39290

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

KENDİNDEN AYARLAMALI PID KONTROLÖRÜNÜN MİKROKONTROLÖR TABANLI GERÇEKLENMESİ

39290

YÜKSEK LİSANS TEZİ

Elektronik ve Hab. Müh. Taner ARSAN

Tezin Enstitüye Verildiği Tarih : 24 Ocak 1994

Tezin Savunulduğu Tarih : 11 Şubat 1994

Tez Danışmanı : Prof.Dr. Atilla BİR

Diğer Jüri Üyeleri : Doç. Dr. İbrahim EKSİN

: Doç.Dr. Metin GÖKAŞAN

T.C. YÜKSEKÖĞRETİM KURULU DOKÜMANTASYON MERKEZİ

ŞUBAT 1994

ÖNSÖZ

Bu çalışmanın konusunun seçiminden , sonuca ulaşılmasına kadar her aşamasında değerli yardımlarını, özendirici ve titiz katkılarını esirgemeyen Hocam, Sayın Prof. Dr. Atilla BİR'e teşekkürlerimi sunmayı bir borç bilirim.

Taner ARSAN

Şubat 1994

İÇİNDEKİLER

ÖZET

Bu yüksek lisans tezinde, mikrokontrolör tabanlı bir PID kontrolörü tasarlanmış ve Åstrom ve Hägglund tarafından geliştirilen Röle yöntemi ile bu kontrolöre, kendinden ayarlama özelliği kazandırılmıştır. Gerçeklenen kendinden ayarlamalı kontrolör, çeşitli sistemlere uygulanarak sonuçlar yorumlanmıştır.

Çalışmada hem dijital PID kontrolörünü hem de kendinden ayarlama işlemini gerçekleyen Intel 80C31 mikrokontrolörlü bir kart tasarlanmıştır. Bu özellik şimdiye kadar gerçeklenen kendinden ayarlamalı kontrolörlere mikrokontrolör tabanlı bir yaklaşım getirmekte olup, sıcaklık kontrolundan asenkron motor hız kontroluna kadar geniş bir alanda kullanılmaya elverişlidir. Bu varsayımı kanıtlamak için kontrolör, asenkron motorun hız kontrolu gibi kontrol edilmesi zor olan bir sisteme uygulanmıştır. Kendinden ayarlama yönteminin lineer sistemlerin kontrolu için önerildiği gözönünde bulundurulursa, asenkron motor kontrolunda referans hıza ulaşılırken görülen %4 mertebesindeki bir aşım dışında elde edilen sonuçlar yeterlidir. Yapılan ölçümler çalışmanın son bölümünde ayrıntılı olarak verilmiştir.

SUMMARY

IMPLEMENTATION OF AN AUTO-TUNING PID CONTROLLER BASED ON A MICROCONTROLLER

In this thesis, it is aimed to implement an auto-tuning PID controller based on a microcontroller. For this purpose, at the first step a digital PID controller has been designed and then this has been enhanced by means of Relay Method which is one of the best, but simple auto-tuning method, proposed by Åstrom and Hägglund.

In this project, Intel 80C31 microcontroller board, providing both digital PID controller and auto-tuning properties, has been designed. This was a new approach to auto-tuning PID controllers using such advanced Integrated Circuits. On the other hand, it is possible to see some microprocessors application on implementing auto-tuning PID controllers, during development period. However, microcontroller based applications differ from microprocessor based ones. Microprocessor application can be efficient when they carry out various program of users, multiple data processing, calculating sensitive numerical values. On the contrary, microcontroller based system is quite useful in case of particular states which do not require reprogramming and runs unchanged programs. The advantage of such a system is containing the main components of a digital computer in single chip. A microcontroller individually has a Central Processing Unit-CPU, a memory unit and intensive peripherals such as Analog to Digital Converter-ADC and Digital to Analog Converter-DAC. Therefore, it can be suitable for digital control by adding some extra components.

The structure of PID controller which is implemented in this study, has been illustrated in Figure 1.

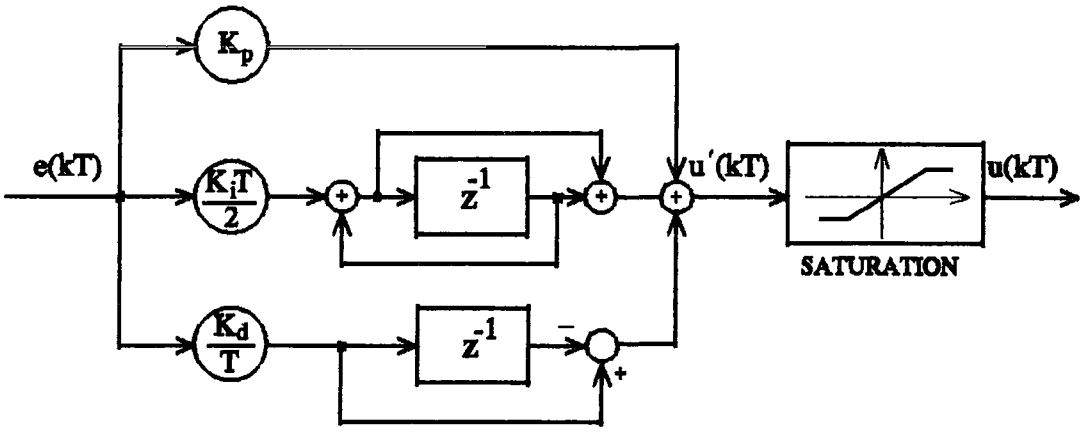


Figure 1 - Structure of implemented PID controller.

An important property of implemented PID controller is saturation on control signal. Control signal has been changed between 0-10 VDC. Error signal which is difference between reference signal and feedback, is interpreted by PID controller algorithm. If control signal is required more than 10 VDC, it is set to maximum voltage 10 VDC. Whereas the control signal is required negative values, it is set to 0 VDC.

There are several methods that can be enhanced digital PID controller to auto-tuning controller. The most efficient of the simple methods is a frequency domain method called Relay Method. Relay Method, proposed by Åstrom and Hägglund, has been the modification of Ziegler-Nichols Frequency Method. The structure of auto-tuning PID controller based on Relay Method is shown in Figure 2.

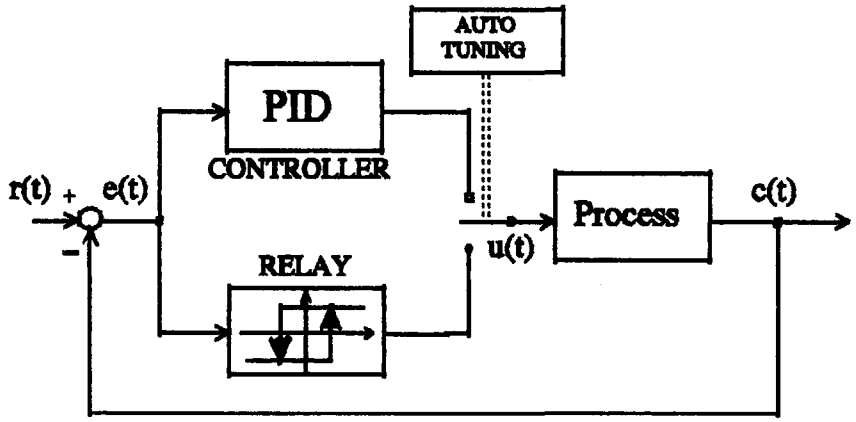


Figure 2 - Auto-tuning PID controller based on the Relay Method.

With a relay, the method gives a control signal to the process whose period is close to the crossover frequency of open-loop system.

Describing function of relay with hysteresis,

$$-\frac{1}{N(a)} = -\frac{\pi}{4d}\sqrt{a^2-h^2} - j\frac{\pi h}{4d} \qquad (1)$$

where d is the relay amplitude, h is the hysteresis width. This function can be regarded as a straight line parallel to the real axis (see Figure 3a and 3b). Thus, by means of the intersection of the Nyquist curve with this straight line, the critical point is determined. In this point, steady-state oscillation is occurred. This gives T_c-the ultimate period and K_c-ultimate gain.

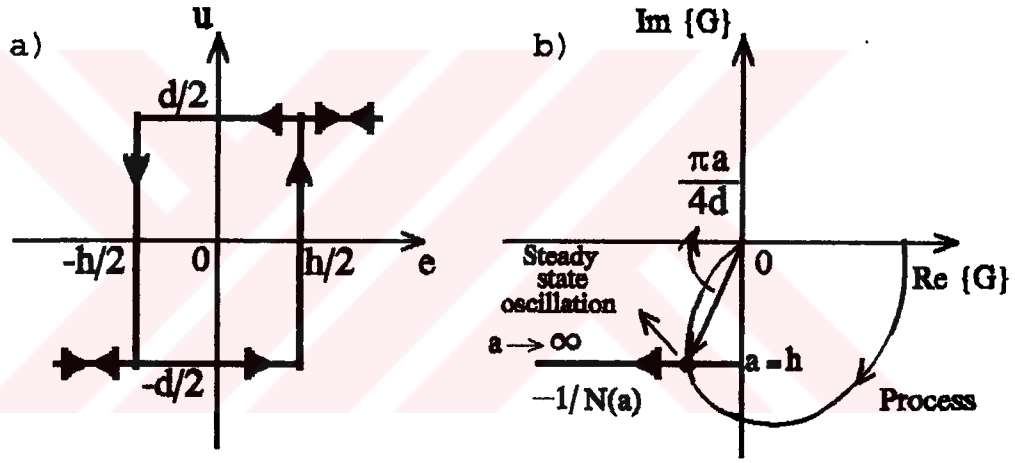


Figure 3.a - Input-output characteristic of the relay with hysteresis,
3.b - Changing describing function of the relay with hysteresis in G complex plane and steady-state oscillation point.

d relay amplitude must be selected in a suitable range. a-error magnitude and ω_c-oscillation frequency are determined from steady-state oscillation. K_c-ultimate gain and T_c-ultimate period can be written as,

$$K_c = \frac{4d}{\pi a} \qquad (2)$$

$$T_c = \frac{2\pi}{\omega_c} \qquad (3)$$

Finally, controller parameters are determined by

$$K_p = 0,6\,K_c \quad , \quad K_i = 2 / T_c \quad , \quad K_d = 0,12\;T_c \qquad (4)$$

In this work, methods suitable for linear systems are applied to a complex control application such as speed control of an Induction Machine having nonlinear characteristics. Then results, obtained from this system, are interpreted. In Figure 4, block diagram of this application is shown.

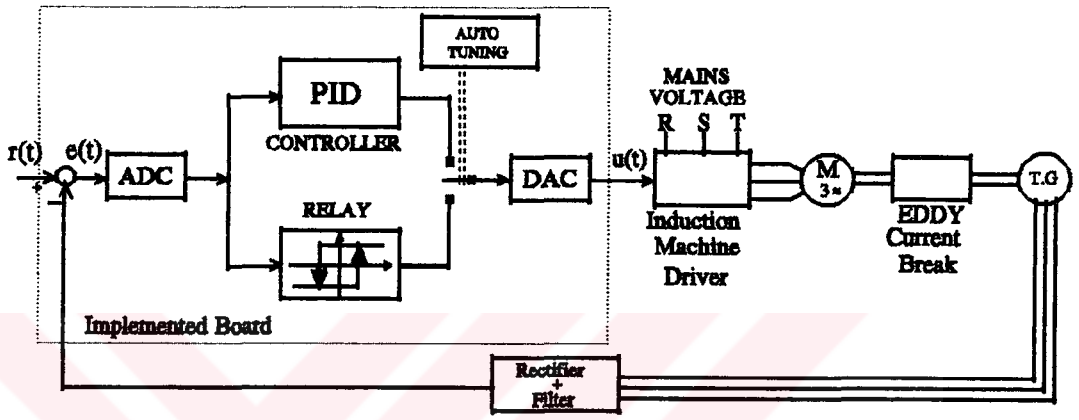


Figure 4 - Speed control of an Induction Machine.

In this application the induction machine and its driver are considered as a plant to be controlled. Control signals feeding the plant is produced by a PID algorithm which process reference and feedback signals. After the control signal is converted into a continuous signal between 0-10V, it is applied to the driver of induction machine. The driver produce pulse width modulated voltage as an equivalence of the input signal.

In the next some amount of speed data are collected to obtain graphical speed characteristics. Thus, performance of the controller can be figured out.

The characteristics in Figure 5 has been obtained for $K_p = 2$, $K_i = 2$ and $K_d = 0$. In this case the overshoot is found to be slightly high. This, actually, disturb the motor speed particularly in case of instant load. But motor speed turn back to its reference value when the load is released.

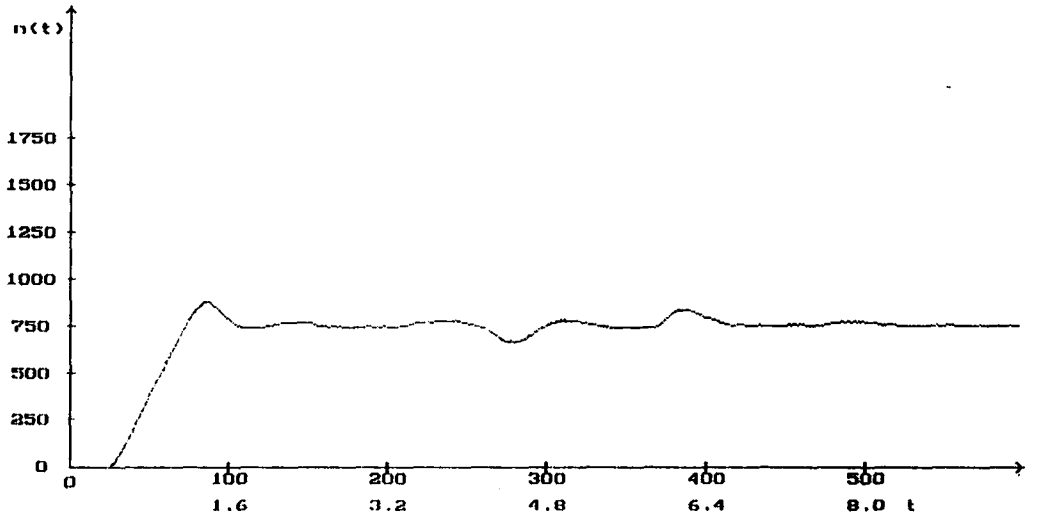


Figure 5 - Graphical speed characteristic of induction machine for controller parameters $K_p = 2$, $K_i = 2$, $K_d = 0$.

When auto-tuning process is applied to the system, the reference value is set to desired speed value. A relay with hysteresis characteristics oscillates the system. This is illustrated in Figure 6.

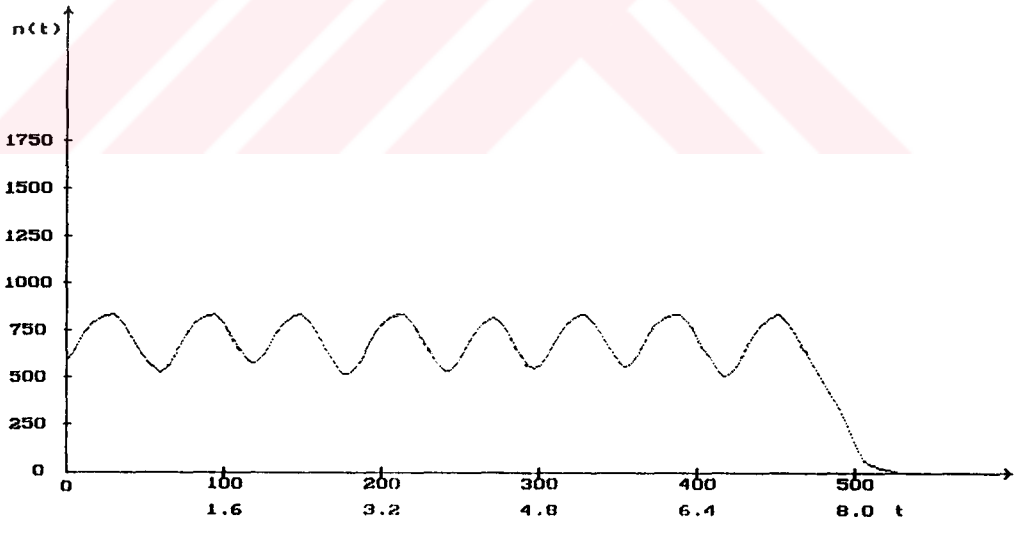


Figure 6 - Oscillation of the system for the speed reference of 750 rpm.

At the end of this routine, K_c-ultimate gain and T_c-ultimate period can be determined from oscillation period. Then controller parameters are obtained as follows,

$$K_p = 2,063 \quad , \quad K_i = 2,101 \quad , \quad K_d = 0,114$$

For sampling period of T=16ms, these parameters are loaded to controller. The characteristics in Figure 7 has been obtained for $K_p = 2,063$, $K_i = 2,101$ and $K_d = 0,114$. In this case the overshoot is smoother than it is found in previous example and induction machine reaches the given reference speed.

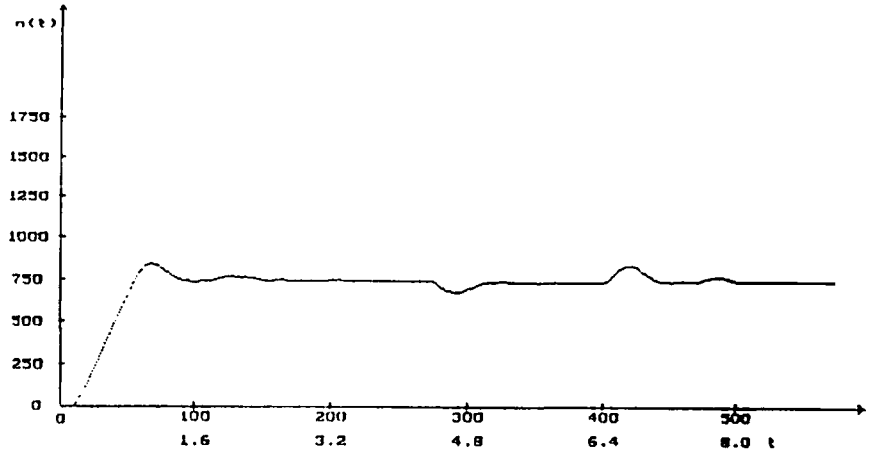


Figure 7 - After auto-tuning graphical speed characteristic of induction machine for controller parameters $K_p = 2,063$, $K_i = 2,101$, $K_d = 0,114$.

From collected measurement, it has been shown that microcontroller based auto-tuning PID controller performs efficiently. In case study of speed control induction machine, the transient behavior of speed is found very sufficient ignoring small oscillations and low level of overshoot. In this work, it also shown that how these small oscillations can be removed totally, particularly in case of speed control of an induction machine.

When the implemented controllers is applied to the non-linear systems, the results are given above. If the plant has a non-linear characteristic such as induction machine, it is advisable to use one of the optimization algorithms. If DC motor is used instead of induction machine, it can be easily said that the auto-tuning PID controller provides better results.

BÖLÜM 1

GİRİŞ

Endüstride PID kontrolörünün kullanımı çok çeşitli evrelerden geçerek günümüze ulaşmıştır. Analog PID kontrolörleriyle başlayan ve bugün artık kendinden ayarlamalı PID kontrolörüne kadar gelen bu yapının gelişmesi sırasında mikroişlemci tabanlı uygulamalara da rastlanır. Ancak mikrokontrolör tabanlı bir uygulama mikroişlemci tabanlı olandan oldukça farklıdır. Mikroişlemcili bir sistem çeşitli kullanıcı programlarını yürütmek, çok sayıda veri işlemek ve duyarlı sayısal hesaplamaları gerçekleştirmek için kullanıldığında yüksek verim sağlayabilmektedir. Mikrokontrolör tabanlı bir sistem ise yeniden programlama gerektirmeyen ve sürekli sabit bir programın yürütüldüğü durumlarda çok kullanışlıdır. Bunun nedeni mikrokontrolörlerin, bilgisayarın temel bileşenlerini tek bir yapıda barındırmasından kaynaklanmaktadır. Tek başına bir mikrokontrolör, merkezi işlem birimi, bellek, giriş-çıkış ve çevre birimlerini kapsamaktadır. Dolayısıyla çok az bir eleman ilavesiyle sistem dijital kontrolöre uygun hale getirilebilir.

