
MOV A,FREKANS1      ;Enter tuşuna basıldı
MOV P2,A            ;Gerçek frekans bilgisini gönder
SETB P1.0           ;Kesme sinyali üret
LCALL DELAY
LCALL DELAY
CLR P1.0
LCALL DELAY
LCALL DELAY
SETB P1.0           ;Kesme sinyali tamam
S2:

;--GİRİLEN DEĞERLERE GÖRE DEVİR SAYISI HESAPLANIYOR--

MOV A,FREKANS1
MOV B,#10
DIV AB
MOV DIJITF1,B
MOV DIJITF2,A

MOV A,KUTUP
MOV B,#10
DIV AB
MOV DIJITP1,B
MOV DIJITP2,A

MOV A,#60
MOV B,KUTUP
DIV AB
```

```
MOV B,FREKANS1
MUL AB
MOV SONUC1,A
MOV SONUC2,B        ;60*F/P Ssonucunun MSB Değeri

MOV A,SONUC1
MOV B,#10H
DIV AB
MOV DIJIT1,B
MOV DIJIT2,A

MOV A,SONUC2
MOV B,#10H
DIV AB
MOV DIJIT3,B
MOV DIJIT4,A


MOV A,DIJIT4
MOV B,#10
DIV AB
MOV BOL4,A
MOV A,B

SWAP A
ORL A,DIJIT3
MOV B,#10
DIV AB
MOV BOL3,A
MOV A,B

SWAP A
ORL A,DIJIT2
MOV B,#10
DIV AB
MOV BOL2,A
MOV A,B

SWAP A
ORL A,DIJIT1
MOV B,#10
DIV AB
MOV BOL1,A
MOV DIJITN1,B       ; Devir sayısı 1.Dijit

MOV R0,#3
BAS:
MOV DIJIT4,BOL4
MOV DIJIT3,BOL3
MOV DIJIT2,BOL2
MOV DIJIT1,BOL1

MOV A,DIJIT4
MOV B,#10
DIV AB
MOV BOL4,A
```

```
MOV A,B

SWAP A
ORL A,DIJIT3
MOV B,#10
DIV AB
MOV BOL3,A
MOV A,B

SWAP A
ORL A,DIJIT2
MOV B,#10
DIV AB
MOV BOL2,A
MOV A,B

SWAP A
ORL A,DIJIT1
MOV B,#10
DIV AB
MOV BOL1,A

CJNE R0,#3,K1
MOV DIJITN2,B          ; Devir sayısı 2.Dijit
LJMP SON

K1:
CJNE R0,#2,K2
MOV DIJITN3,B          ; Devir sayısı 3.Dijit
LJMP SON
K2:
CJNE R0,#1,SON
MOV DIJITN4,B          ; Devir sayısı 4.Dijit
MOV DIJITN5,A          ; Devir sayısı 5.Dijit
SON:
DJNZ R0,BAS

LCALL DISPLAY          ;Değerleri ekranda göster
MOV IE,#85H
RETI

;--DİSPLAY ALT PROGRAMI--

DISPLAY:
lcall resetleme
;LCALL LINE1
mov a,#46H             ;F Harfi
lcall data_gonder
mov a,#3AH             ;: Karakteri
lcall data_gonder
MOV A,DIJITF2
ADD A,#48
LCALL DATA_GONDER

MOV A,DIJITF1
ADD A,#48
LCALL DATA_GONDER
```

```
LCALL LINE2
MOV A,#50H              ;P Karakteri
LCALL DATA_GONDER

MOV A,#3AH              ;: Karakteri
LCALL DATA_GONDER

MOV A,DIJITP2
ADD A,#48
LCALL DATA_GONDER
MOV A,DIJITP1
ADD A,#48
LCALL DATA_GONDER

LCALL LINE3
MOV A,#4EH              ;N Karaktari
LCALL DATA_GONDER

MOV A,#3AH              ;: Karakteri
LCALL DATA_GONDER

MOV A,DIJITN4
ADD A,#48
LCALL DATA_GONDER

MOV A,DIJITN3
ADD A,#48
LCALL DATA_GONDER

MOV A,DIJITN2
ADD A,#48
LCALL DATA_GONDER

MOV A,DIJITN1
ADD A,#48
LCALL DATA_GONDER

MOV A,KEY               ;Enter tuşuna basıldığında OK yazdır
CJNE A,#2,L1
LCALL LINE4
MOV A,#4FH              ;O harfi
LCALL DATA_GONDER
MOV A,#4BH              ;K karakteri
LCALL DATA_GONDER
L1:
RET

;--LCD İMLEÇ KONUMLANDIRMA--

LINE1:
MOV LCD_DATA,#128
CLR LCD_RS
LCALL CLOCK
RET

LINE2:
```

```
MOV LCD_DATA,#136
CLR LCD_RS
LCALL CLOCK
RET

LINE3:
MOV LCD_DATA,#192
CLR LCD_RS
LCALL CLOCK
RET

LINE4:
MOV LCD_DATA,#200
CLR LCD_RS
LCALL CLOCK
RET

;--LCD DATA GÖNDER ALT PROGRAMI--

DATA_GONDER:
MOV LCD_DATA,A
SETB LCD_RS
LCALL CLOCK
RET

;--CLOCK SİNYALİ ÜRETME ALT PROGRAMI--

CLOCK:
CLR LCD_RW
CLR LCD_E
LCALL DELAY
SETB LCD_E
LCALL DELAY
CLR LCD_E
LCALL DELAY
RET

;--LCD RESETLEME ALT PROGRAMI

RESETLEME:
MOV LCD_DATA,#3CH
CLR LCD_RS
LCALL CLOCK
MOV LCD_DATA,#0CH
CLR LCD_RS
LCALL CLOCK
MOV LCD_DATA,#01H
CLR LCD_RS
LCALL CLOCK
MOV LCD_DATA,#06H
CLR LCD_RS
LCALL CLOCK
MOV LCD_DATA,#02H
CLR LCD_RS
LCALL CLOCK
RET
```

```
;--GECİKME ALT PROGRAMI--
DELAY:
MOV R1,#50
LOOP1:
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
DJNZ R1,LOOP1
RET

END
```

```
;İKİNCİ MİKRODENETLEYİCİ KONTROL YAZILIMI
;BİRİNCİ MİKRODENETLEYİCİDEN KESME GELDİĞİNDE
;FREKANS DEĞERİNİ ALIR VE BUNA GÖRE TABLO DEĞERİNİ
;YÜKLEYEREK MOTORU ÇALIŞTIRIR.
DPTRE1  EQU 40H
DPTRE2  EQU 41H
CEKA    EQU 42H

ORG 0000H           ;Mikroişlemci başlangıç adresi
LJMP INIT           ;INIT altprogramına dallan
ORG 0013H
LJMP AL             ;Frekans bilgisini ilk işlemciden alma
                    ;Kesme alt programı

ORG 0030H
INIT:
MOV P0,#00H         ;Motor çıkışlarını kapat
MOV IE,#85H         ;Kesmeleri aktifle
MOV TMOD,#11H       ;Zamanlayıcıyı ayarla
SETB IT0            ;Zamanlayızı 0 kenar tetiklemeli ayarla
MOV DPTRE2,#0       ;Tablo başlangıç adresi 0 kabul et
MOV DPTRE1,#0
MOV DPTR,#TAB0      ;Varsayılan baslangıç tablosu TAB0
                    ;Motor durma pozisyonunda

INIT1:
MOV P0,#0           ;Motor duruyor
MOV A,DPTRE2        ;AL kesme alt programından yeni bir frekans
CJNE A,#0,MAIN      ;bilgisi gelmişse A kaydedicisini değeri 0 dan
SJMP INIT1          ;farklı olacaktır. 0 dan farklı ise MAIN'e git
                    ;değilse INIT1'e git

MAIN:
MOV DPH,DPTRE2      ;Yeni frekans tablosununun başlangıç
MOV DPL,DPTRE1      ;adresini DPTR kaydedicisine yükle

BAS:
CLR A               ;A=0
MOVC A,@A+DPTR      ;A=Tablonun ilk değerini
MOV R0,A
CJNE R0,#00H,K1
SJMP MAIN