Gerçeklenen PID algoritmasının önemli bir özelliği kontrol işaretinde satürasyon uygulanmasıdır. Verilen parametre değerlerinden ve oluşan hatadan yararlanarak bulunan kontrol işareti 0-10V aralığında sürekli bir

işarete dönüştürülmektedir. Eğer PID programı sonucunda, sistemi kontrol etmek için 10V'dan daha yüksek bir gerilim gerektiği sonucuna varılırsa, çıkışa 10V maksimum gerilim uygulanır. Negatif değerler gerektiğinde ise çıkışa minimum gerilim 0 V kontrol işareti uygulanır.

Dijital PID kontrolörünü, kendinden ayarlamalı hale getirebilmek için çeşitli yöntemler bulunmaktadır. Basit yöntemler içinde en etkin olanı bir frekans tanım bölgesi yöntemi olan Röle yöntemidir.

Åström ve Hägglund tarafından önerilen Röle yöntemi [1], [2], Ziegler-Nichols sürekli salınım yönteminin [2] geliştirilmiş şeklidir. Burada histerezisli bir röle vasıtasıyla proses girişine, periyodu açık çevrim sisteminin kesim frekansına yakın bir işaret uygulanır. Kendinden ayarlama modunda sistem bir osilasyona girer. Kararlı bir salınım elde edildiğinde osilasyonun T_c periyodu ve K_c kritik kazancı belirlenir. Rölenin d genliği uygun seçilerek, sinüsoidal olduğu varsayılan $e(t) = a \sin \omega_c t$ için, a hata genliği ve ω_c salınım frekansı kapalı çevrimli sistemde oluşan salınımdan elde edilir. Sistemi salınıma sokan K_c kazancı ve salınım periyodu

$$K_c = \frac{4d}{\pi a} \qquad (1.1) \qquad\qquad T_c = \frac{2\pi}{\omega_c} \qquad (1.2)$$

ifadelerinden elde edilir. Bu değerlerden kontrolör katsayıları,

$$K_p = 0{,}6\,K_c \;,\; K_i = 2/T_c \;,\; K_d = 0{,}12\;T_c \qquad (1.3)$$

şeklinde belirlenir. Lineer sistemler için geçerli olan

yöntem bu çalışmada lineer olmayan bir sisteme uygulanmış ve elde edilen sonuçlar değerlendirilmiştir.

Çalışmanın ikinci bölümünde, sürekli ve dijital kontrol sistemleri incelenmiş ve PID kontrolörünün genel yapısı verilmiştir. Üçüncü bölümde, kendinden ayarlama prensipleri, mevcut yöntemler ve çalışmada kullanılan yöntem etraflı bir şekilde anlatılmıştır. Mikrokontrolör yazılımı ve ilave donanımla, dijital PID kontrolörünün ve kendinden ayarlama işleminin ne şekilde gerçeklendiği dördüncü bölümde açıklanmıştır. "Kontrol Edilen Sistem ve Kontrolörün Özellikleri" adlı beşinci bölümde, asekron motor ve sürücüsü ile mikrokontrolör kartındaki elemanlar tanıtılmıştır. Altıncı ve son bölümde, kontrol edilen sistemden ölçülen hız grafikleri verilmekte ve sonuçlar yorumlanmaktadır.

BÖLÜM 2

KONTROL TEORİSİ

Bu bölümde, kendinden ayarlamalı PID kontrolörünün gerçeklenmesi aşamasına kadar bilinmesi gereken temel bilgiler özetlenmektedir. Bu amaçla öncelikle sürekli (analog) kontrol sistemleri tanıtılacak, birinci ve ikinci mertebeden, ölü zamanlı sistemlere ilişkin diferansiyel denklemler verilecek ve bu sistemlerin zaman tanım bölgesinde birim basamak cevapları ve frekans tanım bölgesi davranışları incelenecektir. İkinci aşama olarak dijital kontrol sistemlerinin temel elemanları tanıtılacak ve üstünlükleri belirtilecektir. Son olarak PID kontrolörünün analog ve dijital olarak nasıl gerçekleneceği anlatılacaktır.

2.1. SÜREKLİ KONTROL SİSTEMLERİ

Sürekli kontrol sistemleri zaman tanım bölgesinde lineer diferansiyel denklemlerle, frekans tanım bölgesinde ise transfer fonksiyonlarıyla ifade edilirler.

Burada çeşitli mertebeden sistemler zaman ve frekans tanım bölgesinde incelenerek bu sistemlerin davranışları ve bunları karakterize eden katsayıların ne şekilde belirleneceği gösterilecektir.

2.1.1. BİRİNCİ MERTEBEDEN SİSTEMLER VE ZAMAN TANIM BÖLGESİ DAVRANIŞLARI

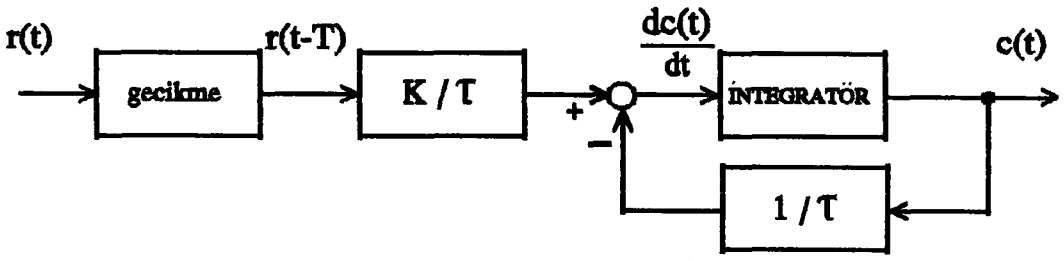
Ölü zamanlı birinci mertebeden dinamik bir sistemi tanımlamak için

$$\frac{dc(t)}{dt} = -\frac{1}{\tau}c(t) + \frac{K}{\tau}r(t-T) \qquad (2.1)$$

diferansiyel denklemi yeterlidir.

Burada ; r(t): sistem girişi (veya referans),
c(t): sistem çıkışı,
K : kazanç,
τ : zaman sabiti,
T : ölü zaman'dır.

Sisteme ilişkin diferansiyel denklem Şekil 2.1'deki gibi blok diyagramları ile ifade edilebilir.



Şekil 2.1- (2.1) denklemiyle verilen sisteme ilişkin blok diyagramı.

(2.1) denklemiyle verilen sisteme ilişkin transfer fonksiyonunu elde etmek için , ilk koşullar sıfır kabulü ile doğrudan Laplace dönüşümü alınırsa,

$$s\,C(s) = -\frac{1}{\tau}C(s) + \frac{K}{\tau}R(s)\,e^{-Ts} \ ,$$

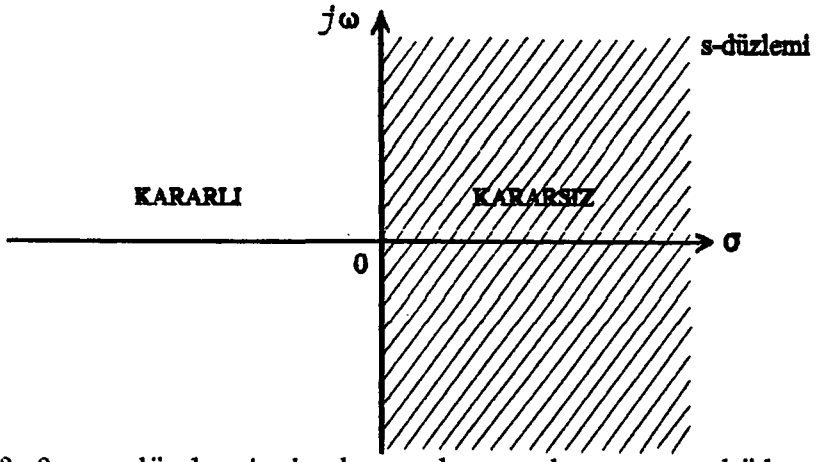
ile

$$G_s(s) = \frac{C(s)}{R(s)} = \frac{K\,e^{-Ts}}{\tau s + 1} \qquad (2.2)$$

elde edilir.

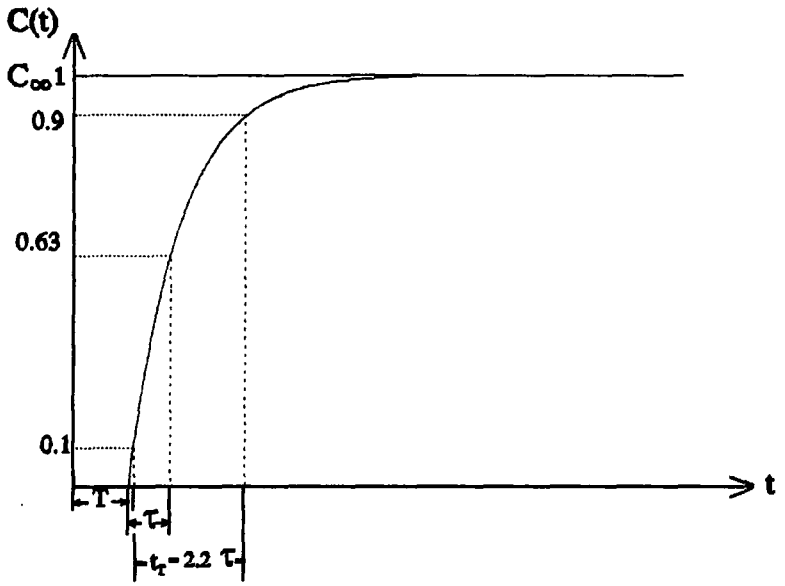
Transfer fonksiyonunda, paydanın köklerinin reel kısımları sistemin kararlılığını veya kararsızlığını belirler. Birinci mertebeden, (2.2) 'deki transfer fonksiyonu ile ifade edilen sistem için bu kök $s_1 = -1/\tau$ 'dir.

Sistemin asimptotik kararlı olabilmesi için, transfer fonksiyonunun paydasının bütün kökleri $Re\{s_1\} < 0$ olacak şekilde tanımlanmalıdır. Eğer transfer fonksiyonunun paydasının bir veya bir kaç kökü $Re\{s_1\} > 0$ olacak şekilde seçilirse sistem kararsız olacaktır. $Re\{s_1\} = 0$ için $j\omega$ ekseninde katsız kutup bulunması durumu hariç sistem yine kararsız olacaktır [3]. s-düzleminde kararlı ve kararsız bölgeler Şekil 2.2'de gösterilmiştir.



Şekil 2.2- s-düzleminde kararlı ve kararsız bölgeler.

Birinci mertebeden ölü zamanlı $G_s(s) = K e^{-Ts}/(\tau s+1)$ şeklindeki transfer fonksiyonuna sahip sistemin birim basamak cevabı Şekil 2.3 'de verilmiştir.



Şekil 2.3 - Birinci mertebeden ölü zamanlı sistemin birim basamak girişe cevabı.

Birim basamak giriş uygulandığında birinci mertebeden sistem davranışını belirleyen büyüklükler şöyle tanımlanabilir:

Son değer (C_∞); $t \to \infty$ için elde edilen sabit çıkış değeridir. Son değer burada, transfer fonsiyonundaki K-kazancına eşit olacaktır.

Ölü zaman (T); birim basamak işretinin ilk değişmeye başladığı an ile sistem cevabında fark edilebilir bir değişimin oluştuğu an arasında arasında kalan süredir.

Yükselme zamanı (t_r); Sistem cevabının, son değerin %10 'dan %90 'a ulaşma süresidir (veya son değerin %90'ına ulaşması için gereken süredir). Birinci mertebeden sistem için $t_r = 2,2\tau$ 'dur. $t = \tau$ için sistem çıkışının , son değerin %63 'üne ulaşmış olur.

2.1.2. İKİNCİ MERTEBEDEN SİSTEMLER VE ZAMAN TANIM BÖLGESİ DAVRANIŞLARI

İkinci mertebeden bir sistem için normalize edilmiş diferansiyel denklem;

$$\frac{d^2 c(t)}{dt} + 2\xi\omega_n \frac{dc(t)}{dt} + \omega_n^2 c(t) = K\omega_n^2 r(t-T) \qquad (2.3)$$

şeklinde verilebilir. Laplace transformasyonu alınır ve düzenlenirse ikinci mertebeden sisteme ilişkin transfer fonksiyonu

$$G_s(s) = \frac{C(s)}{R(s)} = \frac{K\omega_n^2 \; e^{-Ts}}{s^2 + 2\xi\omega_n s + \omega_n^2} \qquad (2.4)$$

şeklinde elde edilir.

Burada; ω_n (rad/s): doğal frekans ($\omega_n = 2\pi f_n$),
ξ (.) : amortisman oranı,
T (s) : ölü zaman'dır.

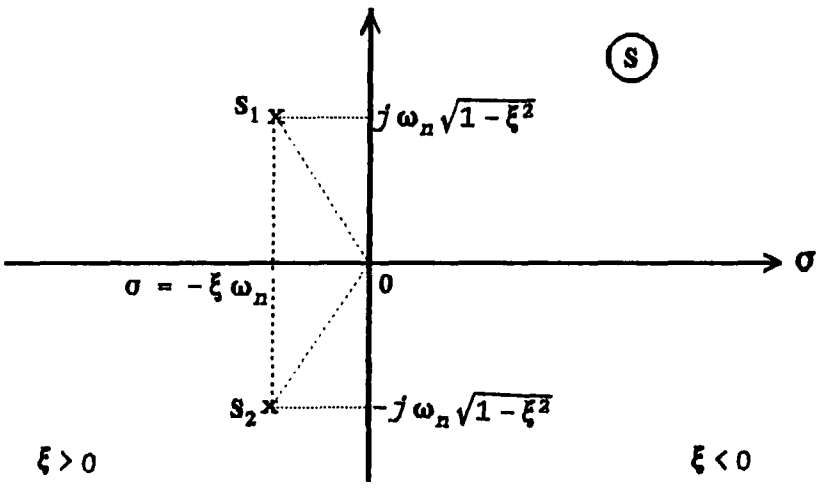
Transfer fonksiyonu paydasındaki ifade,

$$s_{1,2} = -\xi\omega_n \mp \omega_n\sqrt{\xi^2 - 1} \qquad (2.5)$$

olarak elde edilir. Sistemin kararlılığı ξ'ya bağlı olarak incelenirse:

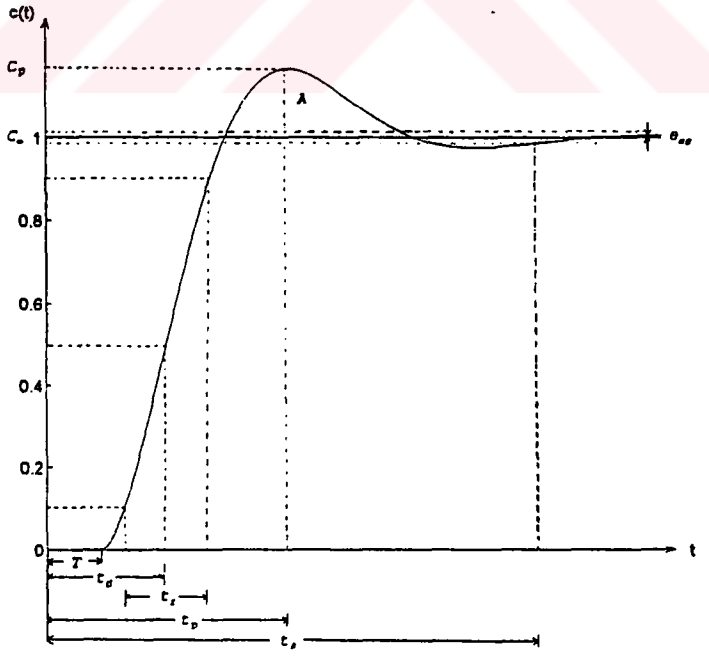
$\xi < 0$: kararsız sistem,
$\xi > 0$: asimptotik kararlı sistem,
$\xi \geq 1$: reel kökler (periyodik olmayan cevap),
$\xi < 1$: karmaşık kökler (osilasyonlu cevap), bu durumda kökler $s_{1,2} = -\xi\omega_n \mp j\omega_n\sqrt{1-\xi^2}$ şeklinde elde edilir.

Karmaşık köklerin s-düzlemindeki konumu Şekil 2.4'te verilmiştir.



Şekil 2.4 - Amortisman oranının fonksiyonu olarak ikinci mertebeden sistemin kökleri.

İkinci mertebeden sistemin birim basamak cevabı Şekil 2.5'te görülmektedir.



Şekil 2.5 - İkinci mertebeden ölü zamanlı sistemin birim basamak girişe cevabı.

Birim basamak giriş uygulandığında ikinci mertebeden sistemin davranışını belirleyen büyüklükler şöyle tanımlanabilir:

Gecikme zamanı (t_d): Sistem cevabının son değerin %50'sine ulaşma zamanıdır.

Yükselme zamanı (t_r): Sistem cevabının , son değerinin %10'undan %90'ına ulaşması için geçen zamandır (veya t_d noktasında, cevap eğrisine çizilen teğetinin 0 ile 1 arasında kalan süredir).

Tepe zamanı (t_p): Sistem cevap eğrisinin maksimum değerine ulaşma süresidir.

Yerleşme zamanı (t_s): Sistem çıkışının son değer etrafında %δ (±%10 , ±%5 veya ±%2) 'lik bir tolerans bölgesi içine bir daha hiç çıkmamak üzere girdiği zamandır.

Maksimum değer (C_p): Sistem cevap eğrisinin ulaştığı en yüksek noktaya karşılık düşer.

% Aşım (A): Sistem cevap eğrisinin C_p maksimum değeri ile C_∞ son değeri arasındaki farkın yüzde olarak ifadesidir:

$$\%A = \frac{C_p - C_\infty}{C_\infty} 100 \qquad (2.6)$$

Kararlı hal hatası (e_{ss}): $t \to \infty$ için r(t)=R=sabit referans değeri ile C_∞ son değer arasındaki farktır. Yani kısaca $e_{ss} = R - C_\infty$ yazılabilir.

Zaman tanım bölgesinde, birim basamak şeklindeki giriş işareti için sistem cevabının aşım, yükselme, gecikme ve yerleşme zamanı değerleriyle sistemin referans girişi ile sistem cevabı arasındaki fark şeklinde tanımlanan e(t) hatasının küçük değerler alması istenir [4].

2.1.3. BİRİNCİ VE İKİNCİ MERTEBEDEN SİSTEMLERİN FREKANS TANIM BÖLGESİ DAVRANIŞLARI

Lineer bir sistemin sinüsoidal giriş işaretine cevabı, aynı frekanslı, ama fazı kayık genliği farklı sinüsoidal bir işaret şeklinde olur. Şekil 2.6'dan görülebileceği üzere $u_1(t) = A\sin(\omega t)$ şeklindeki giriş işareti için sistemin yanıtı $u_2(t) = A'\sin(\omega t+\varphi)$ şeklindedir.

Şekil 2.6 - Lineer sistemin davranışı.

Transfer fonksiyonu

$$G(s) = \frac{K}{\tau s + 1} \tag{2.7}$$

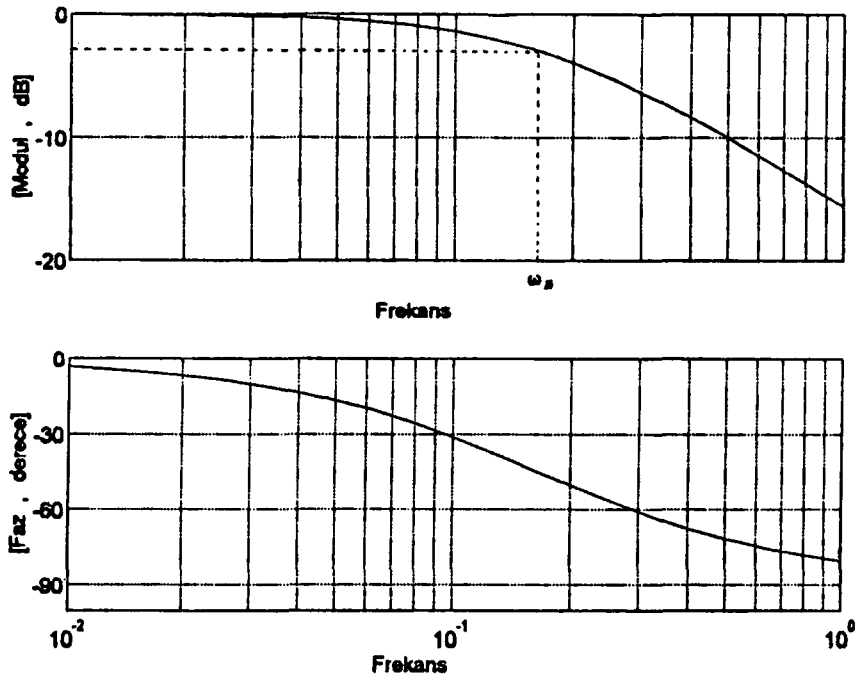
şeklinde olan birinci mertebeden, ölü zamansız bir sistem göz önünde bulundurulsun. Buradan $s=j\omega$ yazılarak sistemin frekans cevabı;

$$G(j\omega) = \frac{K}{j\omega\tau + 1} = |G(j\omega)|\, e^{j\Phi(\omega)} \tag{2.8}$$

şeklinde elde edilir. Frekans cevabına ilişkin modül ve faz ifadeleri ise;

$$|G(j\omega)| = \frac{K}{[\tau^2\omega^2 + 1]^{1/2}} \quad , \quad \Phi(\omega) = -arctg\,\tau\omega \tag{2.9}$$

olarak elde edilir. Bu sistemin frekans cevabına ilişkin modül ve faz eğrileri Şekil 2.7'de verilmiştir.



Şekil 2.7 Birinci mertebeden sistemin frekans cevabı.

Sinüsoidal giriş uygulandığında birinci mertebeden sistemin davranışını belirleyen Band genişliği (ω_b); sıfır frekans kazancının 3 dB zayıfladığı frekans olarak tanımlanır.

Transfer fonksiyonu;

$$G(s) = \frac{K\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} = \frac{K}{(\frac{s}{\omega_n})^2 + 2\xi(\frac{s}{\omega_n}) + 1} \qquad (2.10)$$

şeklinde olan ikinci mertebeden, ölü zamansız sistem göz-önünde bulundurulur ve $s = j\omega$ yazılırsa,

$$G(j\omega) = \frac{K}{[1 - (\frac{\omega}{\omega_n})^2] + j2\xi(\frac{\omega}{\omega_n})} \qquad (2.11)$$

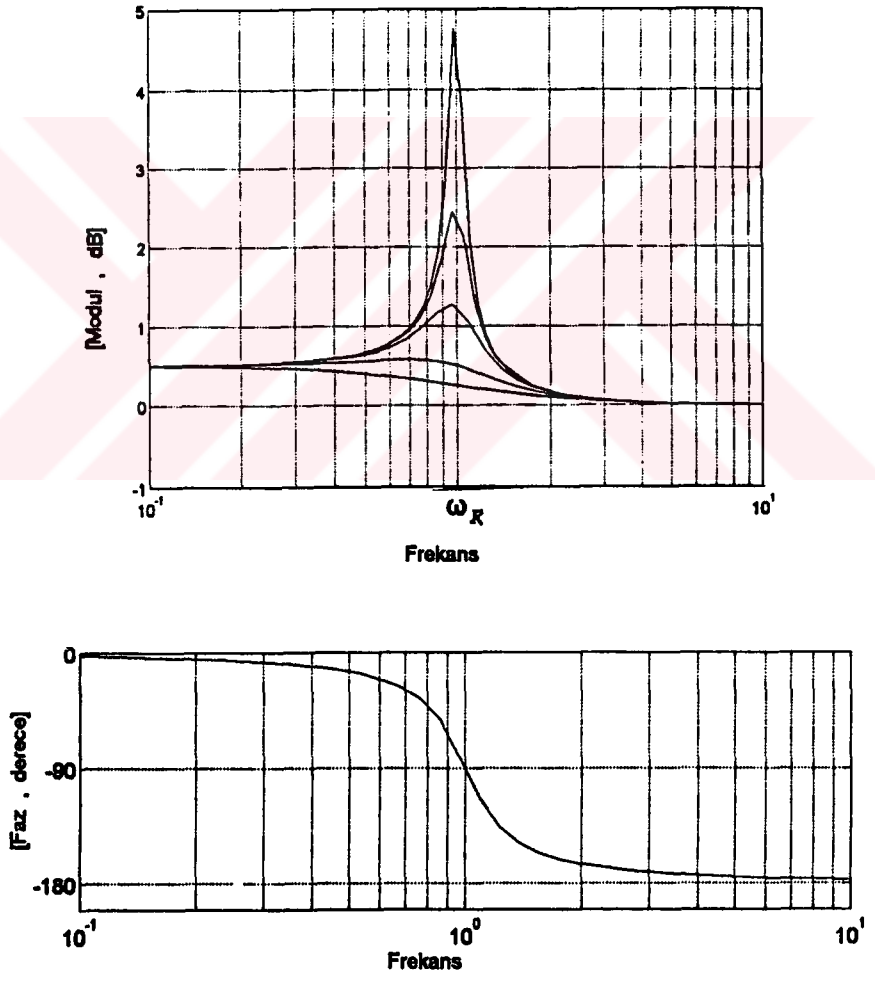
elde edilir.

Frekans cevabına ilişkin modül ve faz ifadesi

$$|G(j\omega)| = \frac{K}{[(1-\frac{\omega}{\omega_n})^2 + (2\xi\frac{\omega}{\omega_n})^2]^{1/2}} \qquad (2.12)$$

$$\phi = -\text{arc}\,tg\,\frac{2\xi(\frac{\omega}{\omega_n})}{1-(\frac{\omega}{\omega_n})} \qquad (2.13)$$

olarak bulunur. Frekans cevabına ilişkin modül ve faz eğrileri Şekil 2.8 'de verilmiştir.



Şekil 2.8 - İkinci mertebeden sistemin frekans cevabı.

Sinüsoidal giriş uygulandığında birinci mertebeden sistemin davranışını belirleyen büyüklükler şöyle tanımlanabilir:

Band genişliği (ω_B): Sıfır frekans kazancının 3 dB zayıfladığı frekanstır. Yani,

$$\omega_B = \omega_n \left[(1-2\xi^2) + \sqrt{4\xi^4-4\xi^2+2} \right]^{1/2} \qquad (2.14)$$

şeklinde ifade edilebilir.

Rezonans frekansı (ω_R): Modül eğrisinde $|G(j\omega)|$ 'nın maksimum değere eriştiği andaki frekanstır. Bu değer band genişliğinin bir ölçütüdür. Rezonans frekansı arttıkça sistem band genişliği de artacaktır. Rezonans frekansı değerinin teorik olarak elde edilebilmesi için $|G(j\omega)|$'nın ω'ya göre türevinin sıfıra eşitlenmesi yeterli olacaktır. Bu durumda rezonans frekansı için,

$$\omega_R = \omega_n\sqrt{1-2\xi^2} \quad , \quad \xi < 0,707 \qquad (2.15)$$

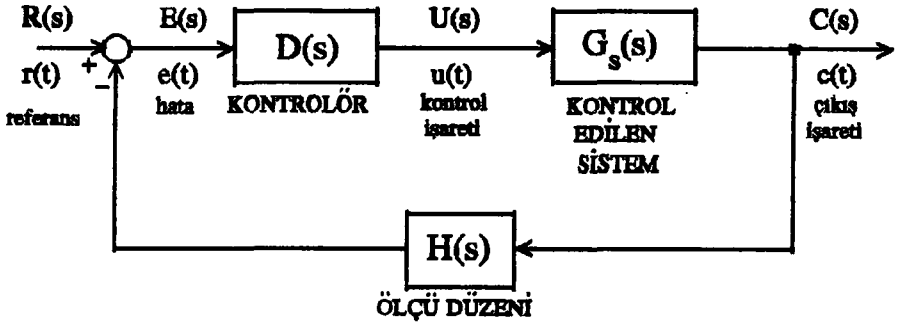
değeri bulunur.

2.1.4. KAPALI ÇEVRİM KONTROL SİSTEMLERİ

Önceki bölümlerde, bundan sonra kontrol edilecek sistem olarak adlandırılacak birinci ve ikinci mertebeden sistemlerin zaman tanım bölgesi ve frekans tanım bölgesi davranışları incelendi.

Kontrol çevrimlerinin amacı kontrol edilen sistemlerin belirli bir amaç doğrultusunda, çıkış büyüklüğü bir ölçü düzeni ile ölçülerek ve bir referans işaretle karşılaştırılarak, bir kontrolör yardımıyla kontrol etmektir.

Böyle bir sistemin elemanları transfer fonksiyonları ile ifade edilerek Şekil 2.9'daki blok diyagramı ile gösterilebilir. Şekil 2.9 ile verilen blok diyagramında r(t); referans değerini, c(t);sistem çıkışını e(t)=r(t)-c(t) hata değerini, u(t) ise kontrol işaretini ifade eder.



Şekil 2.9 - Kapalı çevrim kontrol sisteminin blok diyagramı.

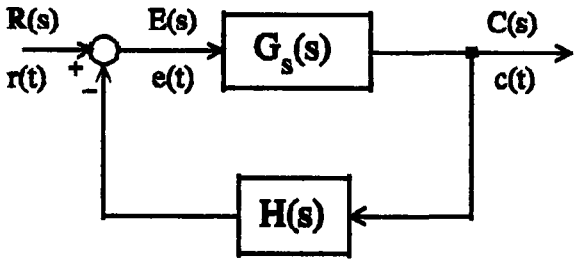
Blok diyagramı Şekil 2.9 ile verilen sistemin kapalı çevrim transfer fonksiyonu

$$T(s) = \frac{C(s)}{R(s)} = \frac{D(s)\,G_s(s)}{1+D(s)\,G_s(s)\,H(s)} \qquad (2.16)$$

olarak bulunur. Bu denklem $G(s) = D(s)\,G_s(s)$ tanımı ile,

$$T(s) = \frac{C(s)}{R(s)} = \frac{G(s)}{1+G(s)\,H(s)} \qquad (2.17)$$

şeklinde Şekil 2.10'de verilen en basit kontrol sisteminin yapısı elde edilmiş olur.



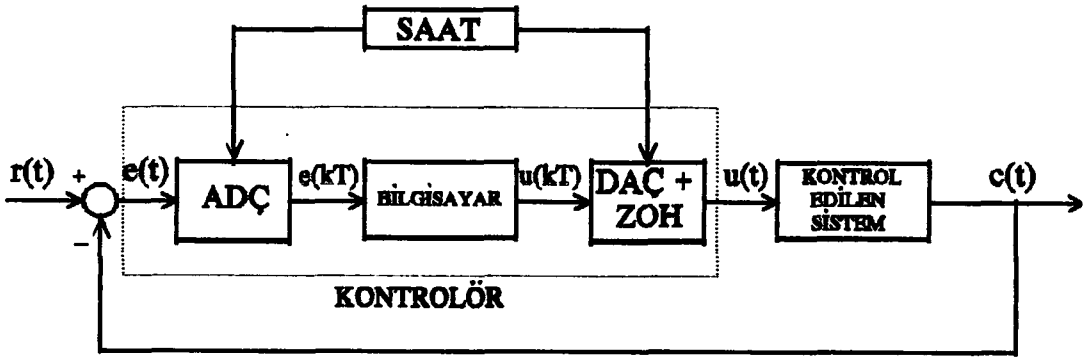
Şekil 2.10 - En basit kontrol sisteminin indirgenmiş blok diyagramı.

2.2. DİJİTAL KONTROL SİSTEMLERİ

Bu bölümde dijital kontrol sistemlerinin temel kavramları verilecek ve temel elemanları tanıtılacaktır. Dijital kontrol sistemlerinin yapısını belirleyebilmek için önce sürekli işaretlerin dijital işaretlere çevrilmesi işlemleri, örnekleme tutma işlemleri dijital işaretlerin tekrar sürekli işarete çevrilmesi işlemleri açıklanacaktır.

2.2.1. TEMEL KAVRAMLAR VE ELEMANLAR

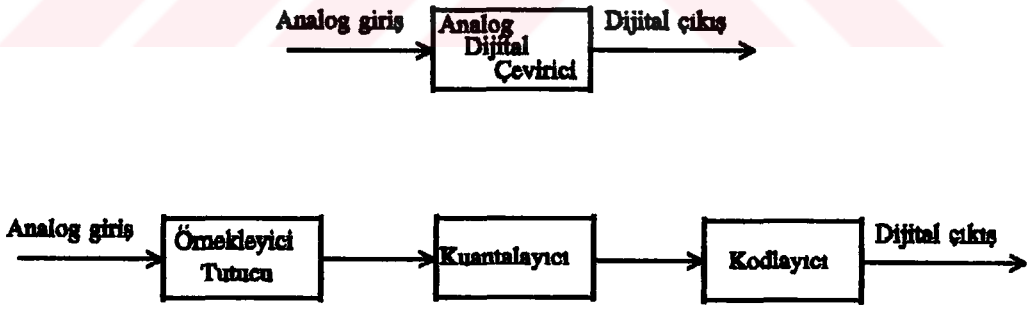
Dijital işaret, sürekli işaretin belirli bir örnekleme frekansı ile örneklenmesi, tutulması, kuantalanması (sayısal değerlendirilmesi) ve özel bir şekilde kodlanmasıyla elde edilir. Buna göre belirli bir aralıkla değişen sürekli bir işaret 0 ve 1'lerden oluşmuş bir darbe katarı şeklinde dijital olarak elde edilir. Bu dijital işaret bir bilgisayar (mikroişlemci, mikrokontrolör veya dijital işaret işleyici) vasıtasıyla değerlendirilerek sistemi kontrol eden bir işaret üretilir. Şekil 2.11'de bir dijital kontrol sistemi blok diyagramı verilmiştir [3].



Şekil 2.11 - Kapalı çevrim dijital kontrol sisteminin blok diyagramı.

Burada r(t) referans değeriyle c(t) sistem çıkışı arasındaki e(t) hatası kT örnekleme zamanlarında Analog Dijital Çevirici-ADÇ vasıtasıyla e(kT) dijital veriye çevrilir. Bilgisayar (veya mikroişlemci, mikrokontrolör veya dijital işaret işleyici) e(kT) işaretini, kontrol algoritmasını kullanarak yorumlar ve u(kT) şeklinde yeni bir kontrol işareti üretir. Dijital Analog Çevirici-DAÇ vasıtasıyla bu kontrol işareti sıfırıncı mertebeden tutucu (ZOH) tarafından örnekleme anlarında sabit tutulan bir sürekli işarete çevrilir. Yukarıda adı geçen ADÇ, DAÇ ve ZOH elemanlarının yapılarını ve çalışma biçimlerini açıklamak yararlı olacaktır.

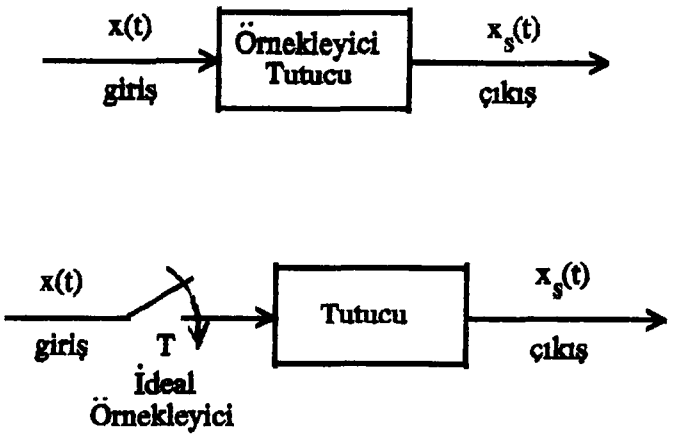
Sürekli işaretten dijital işaret dizisini elde etmede kullanılan elemana Analog Dijital Çevirici adı verilir. Bu elemanın yapısı ve işlevi [5] Şekil 2.12'de verilmiştir.



Şekil 2.12 - Analog Dijital Çevirici-ADÇ elemanının yapısı.

Şekil 2.12 ile verilen ADÇ yapısı içinde bulunan ilk eleman örnekleme ve tutma elemanıdır. Bu elemanın görevi, belirli aralıklarla sürekli işaretten örnekler alıp bu

değerleri belirli bir süre tutmaktır. Elemanın yapısı Şekil 2.13'te tanımlanmıştır.



Şekil 2.13 - Örnekleme-tutma elemanının yapısı.

Bu yapıda T örnekleme periyodunu belirtmektedir. Örneklemede işlem süresi, örnekleme periyodundan çok küçük olduğu için, sıfır kabul edilmiştir. Örnekleme elemanının çalışma prensibi Shannon teoremi veya Nyquist örnekleme teoremi olarak da bilinen örnekleme teoremine dayanmaktadır. Bu teoremin tanımı verilmeden önce band sınırlı işaret kavramı açıklanmalıdır.

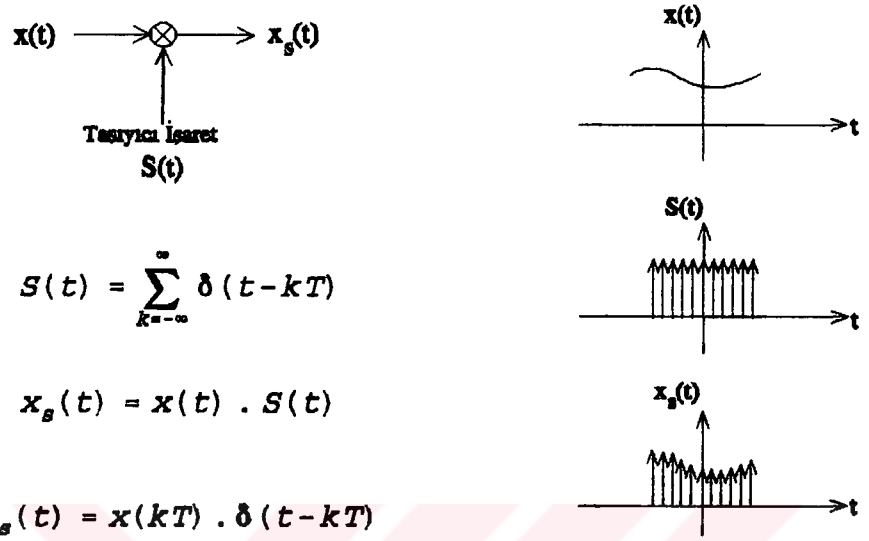
$$X(\omega) = 0 \quad , \quad |\omega| \geq \omega_m \tag{2.18}$$

koşulunu sağlayan x(t) işareti için "ω_m rad/s. ile band sınırlıdır" denir. Bu durumda örnekleme teoremi için; ω_m rad/s. ile band sınırlı bir x(t) işaretinin, bu işaretten eşit aralıklarla (T) alınan örnek değerlerinden tek ve bozulmasız olarak yeniden elde edilmesi için gerek ve yeter koşul

$$f_s = \frac{1}{T} \geq 2\omega_m \tag{2.19}$$

olmasıdır (f_s : örnekleme frekansı) [6].

Şekil 2.14'te örnekleme devresi ve ara işlemleri gösterilmiştir.



$$S(t) = \sum_{k=-\infty}^{\infty} \delta(t-kT)$$

$$x_s(t) = x(t) \cdot S(t)$$

$$x_s(t) = x(kT) \cdot \delta(t-kT)$$

Şekil 2.14 - İdeal örnekleme ve işlemleri.

Örnekleme frekansının teoremde belirtilenden küçük seçilmesi örneklenen işaretin yeniden elde edilmesini imkansızlaştırır bu olaya örtüşme etkisi denir. Kapalı çevrim kontrol sistemlerinde bu durum sistem kararlılığını bozar. Kontrol sistemlerinde örnekleme frekansı genellikle $f_s = (6 \text{-} 25)\, f$ olacak şekilde belirlenir. Endüstride kullanılan prosesler için örnekleme periyodları şöyledir:

Servo mekanizmalarda $T = 0{,}001 \text{-} 1\ s$

debi kontrolünde $T = 5 \text{-} 10\ s$

basınç kontrolünde $T = 1 \text{-} 5\ s$

sıcaklık kontrolünde $T = 10 \text{-} 45\ s$

kimyasal sistemlerde $T = 10 \text{-} 180\ s$

çimento sanayii ve kurutma sistemlerinde $T = 20 \text{-} 45\ s$.