K1:
MOV TH0,A           ;TH0=A
CLR A
INC DPTR            ;Adresi bir artır
MOVC A,@A+DPTR      ;A=Tablonun ikinci değeri
MOV TL0,A           ;TL0=A
CLR A
INC DPTR            ;Adresi bir artır
MOVC A,@A+DPTR      ;Tablonun üçüncü değeri
MOV P0,A            ;Üçüncü değeri P0 portuna gönder.
                    ;Tablonun ilk iki değeri zaman bilgisi, üçüncü
                    ;değer motora gönderilen değeri ifade eder
```

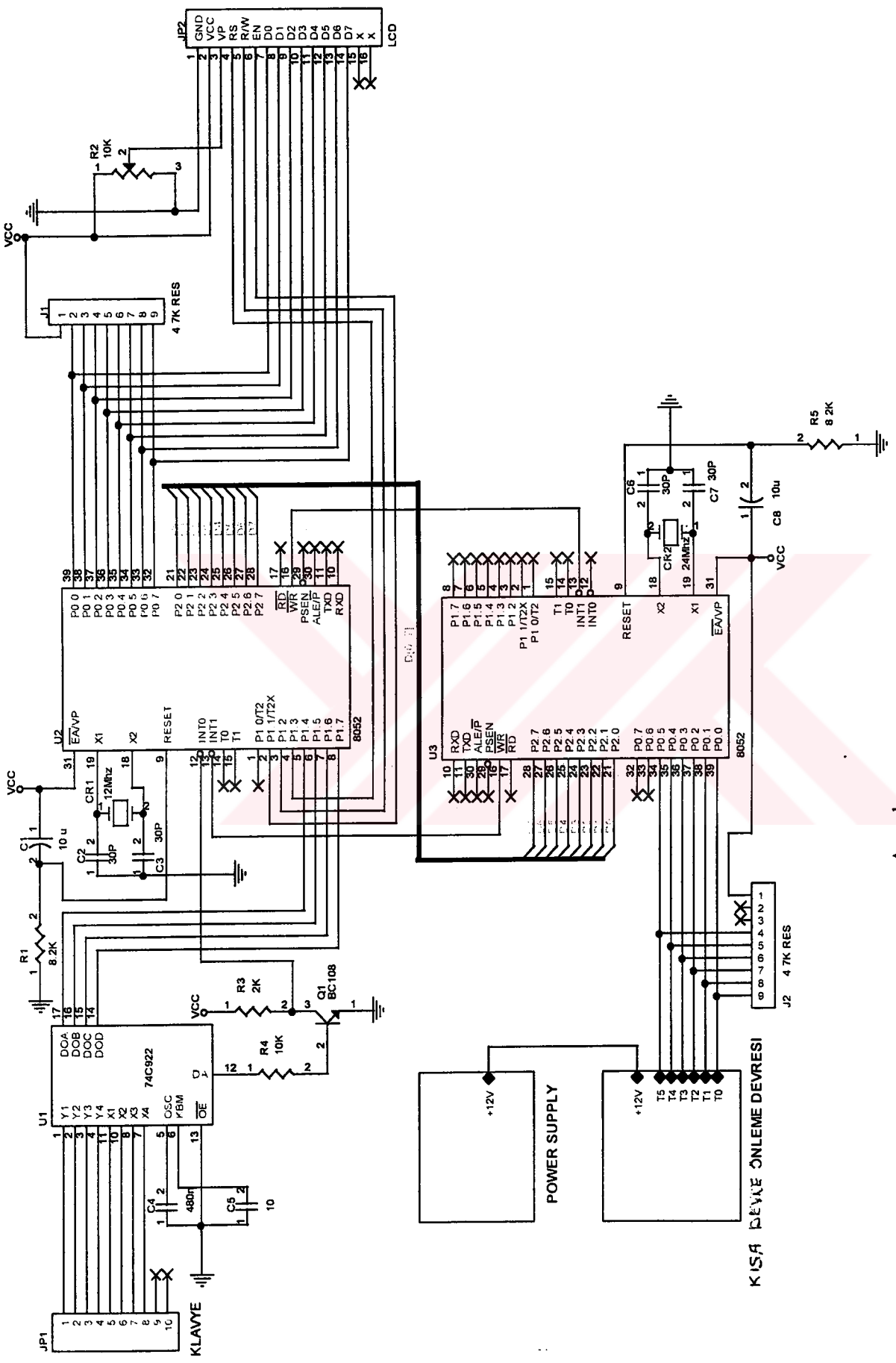
```
INC DPTR              ;Adresi bir artır
CLR TF0
SETB TR0              ;zamanlayıcıyı çalıştır
LOOP: JNB TF0,LOOP    ; TF0 set oluncaya kadar dön
CLR TR0               ;Zamanlayıcıyı durdur
LJMP BAS