Tutma elemanı olarak doğrudan sıfırıncı mertebeden tutucu-ZOH kullanılır. Amaç bir sonraki örneklenmiş değer elde edilinceye kadar eski işaretin sabit tutulmasıdır. Lineer bir eleman olan sıfırıncı mertebeden tutucunun transfer fonksiyonunu bulmak için impuls cevabını birim basamak cinsinden ifade etmek ve Laplace dönüşümünü almak yeterlidir. Bu durumda

$$g_{ZOH}(t) = u(t) - u(t-T) \qquad (2.20)$$

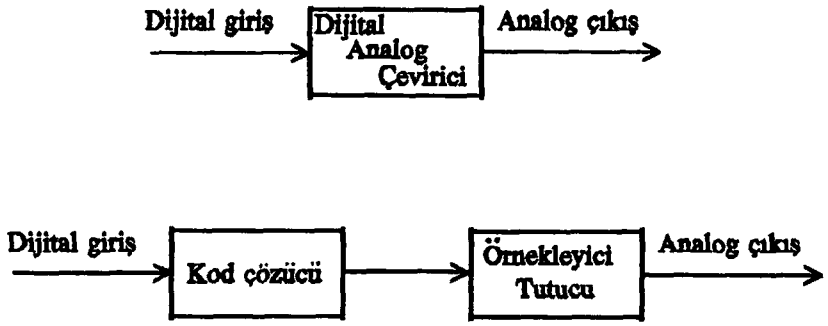
impuls cevabından,

$$G_{ZOH}(s) = \frac{1-e^{-Ts}}{s} \qquad (2.21)$$

transfer fonksiyonu elde edilir.

ADÇ yapısında, örnekleyici tutucu elemandan sonra kuantalayıcı adı verilen birim yer almaktadır. Örneklenip tutulan her değer önceden belirlenmiş düzeylere karşı düşürülür ve bu düzeylerden gerçek değere yaklaştırma yapılır. Bu işleme kuantalama, bu işlemi gerçekleştiren birime kuantalayıcı adı verilir. Son olarak her kuanta düzeyi özel bir sözcükle kodlanır. Bu son birim kodlayıcı olarak bilinir. Bu şekilde sürekli işaret, darbe katarı şeklinde dijital işarete dönüştürülmüş olur.

Kontrolör vasıtasıyla sisteme uygulanacak dijital kontrol işareti belirlendikten sonra bu işaret DAÇ vasıtasıyla sürekli işarete çevrilir (Burada elde olunan bu işaretin şekli tam olarak sürekli bir işarete uymayabilir. Bunun sebebi ADÇ dönüşümü sırasında oluşan kuantalama hatasıdır). DAÇ elemanının yapısı Şekil 2.15'de verilmiştir.



Şekil 2.15 - Dijital analog çevirici-DAÇ yapısı.

Burada gerçeklenen işlem önce dijital işaretin kodunun çözülmesi, DAÇ'ye uygulanan referans gerilimiyle orantılı olarak bir analog gerilim değeri belirlenmesi ve bu gerilim değerinin çıkışta bir sonraki değer gelene kadar sabit tutulmasıdır.

Bu şekilde gerçeklenen dijital kontrol sisteminin sürekli kontrol sistemine göre çeşitli avantajları bulunmaktadır. Dijital kontrol sistemleri gürültü ve bozucu etkilerden daha az etkilendikleri ve sistem parametrelerinin değişimlerine daha az duyarlı olduklarından daha güvenilir kontrol sistemleri oluştururlar [7], [5]. Dijital kontrolörlerle, kuantalama aralığının uygun seçilmiş olması halinde, yeterli derecede hassasiyete ulaşılır. Ayrıca programlama işleminin çok basit ve esnek olmasından dolayı analog sistemlerde aşılması çok güç olan tasarım değişikliklerini, dijital sistemlerde gerçekleştirmek mümkündür. Bir önemli avantaj ise bu sistemlerin giderek ucuzlayan fiatlarıdır. Bilgisayar mimarisindeki gelişmeye elverişli yapı yeni elemanlar ürettikçe, eskiden üretilen mamüllerin fiatlarının düşmesi de kaçınılmaz olmaktadır.

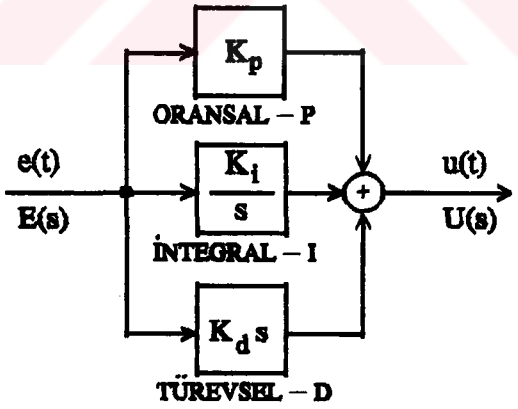
Gerçek zamanda çalışmaları ve çok iyi işaret duyarlığına sahip olmaları da analog kontrolörlerin üstün olduğu durumlardır. Ancak sağladığı yararlar ve getirdiği kolaylıklar nedeniyle dijital kontrol sistemlerinin kullanımı giderek yaygınlaşmaktadır.

2.3. PID KONTROLÖRÜNÜN GENEL YAPISI

Paralel yapıdaki PID kontrolörü zaman tanım bölgesinde

$$u(t) = K_P e(t) + K_I \int_0^t e(t)\,dt + K_D \frac{d}{dt} e(t) \qquad (2.22)$$

denklemiyle ifade edilebilir. Bu yapı Şekil 2.16'da gösterilmiştir.



Şekil 2.16 - Paralel yapıdaki PID kontrolörünün zaman tanım bölgesindeki blok diyagramı.

PID kontrolörünün z-tanım bölgesi transfer fonksiyonunu yaklaşık elde edebilmek için;

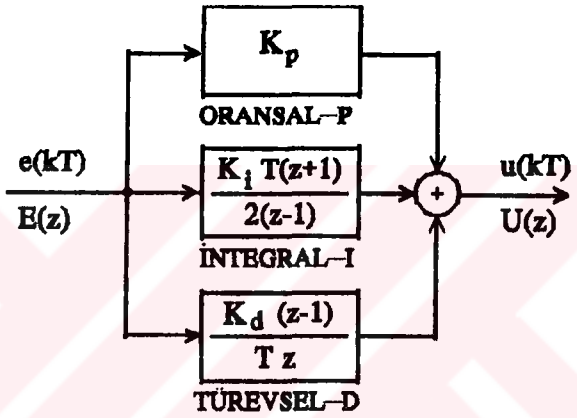
İntegral işlemi trapezoidal yaklaşım ile,

$$X(z) = Z\{K_i \int_0^t e(t)\,dt\} \approx \frac{K_i T(z+1)}{2(z-1)} E(z) \qquad (2.23)$$

Türev işlemi yaklaşık türev yaklaşımı ile,

$$X(z) = Z\{K_d \frac{de(t)}{dt}\} \approx K_d \frac{(z-1)}{Tz} E(z) \qquad (2.24)$$

ifade edilir. Bu durumda paralel yapıdaki PID kontrolörü z-tanım bölgesinde Şekil 2.17'de görüldüğü gibidir.



Şekil 2.17 - Paralel yapıdaki PID kontrolörünün z-tanım bölgesindeki blok diyagramı.

PID kontrolörünün z-tanım bölgesindeki transfer fonksiyonu

$$\frac{U(z)}{E(z)} = K_p + K_i \frac{T(z+1)}{2(z-1)} + K_d \frac{(z-1)}{Tz} \qquad (2.25)$$

olarak elde edilir. Buradan,

$$\frac{U(z)}{E(z)} = \frac{(2K_d + 2K_pT + K_iT^2)\,z^2 - (4K_d + 2K_pT - K_iT^2)\,z + 2K_d}{2Tz(z-1)} \qquad (2.26)$$

ifadesi türetilebilir.

BÖLÜM 3

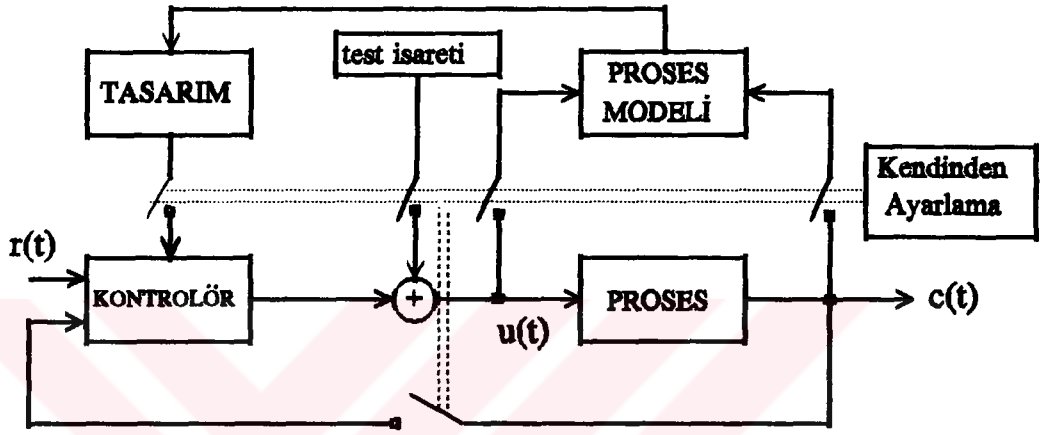
KENDİNDEN AYARLAMALI KONTROLÖRLER

Bu bölüme kadar özetlenen bilgilerin ışığı altında bir dijital kontrolör tasarlamak ve çeşitli referans giriş işaretleri için bu kontrolörün davranışını incelemek mümkündür. Gerçeklenen bu kontrolör ile çeşitli prosesler kontrol edilebilir. Bu kontrolörün çalışması, kullanıcının uygun parametre değerlerini (orantı, integral, türev katsayılarını) kontrolöre aktarması, örnekleme zamanını belirlemesi ve çalışma işlemini başlatmasıyla sağlanır. Eğer parametre değerleri uygun seçilmemişse sistem gerekli davranışı gösteremeyeceğinden kullanıcının parametre değerlerini yeniden vermesi gerekecektir. Bu durumda kullanıcı, doğru değerleri bulana kadar arama yapmak zorundadır. İşte bu ayarlama işlemini kullanıcı yerine kontrolörün kendisinin yapması işlemine "Kendinden Ayarlama" denir. Burada kullanıcıya düşen görev, sadece kontrolörü kendinden ayarlama durumuna getirmektir. Kontrolör, uygun parametre değerlerini bulup, kullanıcıya önerir. Böylece manuel olarak uzun sürecek ugun parametre değerlerini arama işlemi saniyeler mertebesinde yerine getirilir.

Bu bölümde kendinden ayarlama prensipleri açıklanacak ve mevcut yöntemler incelenecektir.

3.1. KENDİNDEN AYARLAMA PRENSİPLERİ

Giriş bölümünde de açıklandığı gibi kendinden ayarlama, kullanıcıdan gelen istek üzerine kontrolörün kendi kendini ayarlayabilme özelliğidir. Kendinden ayarlamalı bir kontrolörünün blok diyagramı Şekil 3.1'de verilmiştir.



Şekil 3.1 - Kendinden ayarlamalı kontrolörün blok diyagramı

Kendinden ayarlamalı sistemlerde, proses modeli ile hakiki proses arasında farklılık görüldüğünde, diğer bir deyişle verilen kontrolör parametreleri sistemi kontrol etmede yetersiz kaldığında, sistem kendinden ayarlama moduna geçirilir. İlkin test işaretlerinden yararlanarak proses modeli güncelleştirilir, ikinci aşamada yeni kontrolör parametreleri belirlenir. Ayarlama işlemi bittikten sonra kontrolöre yeni kontrolör parametreleri yüklenerek sistem normal çalışma moduna geçirilir [2].

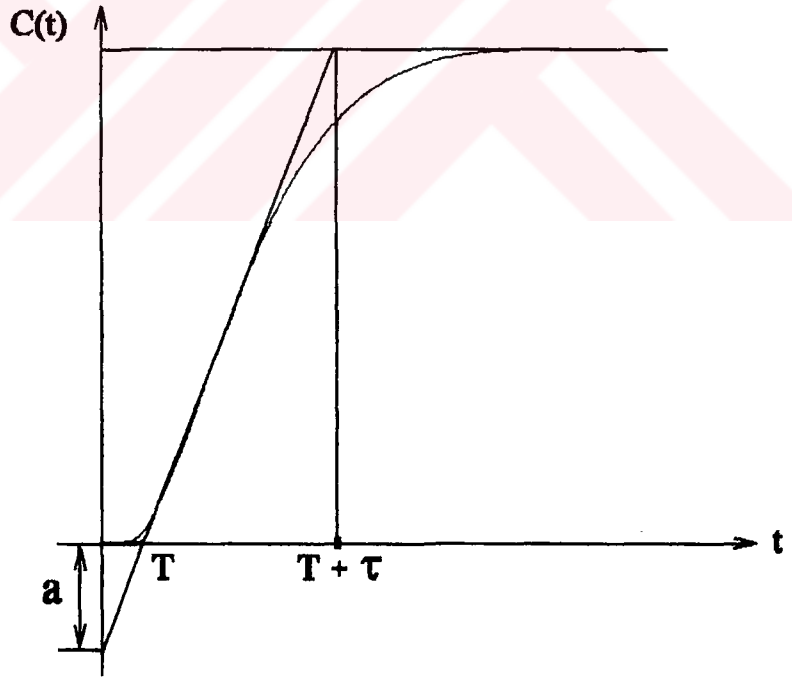
3.2. KENDİNDEN AYARLAMA İÇİN ÖNERİLEN YÖNTEMLER

Kendinden ayarlamalı sistemler için kullanılan yöntemler zaman tanım bölgesi ve frekans tanım bölgesi yöntemleri olarak iki gruba ayrılır. Burada, Ziegler-Nichols [2] tarafından önerilen ve Åstrøm-Hägglund [1], [2]

tarafından geliştirilen yöntemler açıklanacak ve elde edilen sonuçlar karşılaştırılacaktır.

3.2.1. ZIEGLER-NICHOLS BASAMAK CEVABI YÖNTEMİ

Bu yöntemde, açık çevrimli sistemin birim basamak cevabından yararlanılarak yeni kontrolör parametreleri belirlenir [2]. Sistemin birim basamak cevabına ilişkin eğriye, yükselme zamanı içinde çizilen maksimum eğimli teğet, a-maksimum eğimi ve T-başlangıçtaki ölü zamanı belirler (Bkz. Şekil 3.2). Elde edilen bu değerlerden ve Tablo 3.1'de verilen bulunmuş bulunan ifadelerden yararlanılarak, kontrolör parametreleri belirlenir.



Şekil 3.2 - Ziegler-Nichols basamak cevabı yönteminde a-maksimum eğim ve T-ölü zamanının bulunması.

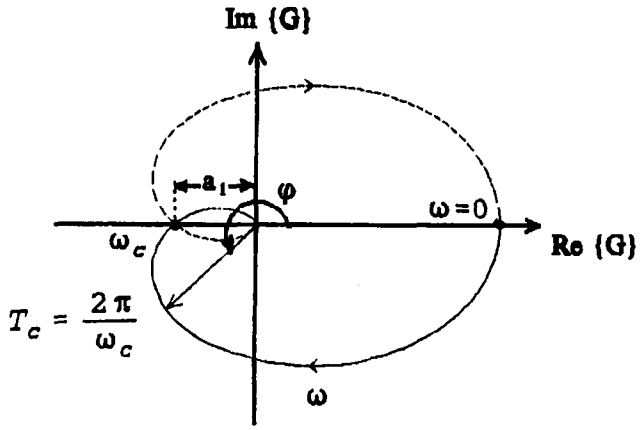
Tablo 3.1 - Ziegler-Nichols basamak cevabı yönteminden kontrolör parametrelerinin belirlenmesi.

Kontrolör Tipi	Kp	Ki	Kd
P	1/a	——	——
PI	0,9/a	1/3T	——
PID	1,2/a	1/2T	T/2

Bu yöntemin olumsuz tarafı, bulunan değerlerin sistem kapalı çevrimli hale getirildiğinde yetersiz kalmalarıdır. Bir önemli nokta da T-ölü zamanının τ-zaman sabitine bölünmesiyle elde edilen değerin büyük olması durumudur. Böyle bir durum oluştuğunda T-ölü zamanını kompanze eden başka yöntemler önerilir [2].

3.2.2. ZIEGLER-NICHOLS FREKANS CEVABI YÖNTEMİ

Bu yöntem, Ziegler-Nichols sürekli salınım yöntemi olarak da bilinir. Bu yöntemde amaç sistemin kararlılık sınırını belirlemektir. Bu durum sisteme ait Nyquist diyagramında eğrinin negatif gerçek eksenle kesiştiği a_1 noktasından ve ω_c kesim frekansından belirlenir (Bkz. Şekil 3.3). Bu nokta $K_c = 1/a_1$-kritik kazanç (ultimate gain) ve $T_c = 2\pi/\omega_c$-kritik periyodundan (ultimate period) belirlenir. Bu noktayı bulabilmek için referans girişe sabit bir değer uygulanır ve kontrolörde Ki=0 ve Kd=0 alınarak Kp parametresinin değeri arttırılır . Sistemi kararlılık sınırında osilasyona sokan Kp değeri Kc olarak alınır. Bu kazanç değerindeki salınımın periyodu ise Tc'yi belirler. Son olarak Ziegler-Nichols tarafından önerilen yönteme [2] ilişkin kontrolör parametreleri Tablo 3.2 yardımıyla belirlenir .



Şekil 3.3 - Nyquist diyagramından sistemin kararlılık sınırının belirlenmesi.

Tablo 3.2 - Ziegler-Nichols sürekli titreşim yönteminden kontrolör parametrelerinin belirlenmesi.

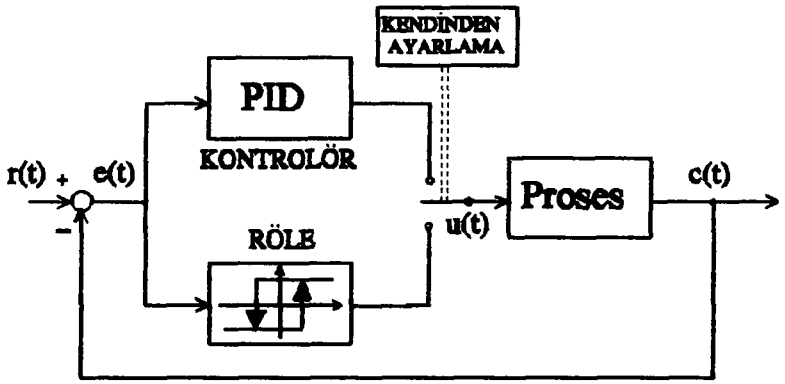
Kontrolör Tipi	Kp	Ki	Kd
P	0,5Kc	——	——
PI	0,4Kc	1,25/Tc	——
PID	0,6Kc	2/Tc	0,12Tc

Bu yöntemin olumsuz tarafı, K_p parametre değerinin artmasıyla her sistemin osilasyona sokulamaması veya oluşan osilasyonun genliğinin kontrol edilememesidir.

3.2.3. RÖLE YÖNTEMİ

Åstrøm ve Hägglund tarafından önerilen bu yöntem Ziegler-Nichols sürekli salınım yönteminin geliştirilmiş şeklidir [1], [2]. önceki yöntemde kapalı çevrimli sistemin belirli bir K_p kazancı için kararsız olduğu varsayılır. Röle yönteminde sürekli salınım herhangi bir

çalışma noktasında gerçeklenebilir. Röle yöntemine dayanan kendinden ayarlamalı PID kontrolörünün yapısı Şekil 3.4'de verilmiştir.



Şekil 3.4 - Röle yöntemine dayanan kendinden ayarlamalı PID kontrolörünün blok diyagramı.