;--------KESME ALT PROGRAMI-------------

AL:                   ;İnterrupt geldiğinde
MOV IE,#00H           ;Kesmeleri Disable et
PUSH DPH              ;Eski tablo baslangıç adresini sakla
PUSH DPL
MOV CEKA,A            ;A değerini sakla
PUSH CEKA
MOV P0,#00H           ;Motor çıkışlarını kapat
MOV A,P2              ;Frekans değerini porttan oku

CJNE A,#0,G0          ;Frekans 0'dan farklı ise G0'a git
MOV DPTR,#TAB0        ;Frekans 0,TAB0'ın Başlangıç adresini al
LJMP SON1             ;SON1'e git
G0:
CJNE A,#5,G1          ;Frekans 5'dan farklı ise G1'e git
MOV DPTR,#TAB5        ;Frekans 5,TAB5'ın Başlangıç adresini al
LJMP SON1
G1:
CJNE A,#10,G2         ;Frekans 10'dan farklı ise G2'ye git
MOV DPTR,#TAB10       ;Frekans 10,TAB10'un Başlangıç adresini al
LJMP SON1
G2:
CJNE A,#15,G3
MOV DPTR,#TAB15
LJMP SON1
G3:
CJNE A,#20,G4
MOV DPTR,#TAB20
LJMP SON1
G4:
CJNE A,#25,G5
MOV DPTR,#TAB25
LJMP SON1
G5:
CJNE A,#30,G6
MOV DPTR,#TAB30
LJMP SON1
G6:
CJNE A,#35,G7
MOV DPTR,#TAB35
LJMP SON1
G7:
CJNE A,#40,G8
MOV DPTR,#TAB40
LJMP SON1
G8:
CJNE A,#42,G9
MOV DPTR,#TAB42
LJMP SON1
```

```
G9:
CJNE A,#44,G10
MOV DPTR,#TAB44
LJMP SON1
G10:
CJNE A,#46,G11
MOV DPTR,#TAB46
LJMP SON1
G11:
CJNE A,#48,G12
MOV DPTR,#TAB48
LJMP SON1
G12:
CJNE A,#50,G13
MOV DPTR,#TAB50
LJMP SON1
G13:

CJNE A,#55,G14
MOV DPTR,#TAB55
LJMP SON1
G14:

CJNE A,#60,G15          ;Frekans 60'den farklı ise G15'e git
MOV DPTR,#TAB60         ;Frekans 60, TAB60'ın Başlangıç adresini al
LJMP SON1
G15:

SON1:
MOV DPTRE2,DPH          ;Tablonu başlangıç adresini DPTRE2
MOV DPTRE1,DPL          ;ve DPTRE1'e at
POP CEKA                ;Saklanılan değerleri
                        ;geri al
POP DPL
POP DPH
MOV A,CEKA
MOV IE,#85H
RETI

TAB0:
DB 0F0H,0F0H,0,000H


TAB60:
DB 0FFH,0FEH,9,0FFH,0FEH,25,0FFH,0FEH,17,0FFH,0DFH,16,0FFH,0FEH,18
DB 0FFH,0E9H,22,0FFH,0E7H,6,0FEH,0F6H,38,0FFH,0F0H,6,0FFH,0FEH,4
DB 0FFH,0FDH,20,0FFH,0FEH,16,0FFH,0F3H,17,0FFH,0FEH,25,0FFH,0EBH,17
DB 0FFH,0FEH,16,0FFH,0FEH,20,0FFH,0F9H,4,0FFH,0F7H,6,0FEH,0C9H,38
DB 0FFH,0F7H,36,0FFH,0F9H,4,0FFH,0FEH,5,0FFH,0FEH,1,0FFH,0EBH,17
DB 0FFH,0FEH,25,0FFH,0F3H,17,0FFH,0FEH,1,0FFH,0FDH,5,0FFH,0FEH,4
DB 0FFH,0F0H,36,0FEH,0F6H,38,0FFH,0E7H,36,0FFH,0E9H,37,0FFH,0FEH,33
DB 0FFH,0DFH,1,0FFH,0FEH,17,0FFH,0FEH,25,0FFH,0FEH,24