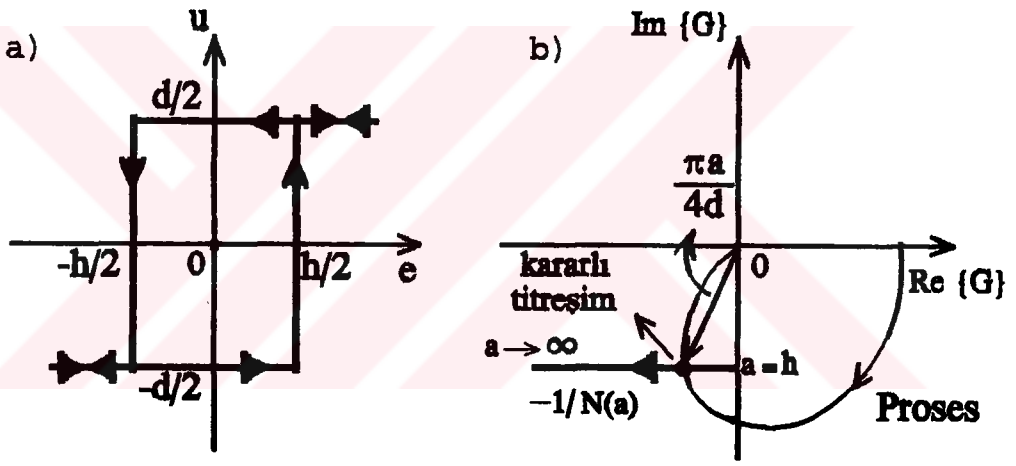
Burada histerezisli bir röle vasıtasıyla proses girişine, periyodu açık çevrim sisteminin kesim frekansına yakın bir işaret uygulanır. Bu işlemi gerçeklemek için proses girişindeki anahtardan yararlanılır. Doğrudan PID kontrolu devrede tutulabildiği gibi, kendinden ayarlama moduna da bu anahtar vasıtasıyla geçilir. Kendinden ayarlama modunda sistem bir osilasyona girer. Kararlı bir salınım elde edildiğinde osilasyonun T_c periyodu ve K_c kazancı belirlenir. Ziegler-Nichols tarafından Tablo 3.2'de verilen değerlerden yararlanarak PID kontrolörünün katsayıları elde edilir. Kontrolöre bu yeni parametreler aktarılarak normal çalışmaya geçilir.

Histerezli röle kullanılması çeşitli avantajlar sağlar. Sıradan rölelerde görülen gürültünün röle anahtarlamasına olan etkisi ve bunun sonucu olarak kontrol işaretinin zayıflaması olayı görülmez. Histerezisli rölenin giriş-çıkış karakteristiği Şekil 3.5a'da verilmiştir.

Histerezisli rölelerde açıklayıcı fonksiyon [8], [9], [1],

$$-\frac{1}{N(a)} = -\frac{\pi}{4d}\sqrt{a^2-h^2} - j\frac{\pi h}{4d} \qquad (3.1)$$

şeklinde bulunur. Burada d röle karakteristiğinin genliği, h histerezisin genişliğidir. Bu fonksiyonun karmaşık düzlemdeki karşılığı reel eksene paralel bir doğrudur (Bkz. Şekil 3.5b). Dolayısıyla prosese ilişkin Nyquist eğrisi ile bu doğrunun kesişim noktasında kararlı bir salınım elde edilecektir.



Şekil 3.5a - Histerezisli rölenin giriş-çıkış karakteristiği,

3.5b - Histerezisli röleye ait -1/N(a) açıklayıcı fonksiyonun G kompleks düzlemdeki değişimi ve kararlı salınım noktası.

Rölenin d genliği uygun seçilerek, sinüsoidal olduğu varsayılan $e(t) = a \sin \omega_c t$ için, a hata genliği ve ω_c salınım frekansı kapalı çevrimli sistemde oluşan salınımdan elde edilir. a hata genliği teorik olarak, hata işaretinin yarı periyoduna karşılık gelen A alanı hesaplanarak da belirlenebilir. Bu durumda hata genliği için

$$a = \frac{A\,\omega_c}{2} \qquad (3.2)$$

değeri bulunur. Sistemi salınımına sokan K_c kazancı ve salınım periyodu

$$K_c = \frac{4\,d}{\pi\,a} \qquad (3.3)$$

$$T_c = \frac{2\pi}{\omega_c} \qquad (3.4)$$

ifadelerinden elde edilir. Bu değerler Tablo 3.2'de yerine konarak kontrolör katsayıları,

$$K_p = 0,6\,K_c \;,\; K_i = 2\,/\,T_c \;,\; K_d = 0,12\;T_c \qquad (3.5)$$

şeklinde belirlenir.

BÖLÜM 4

MİKROKONTROLÖR TABANLI GERÇEKLEME

Bu bölümde, kendinden ayarlamalı PID kontrolörünün bir mikrokontrolör kartı ile bu çalışmada ne şekilde gerçeklendiği anlatılacaktır. İlk olarak PID kontrolörünün yapısı ele alınacak ve bu yapıya Åstrom ve Hägglund tarafından önerilen Röle yöntemi [1], [2] ile kendinden ayarlama işlevi ilave edilecektir. Beşinci bölümde bütün bu işlevleri yerine getirecek kartın yapısı ve uygulamada kullanılan sistem ayrıntılı bir şekilde açıklanacaktır.

4.1. DİJİTAL PID KONTROLÖRÜNÜN GERÇEKLENMESİ

Dijital PID algoritması, mikrokontrolör tabanlı ortamda assembly dilinde yazılmıştır. Mikrokontrolör kartı ile kontrol edilecek sistem arasındaki bağlantı yine aynı kart üzerindeki ADÇ-DAÇ ikilisiyle sağlanmaktadır.

Paralel yapıdaki PID kontrolörünün zaman tanım bölgesindeki ve z-tanım bölgesindeki ifadeleri Bölüm 2.3'de verilmişti. Adı geçen bu bölümde (2.26) denklemiyle verilen U(z)/E(z) ifadesi düzenlenir,

$$\frac{U(z)}{E(z)} = \frac{(2K_d+2K_pT+K_iT^2) + (K_iT^2-2K_pT-4K_d)\,z^{-1}+2K_dz^{-2}}{2T-2Tz^{-1}} \tag{4.1}$$

ve taraf tarafa çarpılırsa,

$$(2K_d+2K_pT+K_iT^2)\,E(z) + (K_iT^2-2K_pT-4K_d)\,z^{-1}E(z) + 2K_dz^{-2}E(z) = 2TU(z) - 2Tz^{-1}U(z) \tag{4.2}$$

fark denklemi elde edilir.

$E(z) = e_1$: kT. örneklemeye ilişkin hata işareti,

$z^{-1}E(z) = e_2$: (kT-1). -bir önceki- örneklemeye ilişkin hata işareti,

$z^{-2}E(z) = e_3$: (kT-2). -iki önceki- örneklemeye ilişkin hata işareti,

$U(z) = u_1$: kT. örneklemeye ilişkin kontrol işareti,

$z^{-1}U(z) = u_2$: (kT-1). örneklemeye ilişkin kontrol işareti

olmak üzere ve

$$P_1 = K_p + \frac{K_d}{T} + \frac{K_iT}{2} \tag{4.3}$$

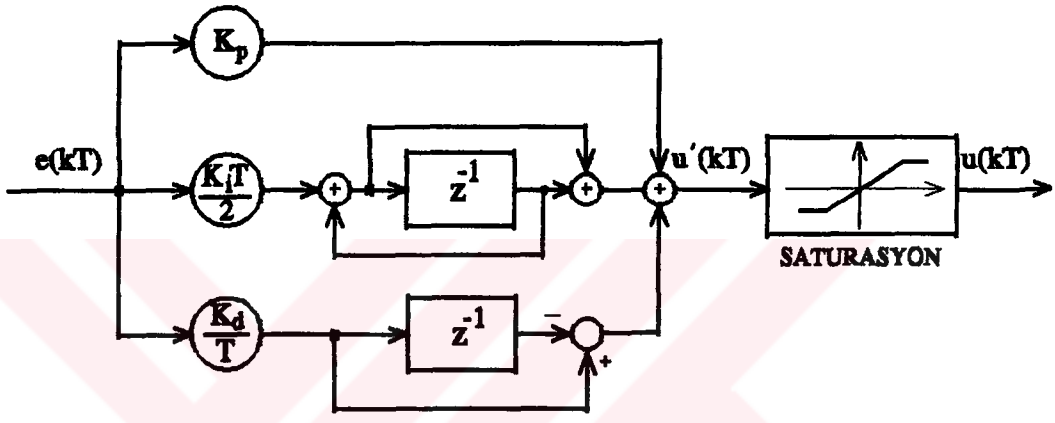
$$P_2 = \frac{K_iT}{2} - \frac{2K_d}{T} - K_p \tag{4.4}$$

$$P_3 = \frac{K_d}{T} \tag{4.5}$$

tanımlamaları ile, her T örnekleme anı için PID kontrolörü algoritmasının oluşturduğu kontrol işareti,

$$u_1 = P_1 e_1 + P_2 e_2 + P_3 e_3 + u_2 \qquad (4.6)$$

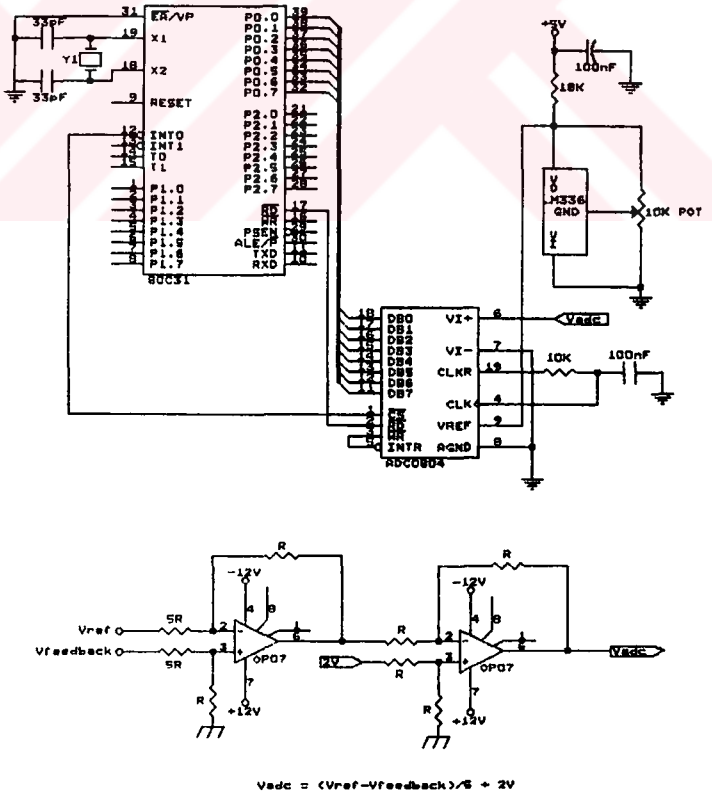
şeklinde elde edilir. Yaklaşık trapezoidal integrasyon ve türev kullanılarak assembly dilinde programlanan PID algoritması Şekil 4.1'deki gibi olacaktır. Bu yapıda görülen satürasyon bloğunun işlevi ileride anlatılacaktır.



Şekil 4.1 - Gerçeklenen PID kontrolörünün yapısı.

Gerçeklenen devrede kontrol edilen sisteme göre örnekleme periyodu 5ms ile 50s arasında seçilebilir. 5ms'lik minimum süre mikrokontrolör yapısında bulunan zamanlayıcı kesmesi (timer interrupt) ile sağlanmaktadır. Bu yapı her 250μs'de bir bir kesme işareti üretmektedir. Yirmi tane kesme işaretinden sonra PID altprogramına dallanılır. Eğer örnekleme zamanı 5ms'lik minimum süreye eşit olarak verilmişse, doğrudan PID altprogramı çalıştırılır. 5ms'den büyük bir örnekleme süresi verilmiş ise PID altprogramı başlangıcında T-5ms süresince bir bekleme yaratılır ve PID programı bu süre sonunda çalıştırılır. Tuş takımından verilen veya kendinden ayarlama ile elde edilen K_p , K_i , K_d parametre değerleri ile ADÇ'den okunan e(t)=r(t)-c(t) hata değeri değerlendirilir ve yeni kontrol işareti oluşturulur. Elde edilen kontrol işareti DAÇ vasıtasıyla sürekli işarete çevrilip sisteme uygulanır.

PID programının, dolayısıyla mikrokontrolör kartının, kontrol edilen sistemle bağlatısını ADÇ ve DAÇ tümdevreleri sağlamaktadır. ADÇ birimi 8 bitlik bir ADC0804 tümdevresi olup bu devre (Bkz. Şekil 4.2a) sürekli hata işaretini doğrudan bir dijital işarete çevirir. Hata işareti ±10V arasında değişen sürekli bir işarettir. Ancak, ADÇ beslemesi +5V ve referans gerilimi +2V olduğundan sadece 0-4V girişlere izin verilebilmektedir. Bu yüzden ±10V hata işareti aralığı, 0-4V aralığına çevrilir. Yapılan işlem giriş geriliminin 1/5 katını alıp, +2V ile toplamak şeklindedir. Dolayısıyla bu yapı Şekil 4.2b'de de gösterildiği gibi kaskad bağlı iki toplama devresine karşılık düşmektedir. ADÇ girişindeki ±10V arasındaki hata gerilimleri için sistem aktif olarak çalışır, bu sınırın dışındaki değerler dikkate alınmaz.

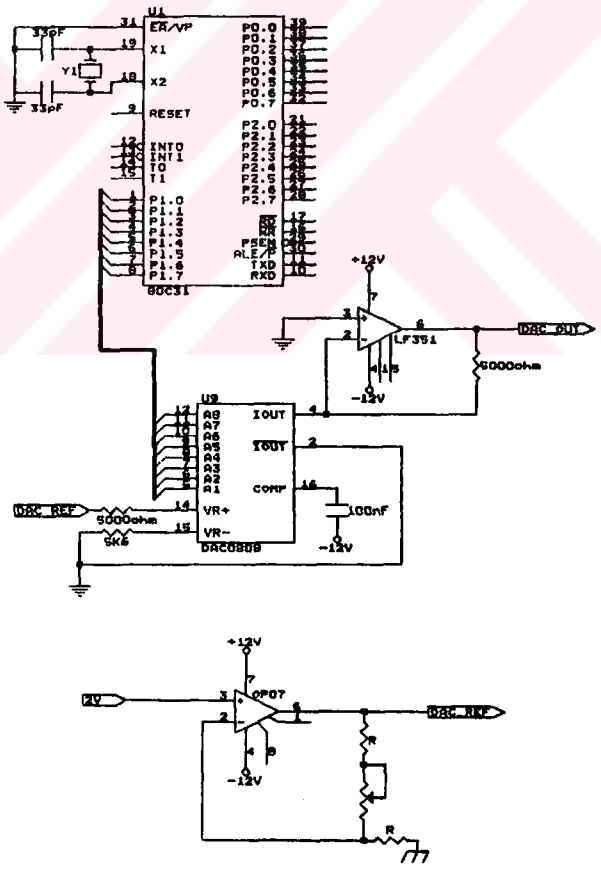


Vadc = (Vref-Vfeedback)/5 + 2V

Şekil 4.2a - ADC0804 tümdevresi yapısı ve bağlantıları,
4.2b - ±10V'luk hata işaretini, 0-4V'a dönüştüren devrenin şeması.

PID programıyla elde edilen dijital kontrol işaretinin kontrol edilen sisteme uygulanması için bu işaret sürekli bir işarete dönüştürülmelidir. Bu işlem 8 bitlik DAC0808 tümdevresi ile sağlanır. Bu tümdevrenin yapısı ve mikrokontrolör ile bağlantıları Şekil 4.3a'da verilmiştir.

Kontrol işaretinin değişim aralığı 0-10V'dur. Dolayısıyla Maksimum 10V'luk bir gerilim elde etmek için DAÇ elemanına 10V'luk bir referans gerilim uygulamak gerekmektedir. Bu gerilim ADÇ'nin 2V'luk referans geriliminin bir işlemsel kuvvetlendirici devre ile 10V'a yükseltilmesiyle elde edilir. Dönüştürücü devrenin yapısı Şekil 4.3b' de verilmiştir.



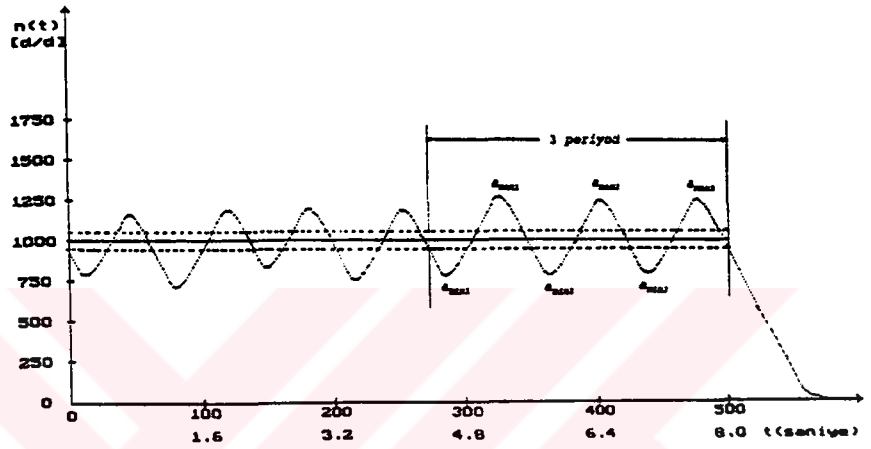
Şekil 4.3a - DAC0808 tümdevresi yapısı ve bağlantıları,
4.3b - 2V'luk ADÇ referans gerilimini, DAÇ için 10V'a yükselten dönüştürme devresi.

PID programının önemli bir özelliği Şekil 4.1'deki satürasyon bloğu ile ifade edilmiştir. Verilen parametre değerlerinden ve oluşan hatadan yararlanarak bulunan kontrol işareti, yukarıda da söz edildiği üzere DAÇ vasıtasıyla 0-10V aralığında sürekli işarete dönüştürülmektedir. Eğer PID programı sonucunda, sistemi kontrol etmek için 10V'dan daha yüksek bir gerilim gerektiği sonucuna varılırsa, DAÇ çıkışına 10V maksimum gerilim uygulanır. Negatif değerler gerektiğinde ise çıkışa minimum gerilim 0V kontrol işareti uygulanır. Bu işleme Şekil 4.1'de belirtildiği gibi satürasyon adı verilir.

4.2. KENDİNDEN AYARLAMALI PID KONTROLÖRÜ

Kendinden ayarlamalı PID algoritması da, assembly dilinde yazılmıştır. Burada, dijital PID kontrolörüne, Åstrom ve Hägglund tarafından önerilen kendinden ayarlama özellikli Röle yöntemi [1],[2] ilave edilmiştir. Bu işlemi gerçeklemek için, öncelikle referans hız sıfır değerine getirilir ve klavyeden kendinden ayarlama yapılacak hız değeri, örneğin 750 dev/dak, verilir. Verilen bu hız değerinden d-röle genliği bulunur. Bu uygulamada Röle'ye ilişkin histerezis genişliği sabit bir değer olarak deneysel yolla tespit edilmiştir. d röle genliği ise verilen referans hızın gerilim olarak değeri ile orantılı, yani değişken olarak kabul edilir ve DAÇ vasıtasıyla sisteme kontrol işareti olarak bu değer uygulanır. Sistem çıkışı belirlenen histerezisin üst sınırı olan C_{ust} gerilimini aşınca, kontrol işareti minimum gerilim olan 0 V'a yakın bir gerilim değerine çekilir. Sistem çıkışı, histerezisin alt sınırı C_{alt} gerilimine düşünce, kontrol işareti yine röle genliği seviyesine yükseltilir. Bu şekilde sistemin osilasyona girmesi sağlanır. Bu işlem belirli bir süre sürdürülür. Son üç periyoda girilince kontrolör içinde bulunan zamanlayıcı çalıştırılır ve genliğin a_{max} ve a_{min}

değerleri tespit edilir. Bu değerlerden a-osilasyonun genliği, $[(a_{max1}-a_{min1})+(a_{max2}-a_{min2})+(a_{max3}-a_{min3})]/3$ şeklinde belirlenir. Üçüncü periyodun sonunda zamanlayıcı durdurulur ve zamanlayıcıdan okunan değerin 3'e bölümünden T_c-osilasyonun periyodu elde edilir. Bu işlemler Şekil 4.4'te gösterilmiştir.