DB 0FFH,0FEH,24,0FFH,0FEH,26,0FFH,0FEH,10,0FFH,0DFH,2,0FFH,0FEH,6
DB 0FFH,0E9H,38,0FFH,0E7H,36,0FEH,0F6H,37,0FFH,0F0H,36,0FFH,0FEH,32
DB 0FFH,0FDH,34,0FFH,0FEH,2,0FFH,0F3H,10,0FFH,0FEH,26,0FFH,0EBH,10
DB 0FFH,0FEH,2,0FFH,0FEH,34,0FFH,0F9H,32,0FFH,0F7H,36,0FEH,0C9H,37
```

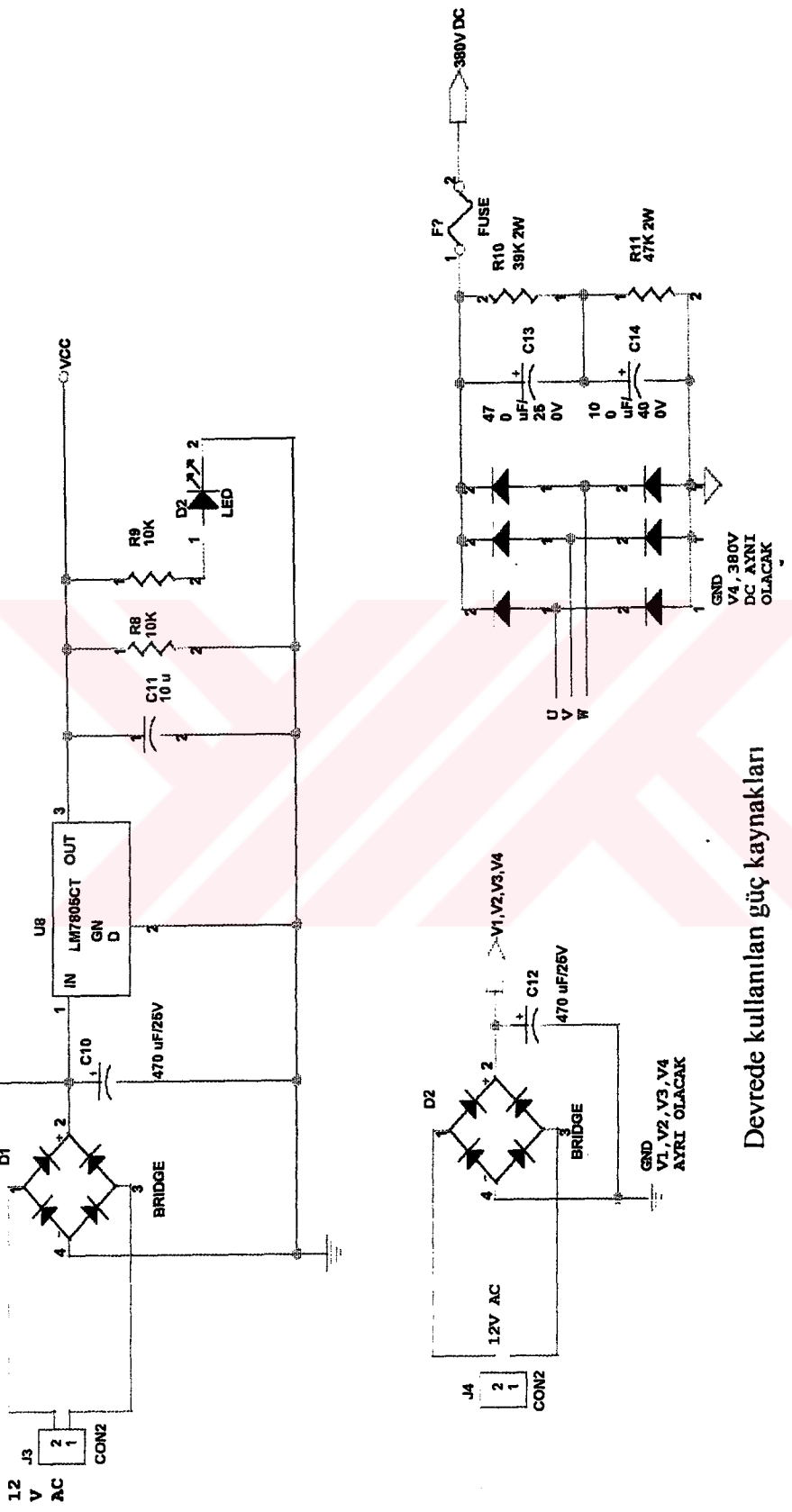
```
DB 0FFH,0F7H,33,0FFH,0F9H,32,0FFH,0FEH,40,0FFH,0FEH,8,0FFH,0EBH,10
DB 0FFH,0FEH,26,0FFH,0F3H,10,0FFH,0FEH,8,0FFH,0FDH,40,0FFH,0FEH,32
DB 0FFH,0F0H,33,0FEH,0F6H,37,0FFH,0E7H,33,0FFH,0E9H,41,0FFH,0FEH,9
DB 0FFH,0DFH,8,0FFH,0FEH,10,0FFH,0FEH,26,0FFH,0FEH,18