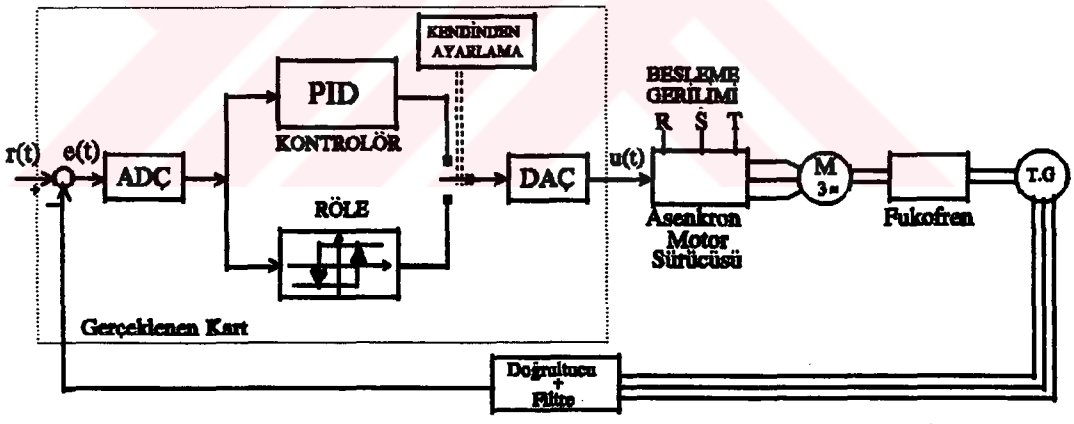
Şekil 4.4 - Salınıma ait genlik ve periyodun belirlenmesi.

a-osilasyonun genliği ve d-röle genliği değerlerinden kazanç $K_c = 4d/\pi a$ şeklinde belirlenir. Bu durumda PID kontrolör katsayıları $K_p = 0,6K_c$, $K_I = 2/T_c$, $K_d = 0,12T_c$ şeklinde bulunur ve kullanıcıya tavsiye edilir. Kullanıcıdan olur alındıktan sonra bu parametre değerleri PID algoritmasına aktarılır veya manuel olarak kullanıcı tarafından tanımlanması istenebilir.

BÖLÜM 5

KONTROL EDİLEN SİSTEM VE KONTROLÖRÜN ÖZELLİKLERİ

Bu çalışmada, kendinden ayarlamalı dijital PID kontrolörü, skaler kontrol yöntemi ile çalışan, doğru gerilim aradevreli frekans çevirici üzerinden beslenen, bilezikli asenkron motorun hız kontrolünde kullanılmıştır. Sisteme ilişkin blok diyagramı Şekil 5.1'de verilmiştir.



Şekil 5.1 - Asenkron motor hız kontrolünde kendinden ayarlamalı dijital PID kontrolörünün kullanılması.

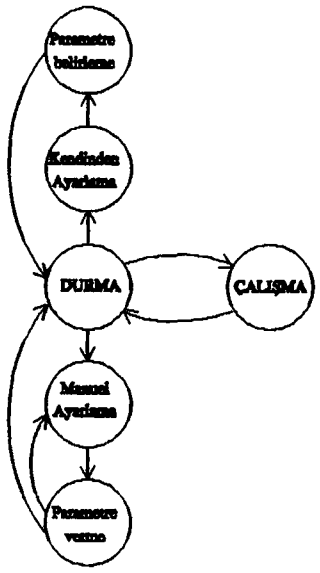
Bu yapıda, bilezikli asenkron motor ve sürücüsü, kontrol edilen sistem olarak kabul edilmektedir. PID algoritmasının referans değerle geribesleme işaretini değerlendirerek elde ettiği 0-10V arasındaki sürekli kontrol

işareti asenkron motor sürücüsüne uygulanır. Sürücü bu kontrol işaretine bağlı olarak, frekansı 0,5-60 Hz ve genliği 50-380 V arasında değişen, darbe genişlik modülasyonlu bir gerilim değeri üretir ve bu gerilimi asenkron motorun statoruna uygular. Asenkron motorun yüklenmesi motorun miline bağlı Fukofreni ile sağlanır. Hıza ilişkin geribesleme bilgisi motor miline bağlı takogeneratör üzerinden üç fazlı alternatif gerilim olarak elde edilir. Bu gerilim hızla doğru orantılıdır ve üç fazlı tamdalga doğrultucu ile doğrultulup, filtrelenerek 0-10V arasında değişen sürekli işarete çevrilir. Bu gerilim geribesleme işareti olarak PID algoritması tarafından tekrar değerlendirilir.

Yazılım sistemin çalışmasını çeşitli modlara böler. Bu modlar;

1-Durma modu,
2-Manuel ayarlama modu,
3-Kendinden ayarlama modu,
4-Çalışma modu

şeklindedir. Bu modlara ilişkin durum diyagramı Şekil 5.2'de verilmiştir.



Şekil 5.2 - Çalışma modlarının durum diyagramı.

Sistem ilk açıldığında durma modundadır. Bu modda hiç bir işlem yapılmaz, sadece tuş takımı taranarak diğer modlara geçme isteği olup olmadığı araştırılır.

Manuel ayarlama modunda, kullanıcıdan sırasıyla K_p-oran katsayısı, K_i-integral katsayısı, K_d-türev katsayısı ve T-örnekleme zamanının klavyeden verilmesi istenir.

Kendinden ayarlama modunda referans gerilim ayarlama yapılacak gerilim değerine ayarlanarak sistem çıkışının yorumlanması sağlanır. Sistem osilasyona sokularak oluşan salınımın genliği ve periyodu dolayısıyla K_p, K_i ve K_d katsayıları belirlenir.

Çalışma modunda ise, manuel olarak verilen ya da kendinden ayarlama sonucunda belirlenen katsayı değerleri kullanılarak sistem çalıştırılır. Çalışma modundan durma moduna geçiş, tuş takımındaki bir tuşa basılarak gerçeklenir.

5.1. MİKROKONTROLÖR TABANLI KARTIN YAPISI VE ÖZELLİKLERİ

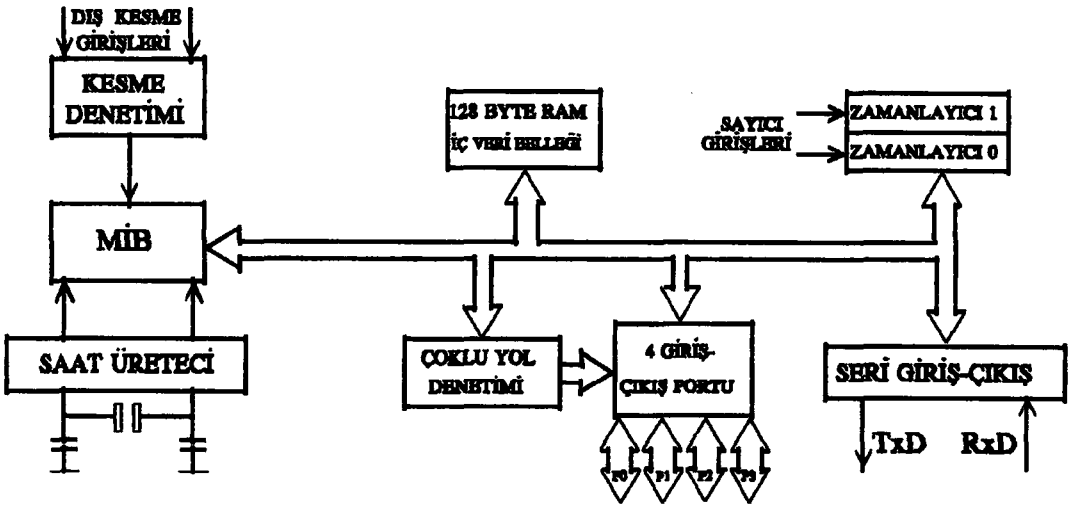
Gerçeklenen kartın yapısı üç bölümde incelenecektir. İlk bölümde temel bileşenler, mikrokontrolör ve bellek birimleri, ikinci bölümde, tuş takımı ve gösterge elemanları, üçüncü bölümde ise besleme katları ele alınacaktır. Kontrol edilen sistemle iletişimi sağlayan ADÇ ve DAÇ katları daha önce Bölüm 4.1'de incelenmişti.

5.1.1. MİKROKONTROLÖR VE BELLEK BİRİMLERİ

Gerçeklenen kartta CHMOS teknolojisi ile Intel tarafından üretilen, 8 bitlik 80C31 mikrokontrolörü kullanılmıştır. CHMOS teknolojisiyle üretilen 80C31 mikrokontrolörü, HMOS teknolojisiyle üretilen 8031 mikrokontrolörüne oranla çok az akım (20 mA mertebesinde) çeker ve daha yüksek çalışma frekanslarında (12-16 MHz) çalıştırılabilirler [10].

80C31 mikrokontrolörü aşağıdaki özelliklere sahiptir;

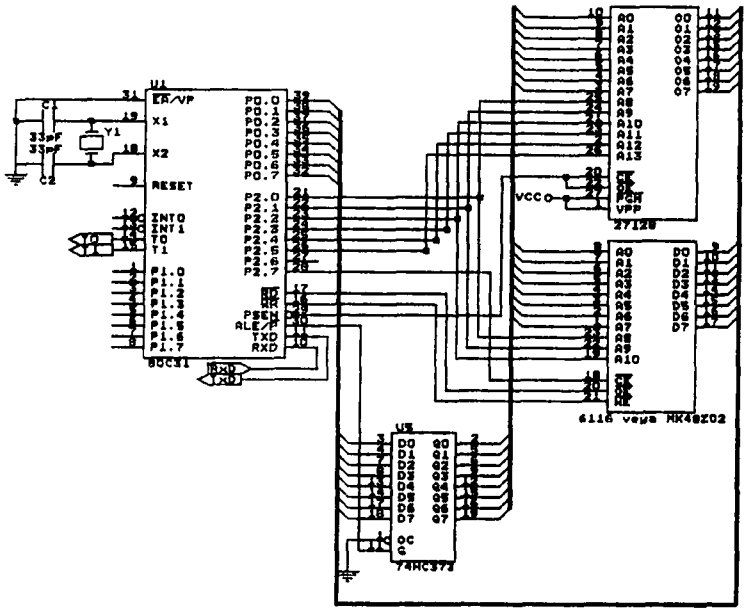
128 Byte'lık bir iç belleği bulunur,
8 bitlik 4 adet giriş-çıkış portuna sahiptir,
iki adet 16 bit'lik zamanlayıcısı bulunur,
iki öncelik düzeyi olan beş kesme kaynağına sahiptir,
programlanabilir seri giriş-çıkış kapısı (TxD-"Veri gönder" ve RxD-"Veri al") vardır,
tümdevre üzerinde osilatör ve saat işareti oluşturma katı bulunur [10]. Bu özellikleri nedeniyle kullanımı ve programlaması kolaydır. Mikrokontrolörün sahip olduğu bu özellikleri sergileyen yapısı Şekil 5.3'de verilmiştir.



Şekil 5.3 - 80C31 mikrokontrolörünün özellikleri.

Mikrokontrolörde, 128 Byte'lık iç veri belleği özel işlev kaydedicileriyle program içindeki değişkenlere ayrılmıştır. 80C31'de tümdevre içinde bulunmayan program belleği, 64 Kbyte'a kadar, her iki bellekten farklı olan dış veri belleği de yine 64 Kbyte'a kadar seçilebilmektedir. Mikrokontrolörde, her biri 8 bit olan 4 giriş portu P0,P1,P2,P3 olarak adlandırılır. Uygulamada kullanılan mikrokontrolörün P0 portu, 74HC373 D-tipi tutucu (D-type latch) tümdevresiyle hem veri yolunu hem de adres yolunun düşük anlamlı 8 bitine ayrılmıştır. P2 portu adres yolunun yüksek anlamlı 8 biti olarak kullanılır. P1 portundan DAC0808 tümdevresine kontrol işaretinin dijital olarak değeri gönderilir. P3 portunda ise her port ucunun ayrı bir anlamı bulunur. P3.0 (RxD) ve P3.1 (TxD) tuş takımını okuma işlemlerine ayrılmıştır. P3.2 (INT0-dış kesme 0), ADÇ çalışırken göstergenin etkilenmemesi için anahtar görevi görmektedir. P3.3 ve P3.4 (T0 ve T1 sayıcı girişleri) tuş takımı işlemlerinde kullanılmaktadır. P3.5 (WR-yazma) dış veri belleğine (RAM) yazma yapılmasını veya göstergenin seçilmesini (RD ve INT1 ile birlikte) sağlar. P3.6 (RD-okuma), dış veri belleği ve ADC0804'ten okuma yapılmasında veya göstergenin seçilmesinde (WR ve INT1 ile birlikte) kullanılır.

80C31 mikrokontrolörü üzerinde, 128 Byte'lık iç veri belleği yer almasına rağmen, yazılan programın depolanacağı bir bellek alanı bulunmamakta, dolayısıyla bir dış program belleğine ihtiyaç duyulmaktadır. Bu yüzden gerçeklenen kart üzerinde bütün programı içeren 16 Kbyte'lık 27128 Salt Oku bellek elemanı (EPROM) kullanılmıştır. Ayrıca elektrik kesintilerinde, kendinden ayarlama ile belirlenen kontrolör parametrelerini saklamaya yarayan, kendinden pilli 2 Kbyte'lık MK48Z02 Yaz Oku bellek elemanı (RAM) opsiyonel olarak kullanıma hazır hale getirilmiştir. Kullanılan bellek elemanlarının mikrokontrolör ile bağlantıları Şekil 5.4'te verilmiştir.



Şekil 5.4 - Mikrokontrolör ile 27128 EPROM ve MK48Z02 (veya 6116) RAM bağlantıları.

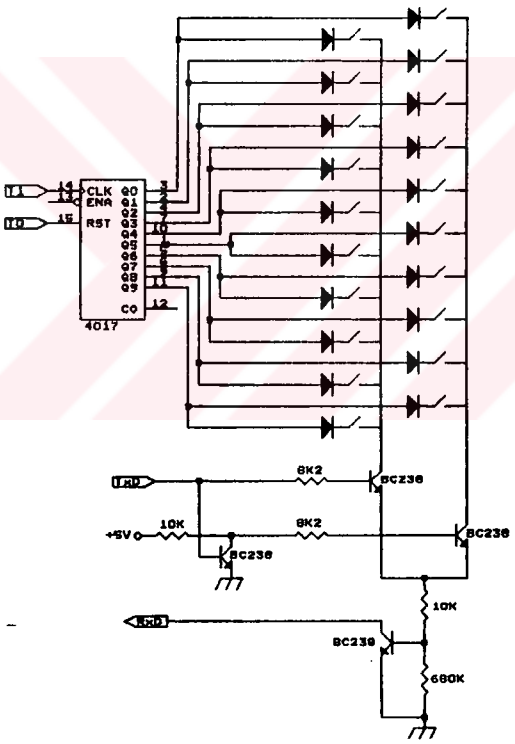
Dış program belleğinden sadece okuma işlemi gerçeklenmektedir. P2 portundan sağlanan adres yolunun yüksek anlamlı 8 biti, doğrudan dış belleğin yüksek anlamlı 8 bitine bağlıdır. P0 portundan, önce adres yolunun düşük anlamlı 8 biti iletilir. Mikrokontrolörün ALE çıkışının düşen kenarı ile 74HC373 tutucu tümdevresinin çıkışında P0 portundan iletilen adres yolunun düşük anlamlı 8 bitinin tutulması sağlanır. Ayrıca, ALE çıkışının yükselen kenarı ile tutucu tümdevresi devre dışı bırakılarak 8 bitlik veri yolunun kullanılması sağlanır.

Dış program belleğinin çıkış izni, mikrokontrolörde $\overline{PSEN}$ ucu ile sağlanmakta ve 12 MHz'de 1μs'lik süre içinde dış program belleğine iki kez erişmek mümkün olmaktadır. Bu şekilde, mikrokontrolörde $\overline{PSEN}$ ucu aktif yapılarak, gerçeklenen yazılım programının yürütülmesi sağlanır.

5.1.2. TUŞ TAKIMI VE GÖSTERGE ELEMANLARI

Mikrokontrolörün, kontrol edilen sistemle iletişimini sağlayan ADÇ ve DAÇ elemanları Bölüm 4.1'de incelenmişti. Burada, mikrokontrolörün kullanıcı ile iletişimini sağlayan tuş takımı ve LCD-gösterge elemanları tanıtılacaktır.

Kullanılan tuş takımı, 4x5 matris düzeninde dizilmiş dokunmatik anahtarlardır. Şekil 5.5a'da tuş takımını oluşturan elemanlar, Şekil 5.5b'de ise kullanılan tuşların anlamları verilmiştir.



1	2	3	
4	5	6	M
7	8	9	A
»	0	«	
	S	R	E

Şekil 5.5a - Tuş takımı oluşturan elemanlar,
5.5b - Kullanılan tuşların anlamları.

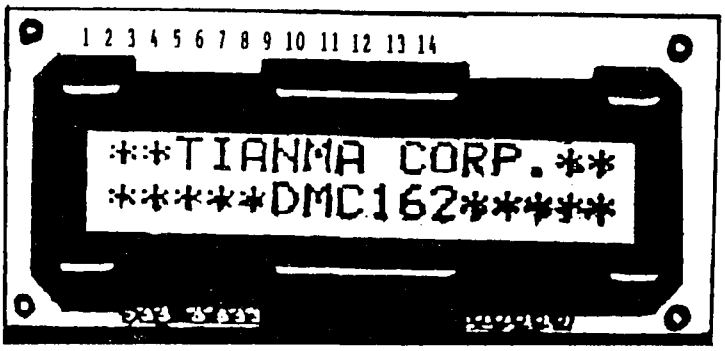
Mikrokontrolörün T1 çıkışı, 4017 BCD sayıcı entegresinin saat girişine, T0 çıkışı reset girişine bağlanmıştır. T1 çıkışı ile BCD sayıcı birer artarak saymaya başlar. Bu sayma sırasında, 10x2 matris şeklinde düşünülen

tuşlar satır satır aktif hale getirilir. BCD sayıcı 10'a kadar sayınca, TO ile sayıcının tekrar başa dönmesi sağlanır. Sayma sırasında, mikrokontrolörün TxD çıkışıyla, BC238 tranzistorları anahtarlanarak, sütunların aktif olanı tesbit edilir. Dolayısıyla sayıcıdaki değer ile hangi sütunun aktif olduğu göz önünde bulundurularak basılan tuş belirlenir ve bu tuşa yazılım ile bir kod karşı düşürülür. Gerçeklenen tuş takımında 0...9 arasında numaratör tuşları bulunduğu gibi, parametrelerin girilmesi sırasında imleci (cursor) sağa ve sola kaydırmak için iki tuş daha tanımlanmıştır. Ayrıca, manuel moda geçişte M tuşu, kendinden ayarlama moduna geçişte A tuşu, çalışma moduna geçişte R tuşu, durma moduna geçişte S tuşu ve kullanıcıdan onay almak için E tuşu tanımlanmıştır.

Devrede kullanılan LCD-gösterge TIANMA firmasının 16x2 karakterlik DMC 162 modelidir (Bkz. Şekil 5.6). Göstergenin devre ile bağlantısı devre üzerindeki soketlerle sağlanmaktadır.