DB 0FFH,0FEH,18,0FFH,0FEH,22,0FFH,0FEH,20,0FFH,0DFH,4,0FFH,0FEH,36
DB 0FFH,0E9H,37,0FFH,0E7H,33,0FEH,0F6H,41,0FFH,0F0H,33,0FFH,0FEH,1
DB 0FFH,0FDH,5,0FFH,0FEH,4,0FFH,0F3H,20,0FFH,0FEH,22,0FFH,0EBH,20
DB 0FFH,0FEH,4,0FFH,0FEH,5,0FFH,0F9H,1,0FFH,0F7H,33,0FEH,0C9H,41
DB 0FFH,0F7H,9,0FFH,0F9H,1,0FFH,0FEH,17,0FFH,0FEH,16,0FFH,0EBH,20
DB 0FFH,0FEH,22,0FFH,0F3H,20,0FFH,0FEH,16,0FFH,0FDH,17,0FFH,0FEH,1
DB 0FFH,0F0H,9,0FEH,0F6H,41,0FFH,0E7H,9,0FFH,0E9H,25,0FFH,0FEH,24
DB 0FFH,0DFH,16,0FFH,0FEH,20,0FFH,0FEH,22,0FFH,0FEH,6

DB 0FFH,0FEH,6,0FFH,0FEH,38,0FFH,0FEH,34,0FFH,0DFH,32,0FFH,0FEH,33
DB 0FFH,0E9H,41,0FFH,0E7H,9,0FEH,0F6H,25,0FFH,0F0H,9,0FFH,0FEH,8
DB 0FFH,0FDH,40,0FFH,0FEH,32,0FFH,0F3H,34,0FFH,0FEH,38,0FFH,0EBH,34
DB 0FFH,0FEH,32,0FFH,0FEH,40,0FFH,0F9H,8,0FFH,0F7H,9,0FEH,0C9H,25
DB 0FFH,0F7H,24,0FFH,0F9H,8,0FFH,0FEH,10,0FFH,0FEH,2,0FFH,0EBH,34
DB 0FFH,0FEH,38,0FFH,0F3H,34,0FFH,0FEH,2,0FFH,0FDH,10,0FFH,0FEH,8
DB 0FFH,0F0H,24,0FEH,0F6H,25,0FFH,0E7H,24,0FFH,0E9H,26,0FFH,0FEH,18
DB 0FFH,0DFH,2,0FFH,0FEH,34,0FFH,0FEH,38,0FFH,0FEH,36

DB 0FFH,0FEH,36,0FFH,0FEH,37,0FFH,0FEH,5,0FFH,0DFH,1,0FFH,0FEH,9
DB 0FFH,0E9H,25,0FFH,0E7H,24,0FEH,0F6H,26,0FFH,0F0H,24,0FFH,0FEH,16
DB 0FFH,0FDH,17,0FFH,0FEH,1,0FFH,0F3H,5,0FFH,0FEH,37,0FFH,0EBH,5
DB 0FFH,0FEH,1,0FFH,0FEH,17,0FFH,0F9H,16,0FFH,0F7H,24,0FEH,0C9H,26
DB 0FFH,0F7H,18,0FFH,0F9H,16,0FFH,0FEH,20,0FFH,0FEH,4,0FFH,0EBH,5
DB 0FFH,0FEH,37,0FFH,0F3H,5,0FFH,0FEH,4,0FFH,0FDH,20,0FFH,0FEH,16
DB 0FFH,0F0H,18,0FEH,0F6H,26,0FFH,0E7H,18,0FFH,0E9H,22,0FFH,0FEH,6
DB 0FFH,0DFH,4,0FFH,0FEH,5,0FFH,0FEH,37,0FFH,0FEH,33