Göstergenin, gerçeklenen karta bağlantı uçları şu şekildedir [11]:

Pin ve anlamı	Mikrokontrolör kartındaki karşılığı
1 - Vss	Toprak bağlantısı
2 - Vdd	+5 V referans gerilimi
3 - VO	Kontrast ayarı
4 - RS	AO-Adres yolunun en yüksek anlamlı biti
5 - R/W	A1-Adres yolunun yüksek anlamlı 2.biti
6 - E	İzin girişi
7..14-DBO.DB7	Veri yolu girişleri



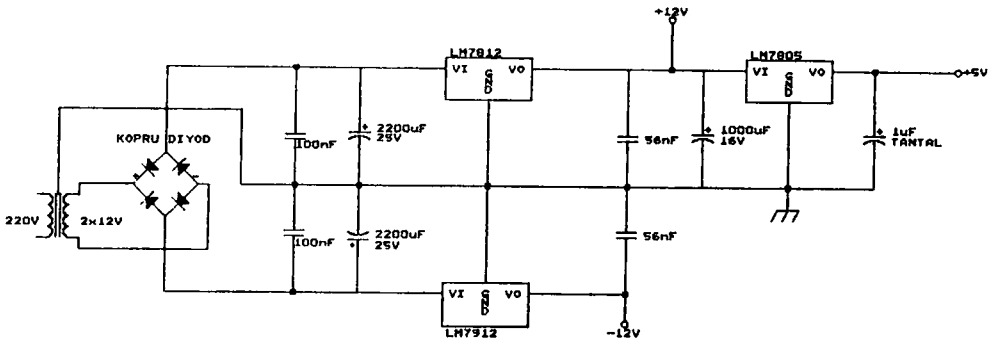
Şekil 5.6 - Gerçeklenen devrede kullanılan gösterge.

Göstergenin her bir gözünde 5x7 matris biçiminde noktalar bulunmakta, bu noktaların istenilenleri aktif yapılarak o göze karakter yazılması sağlanmaktadır. Mevcut 32 göze, o göze ilişkin adres ve karakter gönderilerek yazma işlemi gerçeklenir. Bu işlem öncesinde, göstergenin kullanıma hazırlanması gerekmektedir. İşlemin nasıl yapıldığı EK-A'da verilen assembly programından takip edilebilir.

5.1.3. BESLEME KATLARI

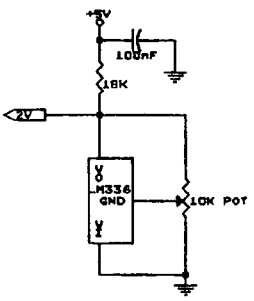
Gerçeklenen kart üzerinde gerekli referans gerilimleri oluşturmak amacıyla gerilim regülatörleri bulunmaktadır. +12 VDC gerilim elde etmek için LM7812, -12 VDC gerilim elde etmek için LM7912, +5 VDC elde etmek için LM7905 gerilim regülatörleri kullanılmıştır.

Besleme katı Şekil 5.7'de gösterilmiştir.



Şekil 5.7 - Gerçeklenen kartta yer alan besleme katı.

Voltmetre ile yapılan ölçümlerde +12 VDC için +12,05 VDC, -12 VDC için -11,97 VDC, +5 VDC için +5,05 VDC gerilim değerleri elde edilmiştir. Bu değerler, devre üzerindeki elemanların ideal besleme gerilimi değerlerine uygundur. ADÇ referans gerilimini (2 VDC) elde etmek için yine kart üzerinde Şekil 5.8'de görülen LM336 (2,5 V) gerilim regülatörü bulunmakta, ayrıca 2 VDC'lik bu gerilim DAÇ referans gerilimi için +10 VDC'ye kuvvetlendirilmektedir (Bkz. Bölüm 4.1, Şekil 4.3b).



Şekil 5.8 - 2 VDC referans geriliminin elde edilmesi.

5.2. KONTROL EDİLEN SİSTEM OLAN ASENKRON MOTOR VE ASENKRON MOTOR SÜRÜCÜSÜNÜN TEKNİK ÖZELLİKLERİ

SIEMENS firması tarafından üretilmiş bilezikli asenkron motorun plaka değerleri şu şekildedir [12];

Stator devresi:

		YILDIZ BAĞLI	ÜÇGEN BAĞLI
Anma gerilimi.....	U_N:	380V	220V
Stator faz akımı anma değeri...	I_N:	9,9A	17,1A
Asenkron motorun anma frekansı.	f_N:	50Hz	
Mil gücü anma değeri.....	P_N:	4kW	
Mil momenti anma değeri.....	M_N:	27,1Nm	
Hızın anma değeri.....	n_N:	1410dev/dak	

Rotor devresi:(Rotor sargıları Yıldız bağlıdır.)

Rotor anma gerilimi.....	U_{N_r}:	130V
Rotor anma akımı.....	I_{N_r}:	21A
Rotor uçları açıkken endüklenen gerilim.....	E_{r0}:	130V

Boşta çalışma, kısa devre, yavaşlatma deneyleri ve doğrudan RLC-metre ile ölçülerek elde edilen asenkron motorun parametreleri şu değere sahiptir [12]:

Stator direnci.....R_s: 1,07 Ω

Stator özendüktansı.....L_s: 54,18mH

Rotor direnci.....R_r: 0,52 Ω

Rotor özendüktansı.....L_r: 10,73mH

Statora indirgenmiş
rotor kaçak özendüktif reaktansı..$X_{r\sigma}$: 2,406 Ω

Stator kaçak özendüktif reaktansı..$X_{s\sigma}$: 2,406 Ω

Demir kayıp direnci.....R_{Fe}:322,42 Ω

Faydalı özendüktif reaktansı.....X_M: 39,57 Ω

Stator özendüktif reaktansı.....X_s: 17 Ω

Rotor özendüktif reaktansı.....X_r: 3,371 Ω

Zaman sabiti.....τ: 12,5s

Stator frekansının anma frekansına
oranı f_s/f_n.....β: 0,00646 Nms

Eylemsizlik momenti.....J: 0,0872 kgm²

Asenkron motor sürücüsü Asea Brown Boveri-ABB firmasının SAMI ministar 12 MB4 modelidir ve plaka değerleri şöyledir [12]:

		Lineer yük	Karesel yük
Anma gerilimi.....	U_N:	380V	380V
Maksimum anma gerilimi.....	U_{Nmax}:	415V	415V
Anma akımı.....	I_N:	17A	24A
Anma gücü.....	P_N:	7,5kW	11kW

Giriş gerilimi frekansı.....: 48-63 Hz

Güç faktörü.....: ≈1

Çıkış gerilimi genliği.....: Üç fazlı 0V-U_N mertebesinde

Çıkış gerilimi frekansı.....: 0,5-120Hz

Frekans çevirici tipi.....: Sabit gerilim aradevreli

Eviricinin çıkış geriliminin frekansı ve genliğinin kontrolü darbe genişlik modülasyonuyla skaler kontrol yöntemine göre yapılmaktadır. Asenkron makina sürücüsüne uygulanan kontrol işareti, sürücü içinde bulunan SIEMENS SAB 80635 N mikroişlemcisi tarafından değerlendirilerek, bu işarete karşılık gelen darbe genişlik modülasyonlu işaret belirlenir. Elde edilen işaret, sürücü elemanlar üzerinden, yaklaşık 2,5kHz'lik anahtarlama frekansına sahip, eviricinin anahtarlama tranzistorlarına gönderilir. Böylece kontrol işaretine darbe genişlik modülasyonlu bir gerilim ve frekans değeri karşı düşürülmüş olur.

Sürücü içinde bulunan mikroişlemci vasıtasıyla, cihazın minimum ve maksimum frekansı, verilen hız değerine ulaşma süresi (acceleration time) ve yavaşlama süresi (deceleration time), aşırı akım koruma sınırı, sabit hız değeri ile çok sayıda koruma ve frenlemeye yönelik işlevler, dijital olarak programlanabilmektedir. Cihazın ön yüzünde bulunan dijital gösterge sistemi ile çalışma anında motora uygulanan gerilimin frekansı, referans frekanslar ve bu frekanslar arası fark, % cinsinden motor akımı, momenti, aktif gücü ve aradevre gerilimi ile soğutucu sıcaklığı doğrudan izlenebilir [12].

Kontrol edilen sisteme ek olarak kullanılan dinamofren ve takogeneratör elemanları ise şu özelliklere sahiptirler:

Asenkron motoru yüklemek için kullanılan dinamofren elemanının özellikleri [12]:

Besleme gerilimi.....:	220 V DC	
Çekilen maksimum akım..:	4,5 A	
Uyguladığı aktif yük...:	1500 devir/dakika'da	6 kW
	3000 devir/dakika'da	7,5 kW

Hız geribeslemesini sağlamak için kullanılan takogeneratör 1000 devir/dakika'lık anma hızında, 6 VA anma yükünde, 30V fazarası gerilim üreten bir sistemdir.

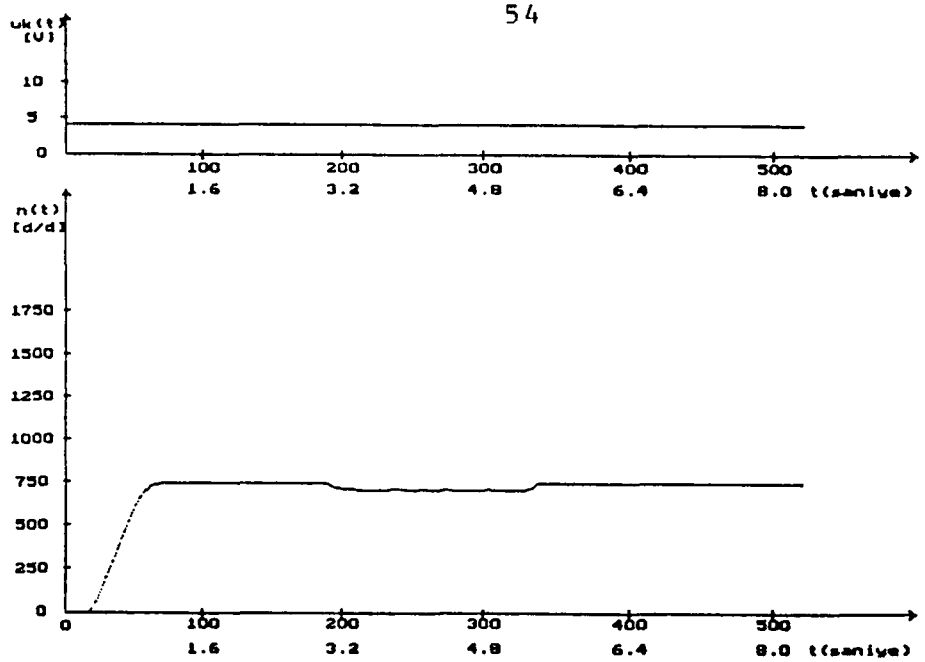
BÖLÜM 6

SONUÇLAR , YORUMLAR VE ÖNERİLER

Bu bölümün amacı, gerçeklenen mikrokontrolör tabanlı kendinden ayarlamalı PID kontrolörünü, kontrol edilmesi zor olan bir sisteme uygulayarak sonuçları yorumlamaktır. Çalışmada, kontrol edilen sistem olarak, asenkron motor ve sürücüsü ele alınmış ve hız kontrolu gerçeklenmeye çalışılmıştır. Ele alınan bu sistem, lineer olmayan bir sistemdir. Bu yüzden, sistemi transfer fonksiyonu cinsinden ifade etmek imkansızdır. Ancak, asenkron motor sürücüsünün bir kutuplu, asenkron motorunun iki kutuplu lineer bir sistem olarak modellenebileceği düşünülürse sistemin transfer fonksiyonu yaklaşık olarak

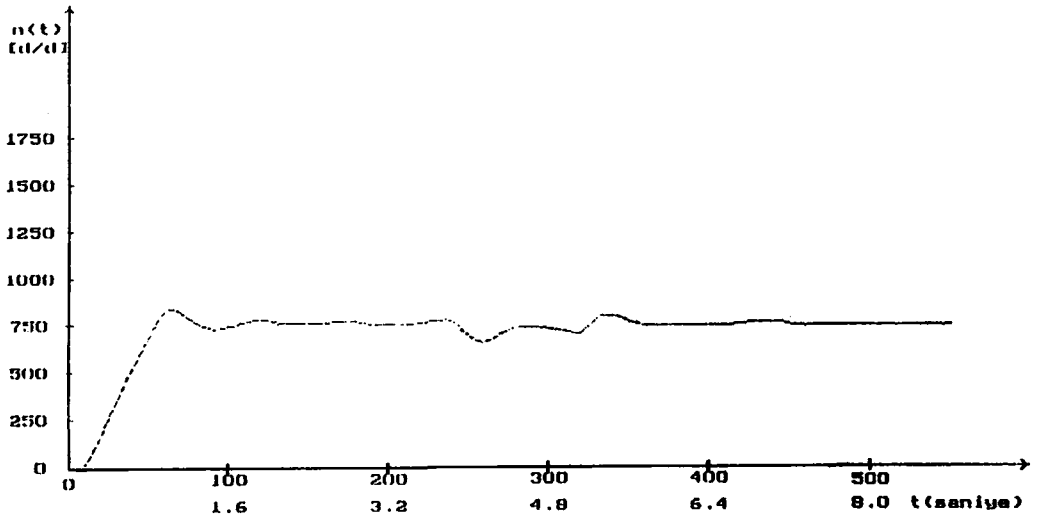
$$T(s) = \frac{K\omega_n^2}{(\tau_1 s+1)\ (\tau_2 s+1)\ (\tau_3 s+1)} \qquad (6.1)$$

şeklinde ifade edilebilir. Bu şekilde yaklaşık modellenmiş olan sisteme birim basamak giriş uygulandığında, sistemin cevabı modele uygun olarak Şekil 6.1'de gösterildiği gibi olur.



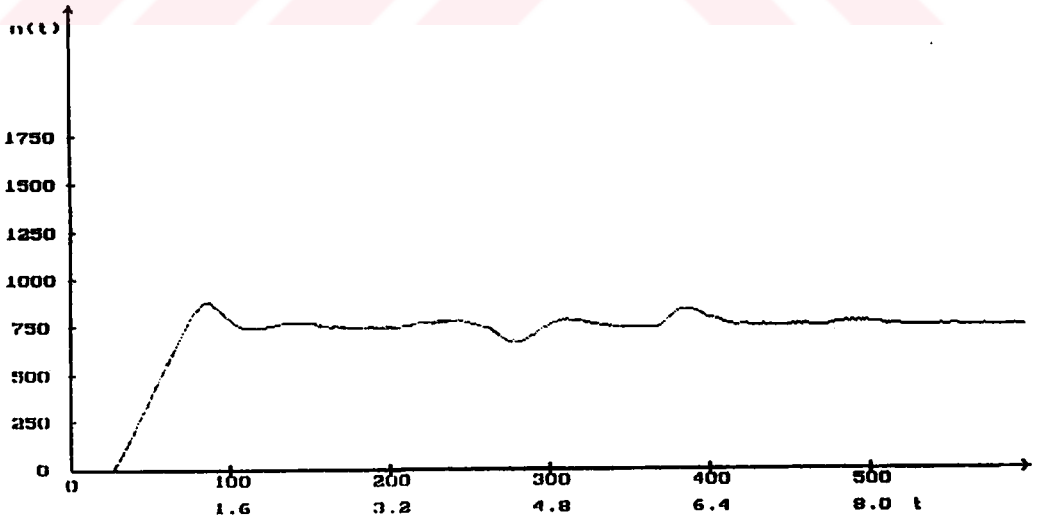
Şekil 6.1 - Açık çevrimli sistemin referans hızı 750 dev/dak için cevabı, ani yüklenme ve yükten çıkılması durumunda sisteme ilişkin hız eğrisi ve kontrol işareti.

Açık çevrimli sistemin cevap eğrisinde iki özellik göze çarpmaktadır. İlk önemli nokta, sistemin hiç salınım yapmadan istenilen referans hızına erişmesidir. Bu iyi bir davranıştır. Ancak kötü olan nokta, sistem yüklendiğinde ortaya çıkmaktadır. Yüklü durumda referans hız değişmediği halde, motor referans hıza çıkamamakta, hız hataları görülmektedir. Yüklü durumdan çıkıldığında, motor tekrar referans hıza erişmektedir. Bu açık çevrimli kontrolun kaçınılmaz bir sonucudur. Gerçeklenen mikrokontrolör tabanlı PID kontrolörü devreye sokularak, sistem davranışı üzerindeki etkileri incelenebilir. Kontrolör parametreleri $K_p = 2$, $K_i = 0$, $K_d = 0$ ve örnekleme zamanı $T = 16\,ms$ alındığında, ani yüklenme ve yükten çıkma durumunda Şekil 6.2'de gösterilen sistem cevabı elde edilir. Burada referans hız 770 dev/dak verildiğinde motor 750 dev/dak hızında çalışmaktadır. $K_i = 0$ olması sebebiyle kararlı hal hataları mevcuttur. Bir olumlu nokta ise motorun ani yüklenme ve yükten çıkılması durumunda hızın çok az etkilenmesidir.



Şekil 6.2 - $K_p = 2$, $K_i = 0$, $K_d = 0$, $T = 16\,ms$ için ani yüklenme ve yükten çıkma durumunda sistem çıkışına ilişkin hız eğrisi.

Şekil 6.3'te ise kontrolör parametreleri $K_p = 2$, $K_i = 2$, $K_d = 0$, $T = 16\,ms$ ve referans hızı 750 dev/dak için sistem çıkışına ilişkin hız grafiği verilmiştir.



Şekil 6.3 - $K_p = 2$, $K_i = 2$, $K_d = 0$ için ani yüklenme ve yükten çıkma durumunda sistem çıkışına ilişkin hız eğrisi.

Bu durumda açıkça görüldüğü gibi kararlı hal hız hataları giderilmiştir. Ancak hız eğrisinde aşımın biraz yüksek olduğu görülür. Motor ani yüklenme durumundan az da olsa etkilenir, yükten çıkışta ise tekrar referans hızına ulaşır.

Bütün bu verilerden sistem davranışını belirleyen büyüklükleri elde etmek mümkün olabilmektedir. Kontrolör sisteme uygulandıktan sonra, sistem kapalı çevrim transfer fonksiyonunun

$$T(s) = \frac{C(s)}{R(s)} = \frac{K\omega_n^2}{(\tau_1 s+1)\ (s^2+2\xi\omega_n s+\omega_n^2)} \qquad (6.2)$$

şeklinde olduğu varsayılabilir. Bu durumda sistem davranışını belirleyen büyüklükler;

$$\%A = \frac{C_p - C_\infty}{C_\infty} = \frac{780-750}{750} = 4$$

$$e_{ss} = 0$$

$$t_r = 0,7\ s$$

$$\xi \cong 0,6$$

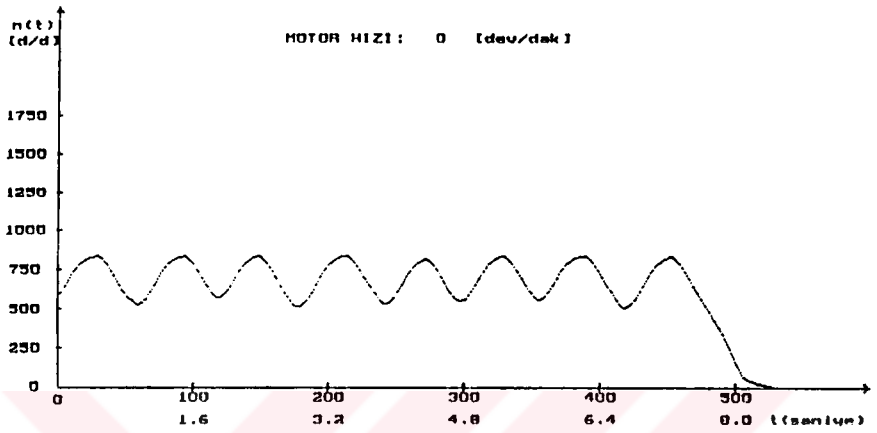
$$\omega_n \cong 7,5$$

şeklinde belirlenir ve sistemin transfer fonksiyonu

$$T(s) = \frac{56,25\,K}{(.6\,s+1)\ (s^2+9\,s+56,25)} = \frac{K}{0,01\,s^3+0,114\,s^2+0,76\,s+1} \qquad (6.3)$$

olduğu bulunur.

Kendinden ayarlama işleminin sisteme uygulanması için öncelikle referans, ayarlama yapılacak hız değerine getirilir. Bölüm 4.2'de açıklandığı gibi histerezisli röle karakteristiği ile sistemin osilasyona girmesi sağlanır. Bu salınım Şekil 6.4'de verilmiştir.



Şekil 6.4 - Yüksüz durumda, 750 dev/dak referans hızında kendinden ayarlama için oluşturulan osilasyon.