DB 0FFH,0FEH,33,0FFH,0FEH,41,0FFH,0FEH,40,0FFH,0DFH,8,0FFH,0FEH,24
DB 0FFH,0E9H,26,0FFH,0E7H,18,0FEH,0F6H,22,0FFH,0F0H,18,0FFH,0FEH,2
DB 0FFH,0FDH,10,0FFH,0FEH,8,0FFH,0F3H,40,0FFH,0FEH,41,0FFH,0EBH,40
DB 0FFH,0FEH,8,0FFH,0FEH,10,0FFH,0F9H,2,0FFH,0F7H,18,0FEH,0C9H,22
DB 0FFH,0F7H,6,0FFH,0F9H,2,0FFH,0FEH,34,0FFH,0FEH,32,0FFH,0EBH,40
DB 0FFH,0FEH,41,0FFH,0F3H,40,0FFH,0FEH,32,0FFH,0FDH,34,0FFH,0FEH,2
DB 0FFH,0F0H,6,0FEH,0F6H,22,0FFH,0E7H,6,0FFH,0E9H,38,0FFH,0FEH,36
DB 0FFH,0DFH,32,0FFH,0FEH,40,0FFH,0FEH,41,0FFH,0FEH,9,000H
```

EK-3

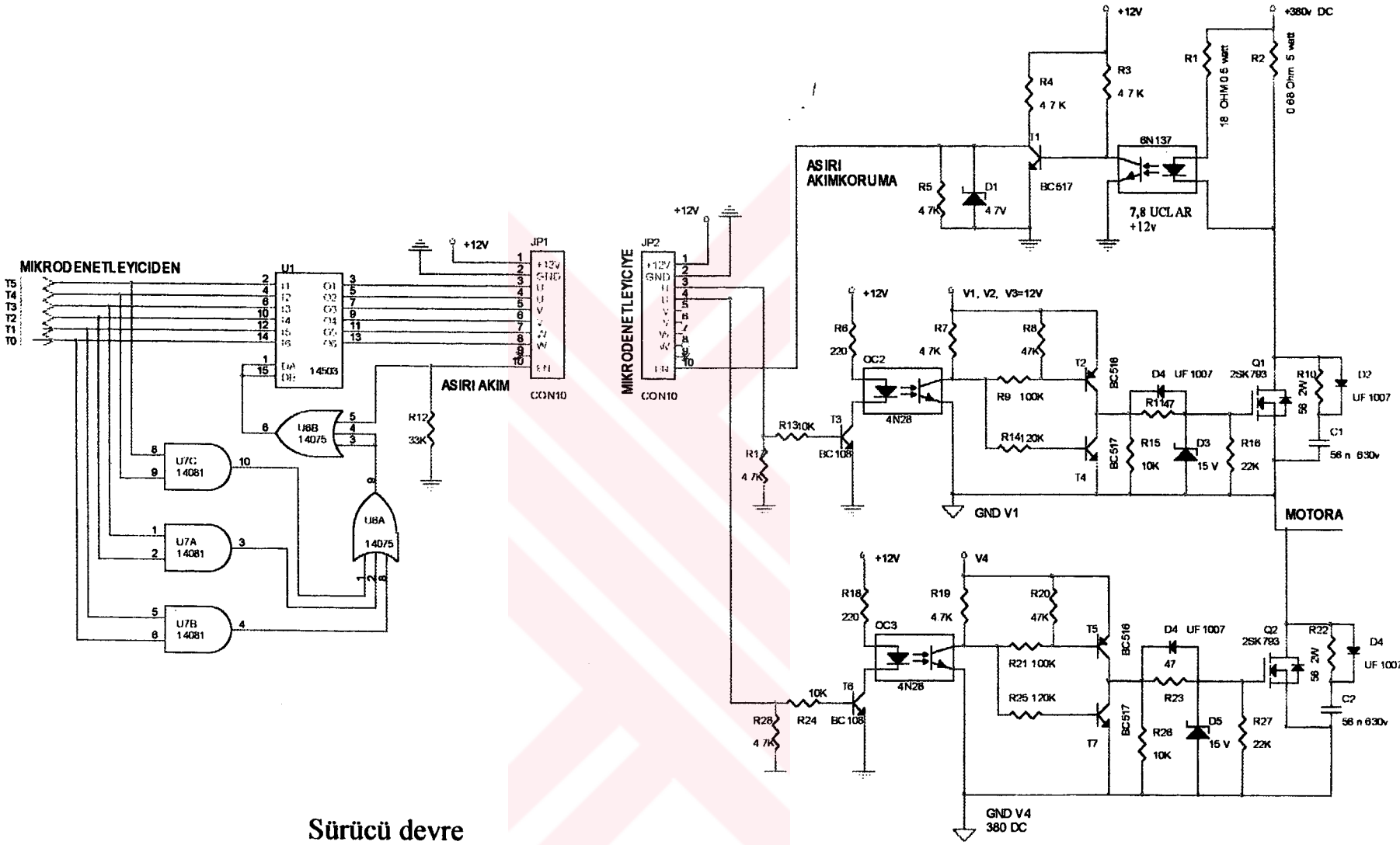
TASARLANAN CİHAZIN ELEKTRONİK DEVRE ŞEMALARI



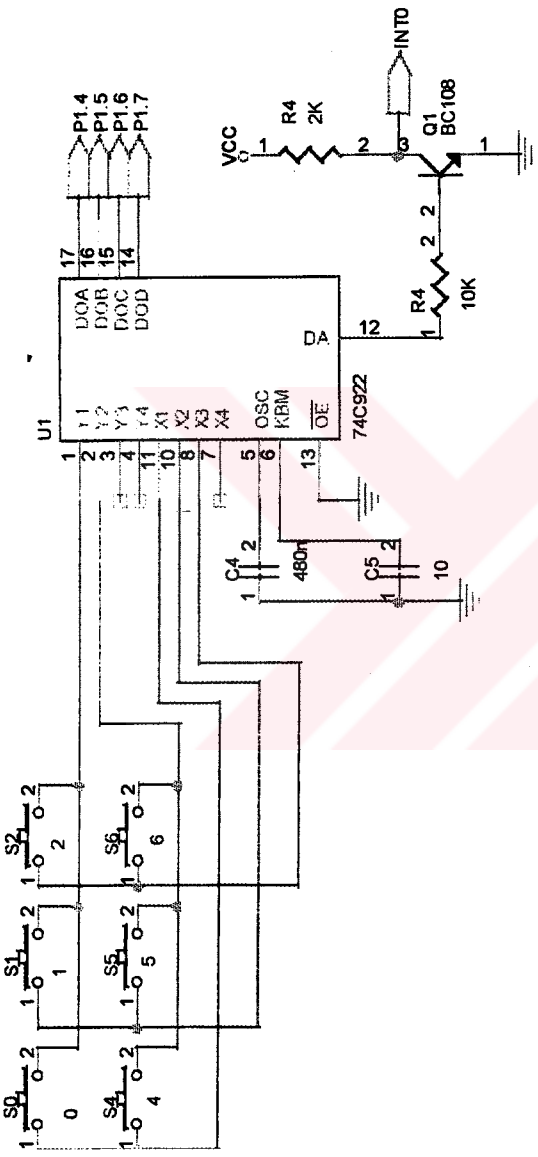
Ana devre şeması



Devrede kullanılan güç kaynakları



Sürücü devre



Tuş takımı devresi

ÖZGEÇMİŞ

Salih GÖRGÜNOĞLU, 1970 yılında Sungurlu'da doğdu. İlk ve orta okulu Ankara da tamamladı. Ankara Yenimahalle Teknik Lisesi Elektrik Bölümünü bitirdi.1991 yılında Gazi üniversitesi Teknik Eğitim Fakültesi Elektronik ve Bilgisayar Eğitimi bölümünü kazandı. Bu fakültenin Bilgisayar sistemleri ana bilim dalından 1995 yılında mezun oldu. Üniversiteden sonra bir yıl Elazığ Teknik Lisesinde, Bir yıl Ankara Yıldırım Beyazıt Endüstri Meslek ve Teknik lisesinde bilgisayar öğretmeni olarak çalıştı. 1997-1998 yılları arasında yedek subay olarak askerlik hizmetini yaptı. Şu an Gazi Üniversitesi Enformatik Bölümünde Öğretim görevlisi olarak çalışmakta Teknik Eğitim Fakültesi Kontrol Sistemleri Eğitimi anabilim dalında yüksek lisans yapmaktadır.