Bu işlem sonucunda salınımın genliğinden K_c-kazanç değeri ve T_c-salınımın periyodu belirlenir. Burada röle genliği $d = 4,467$ olarak alınmış ve salınımın genliği $a = 1,655$ şeklinde bulunmuştur. Bu durumda K_c-kazanç değeri için,

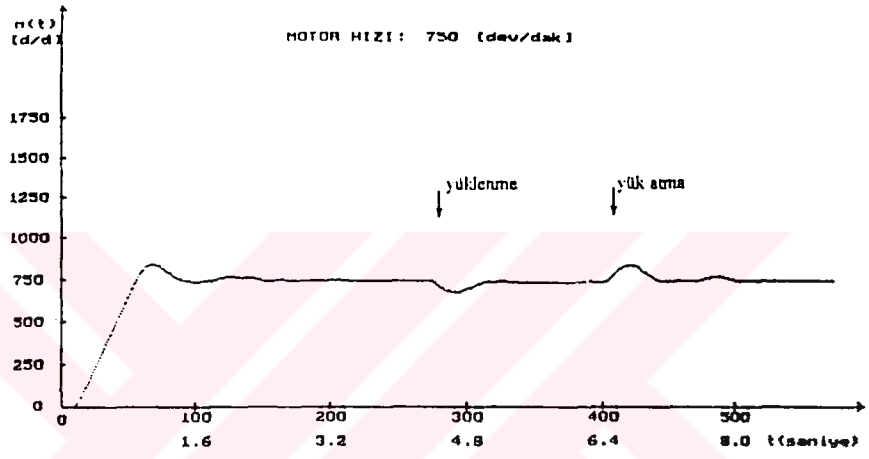
$$K_c = \frac{4d}{\pi a} = \frac{4 \,.\, 4,467}{\pi \,.\, 1,655} = 3,438$$

elde edilir. T_c-salınımın periyodu ise mikrokontrolörde bulunan bir zamanlayıcının aktif hale getirilmesiyle,

$$T_c = 0,952\ s$$

olarak bulunur. Son olarak $K_p = 0,6\,K_c$, $K_i = 2/T_c$, $K_d = 0,12\,T_c$

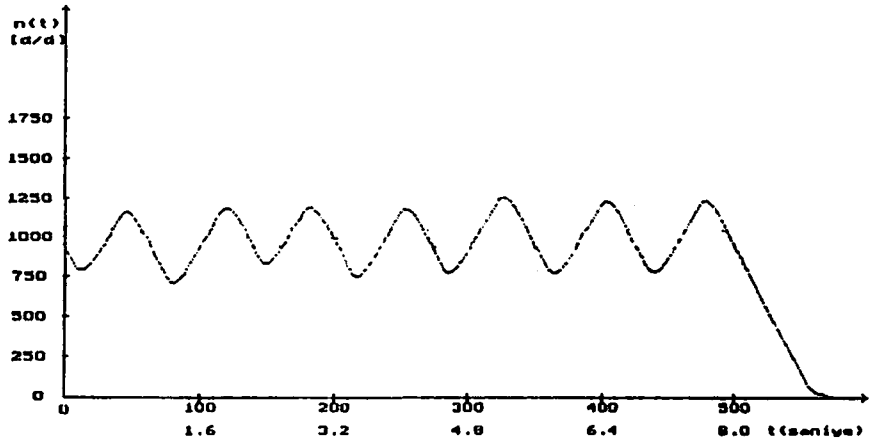
yaklaşımı ile 750 dev/dak referans hızı için yeni kontrolör katsayıları $K_p = 0,6.3,438 = 2,063$, $K_I = 2/0,952 = 2,101$ ve $K_d = 0,12.0,952 = 0,114$ şeklinde bulunmuştur. Örnekleme süresi T=16 ms alınarak kontrolöre bu yeni değerler aktarılır ve sistem çıkışı gözlemlenirse Şekil 6.5'de verilen hız eğrisi ile karşılaşılır. Bu eğri incelendiğinde aşımın azaldığı ve istenilen referans hızına erişildiği görülür.



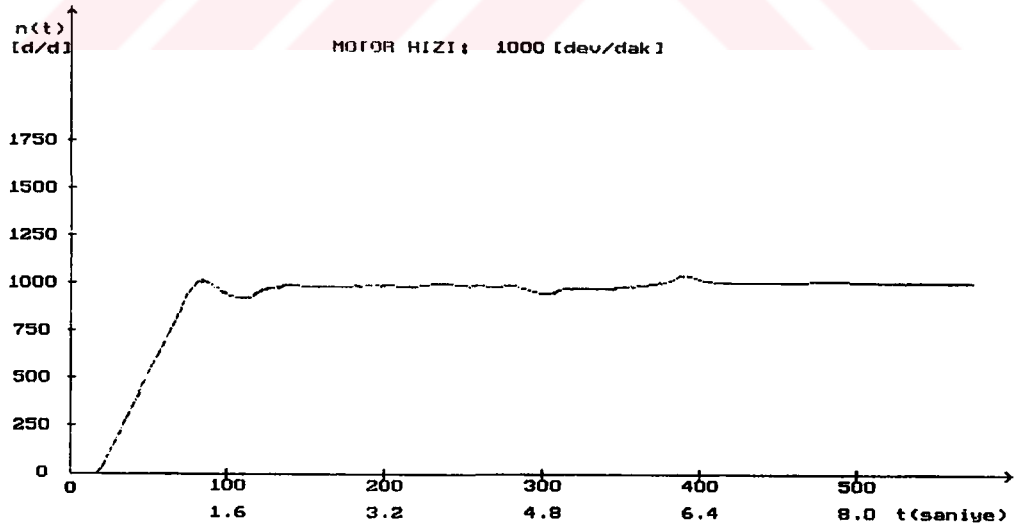
Şekil 6.5 - Yüksüz durumda 750 dev/dak referans hızı için kendinden ayarlama sonucunda sistemin cevabı.

Kendinden ayarlama işlemi 1000 dev/dak referans hızı için tekrarlanırsa oluşan osilasyon Şekil 6.6'da ve elde edilen yeni parametrelerin kontrolöre aktarılarak belirlenen sistem cevabı Şekil 6.7'de görülür.

Bulunan kontrolör parametreleri $K_p = 2,491$, $K_I = 1,790$ $K_d = 0,134$ şeklindedir. Sistemde aşım aynı mertebede kaldığı, ani yüklenme ve yüklenmeden çıkışta motor hızı, referans hız değerinden çok az sapar.

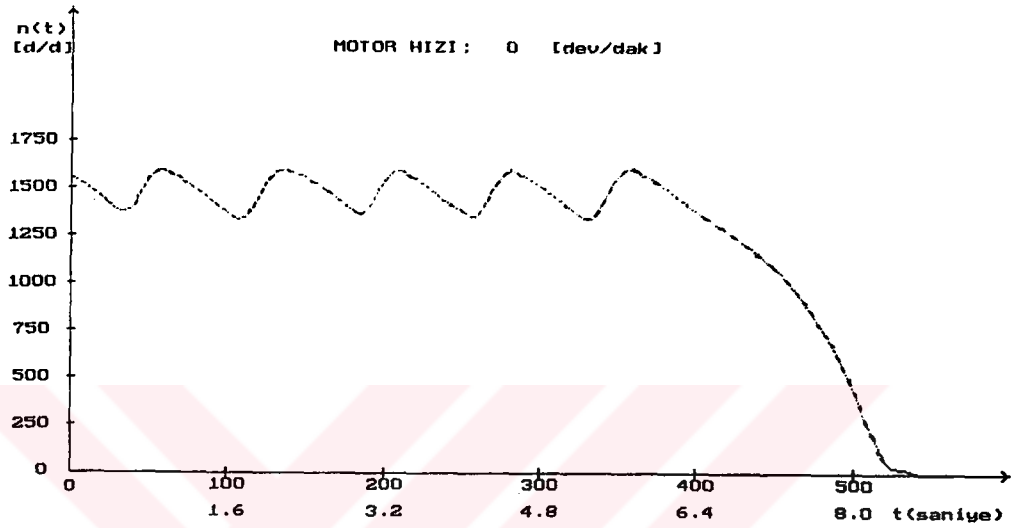


Şekil 6.6 - Yüksüz durumda, 1000 dev/dak referans hızında kendinden ayarlama için oluşturulan osilasyon.

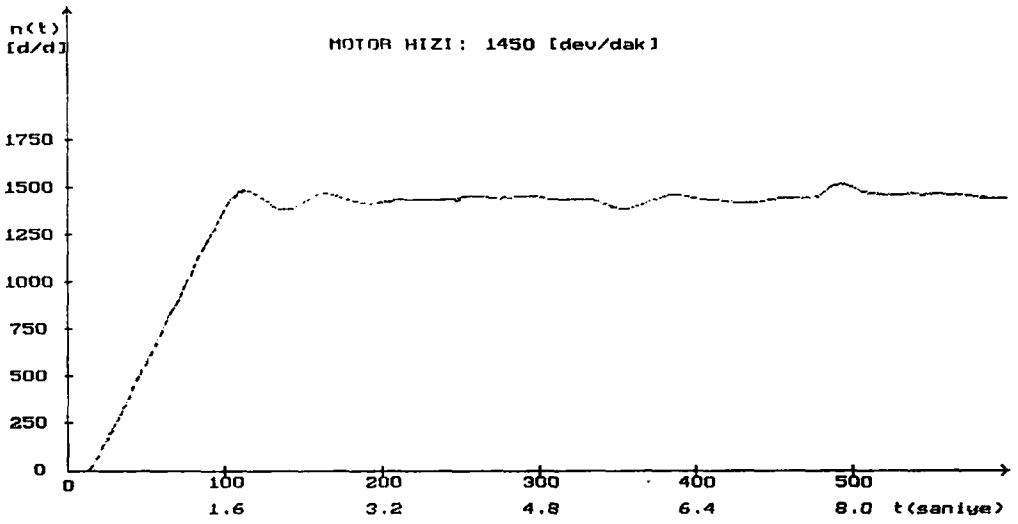


Şekil 6.7 - 1000 dev/dak referans hızı için kendinden ayarlama sonucunda, ani yüklenme ve yüklenmeden çıkışta sistemin hız değişimi.

Son olarak 1450 dev/dak referans hızı için kendinden ayarlama yapılmış ve oluşan osilasyon Şekil 6.8'de gösterilmiştir. Bu işlem sonucunda kontrolör parametreleri $K_p = 2,440$, $K_i = 1,572$, $K_d = 0,152$ olarak bulunmuş ve sistem yanıtı Şekil 6.9'da görüldüğü gibi elde edilmiştir.



Şekil 6.8 - Yüksüz durumda, 1450 dev/dak referans hızında kendinden ayarlama için oluşturulan osilasyon.



Şekil 6.9 - 1450 dev/dak referans hızı için kendinden ayarlama sonucunda, ani yüklenme ve yüklenmeden çıkışta sistemin hız değişimi.

1450 dev/dak referans hızı için sistem yine göze çarpan az bir aşım ile referans hıza erişmektedir. Yüklenmede ve yükten çıkışta motor hızı referans hızdan çok sapmadan tekrar referans hıza ulaşır.

Yapılan ölçümlerden, gerçeklenen mikrokontrolör tabanlı kendinden ayarlamalı PID kontrolörünün işlevini yerine getirdiği görülmektedir. Özellikle, lineer olmayan asenkron motor sisteminde elde edilen sonuçlar, sistemin referans hıza ulaşırken gösterdiği aşım ve küçük salınım haricinde, tatminkardır. Bu salınımın ortadan kaldırılabilmesi mümkündür. Özellikle asenkron motor üzerine yapılan çalışmalarda bu salınımın nasıl ortadan kaldırılabileceği ayrıntılı olarak incelenmiş ve sonuçlandırılmıştır [12].

Gerçeklenen kontrolörün, lineer olmayan bir sisteme uygulanması durumunda elde edilen sonuçlar yukarıda verilmiştir. Lineer olmayan sistemler üzerine yapılacak çalışmalarda, röle yöntemine, optimizasyon problemlerinin çözümü eklenerek geliştirilecek yöntemler kullanılması önerilebilir. Kontrol edilen sistem olarak, lineer olmayan asenkron motor yerine, lineer doğru akım motoru kullanılırsa, kendinden ayarlama ile elde edilen sistem davranışının kusursuz olacağı söylenebilir.

KAYNAKLAR

[1] SORANSEN, Paul H. , Peer ANDERSEN-"PI Auto-tuning Control of a Multivariable Oil-Hydraulic System"-12th World Congress International Federation of Automatic Control, Vol.10, p.393-396, (July-1993).

[2] ÅSTROM, Karl J. , Tore HÄGGLUND- "Automatic Tuning of PID Controllers"-Instrument Society of America,(1988).

[3] LANDAU, Ioan Doré-"System Identification and Control Design"- Prentice Hall, (1990).

[4] SARIOĞLU, M.Kemal-"Otomatik Kontrol I-II Ders Notu", İ.T.Ü E.E.F Yayınları, (1983).

[5] KUO, Benjamin C. -"Digital Control Systems"-Saunders College Publishing, (1992).

[6] DERİN, Haluk , Murat AŞKAR-"İletişim Kuramı, Modülasyon Yöntemleri"-ODTÜ Mühendislik Fakültesi, (1987).

[7] SARIOĞLU, M.Kemal-"Dijital Kontrol Sistemleri"-Sistem Yayıncılık ve Matbaacılık San. ve Tic. AŞ, (1992).

[8] VIDYASAGAR, M.-"Nonlinear System Analysis"-Prentice-Hall, (1987).

[9] SILJAK, Dragoslav D.-"Nonlinear Systems"-John Wiley & Sons, Inc., (1969).

[10] INTEL CORP.-"Embedded Controller Applications Handbook" -Intel Literature Sales, (1991).

[11] TIANMA CORP.-"DOT Matrix LCD Module-DMC Series"-p.33, (1990).

[12] DEMİRÖZ, Murat Ata-"Global Optimizasyon Yöntemi ile Asenkron Motor Hız Kontrolu"-Yüksek Lisans Tezi, (Şubat 1992).

EK A

MİKROKONTROLÖR ASSEMBLY PROGRAMI

```
; 490Y621 İSİMLİ PROGRAMDA;
; ÖRNEKLEME PERİYODU T=00.000 KLAVYEDEN VERİLİR,
; ADCCONV KESME İLE HER 5ms' DE BİR ÇALIŞTIRILIR,
; PID'NİN ÇALIŞMASI İÇİN Ts*10000/5'E KADAR SAYILIR,
; ORANTI, İNTEGRAL VE ÇIKIŞTA SATÜRASYON UYGULANIR,
; KENDİNDEN AYARLAMA İÇİN:
;  RÖLE KARAKTERİSTİĞİ İLE SİSTEM OSİLASYONA SOKULUR,
;  REFERANS GERİLİMİ AYARLANIR,
;  DAC ÇIKIŞINA 10V UYGULANIR,
;  HATA İŞARETİ 4.98V OLUNCA DAC ÇIKIŞINDA 0V,
;  HATA İŞARETİ 4.49V OLUNCA DAC ÇIKIŞINDA 10V İLE
;  SİSTEMİN SALINIMA GİRMESİ SAĞLANIR,
;  SİSTEM ÇIKIŞINDAKİ OSİLASYONUN Tc PERİYODU,
;  OSİLASYONUN GENLİĞİNDEN Kc KAZANCI BELİRLENİR,
;  GERÇEKLENDİĞİ TARİH: 14/01/1994 CUMA saat: 20.40

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! PROGRAMDA KULLANILAN DEĞİŞKEN VE SABİTLER !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

TUNEDAT    DATA    28H
FLAG1      BIT     TUNEDAT.0
FLAG2      BIT     TUNEDAT.1
FLAG3      BIT     TUNEDAT.2
FLAG4      BIT     TUNEDAT.3
FLAG5      BIT     TUNEDAT.4
FLAG6      BIT     TUNEDAT.5
FLAG7      BIT     TUNEDAT.6
FLAG8      BIT     TUNEDAT.7

KEYBUF     DATA    30H ; CHARACTER FROM KEYSCAN
PRCHAR     DATA    31H ; CHARACTER TO BE PRINTED
ADDRESS    DATA    32H ; ADDRESS TO BE PRESSED
ANADAT     DATA    33H ; A/D CONVERTED DATA
ANADAT_O   DATA    34H ; DIGITAL DATA FOR KNOWN SUB
ANADAT_N   DATA    35H ; (ANADAT-128)

SAB0       DATA    36H
SAB1       DATA    37H
SAB2       DATA    38H
SAB3       DATA    39H

KSAYI      DATA    3AH ; VARIABLE COEFFICIENT
TKH        DATA    3BH ; HOB OF THE COEFFICIENT
TKL        DATA    3CH ; LOB OF THE COEFFICIENT
```

```
KP3       DATA      76H ; LSB OF THE PARAMETER OF KP
KI0       DATA      77H ; MSB OF THE PARAMETER OF KI
KI1       DATA      78H
KI2       DATA      79H
KI3       DATA      7AH ; LSB OF THE PARAMETER OF KI
KD0       DATA      7BH ; MSB OF THE PARAMETER OF KD
KD1       DATA      7CH
KD2       DATA      7DH
KD3       DATA      7EH ; LSB OF THE PARAMETER OF KD

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!          PROGRAMIN BAŞLANGIÇ NOKTASI          !
!               YAPILAN İŞLEMLER:               !
! RESET VEKTÖRÜ VE KESME ADRESLERİ BELİRLENİR !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

;*********************
;   reset vector
;*********************

          ORG       0000H
          LJMP      BAS

;*********************
;   interrupts
;*********************
          ORG       03H
          RETI
          ORG       0BH
          JMP       PID
          RETI
          ORG       13H
          RETI
          ORG       1BH
          JMP       TUNEPER
          RETI
          ORG       23H
          RETI
          ORG       2BH
          RETI

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!                YAPILAN İŞLEMLER:                !
!       LCD GÖSTERGE KULLANIMA HAZIRLANIR         !
!    KULLANILACAK ZAMANLAYICILAR BELİRLENİR       !
!   DEĞİŞKENLERE BAŞLANGIÇ DEĞERLERİ VERİLİR      !
!   AÇILIŞ LOGOSU İÇİN KARAKTERLER BELİRLENİR     !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

          ORG       280H
BAS:      CALL      DELAY
          MOV       P1,#00H
          CALL      DELAY
          MOV       TMOD,#00010001B
          MOV       TL0,#0
```

```
MOV       TH0,#-20
MOV       TL1,#0
MOV       TH1,#0

SETB      EA
SETB      ET0
SETB      ET1

SETB      P3.2
CALL      DELAY
SETB      P3.3
LCALL     DELAY
LCALL     DELAY
MOV       ANADAT,#0
MOV       ANADAT_0,#0
MOV       ANADAT_N,#0

MOV       TUNEMAX,#00H
MOV       TUNEMIN,#0FFH
MOV       TC_H,#00H
MOV       TC_L,#00H
MOV       COUNT,#00H

CALL      DELAY
MOV       DPL,#00H
MOV       DPH,#80H
MOV       A,#0
MOV       @R0,A
CALL      DELAY
MOV       A,#00110000B
MOVX      @DPTR,A
LCALL     DELAY
MOV       A,#00111000B
MOVX      @DPTR,A
LCALL     DELAY
MOV       A,#00111000B
MOVX      @DPTR,A
LCALL     BUSY
MOV       A,#00000001B
MOVX      @DPTR,A
LCALL     BUSY
MOV       A,#00000100B
MOVX      @DPTR,A
LCALL     BUSY
MOV       A,#00001111B
MOVX      @DPTR,A
MOV       P1,#00H
LCALL     BUSY
MOV       A,#00111000B
MOVX      @DPTR,A
LCALL     BUSY
SETB      P3.5
CLR       P3.4
MOV       R6,#19H
```

```
XX:     CALL    CLEAR
        CALL    BUSY
        MOV     R2,#00H
TAN1:   MOV     DPH,R6
        MOV     DPL,R2
        MOV     A,#00H
        MOVC    A,@A+DPTR
        MOV     R3,A
        INC     R2
        MOV     DPH,#80H
        MOV     DPL,#01H
        MOV     A,R3
        MOVX    @DPTR,A
        LCALL   BUSY
        CJNE    R3,#2EH,TAN2
        MOV     DPH,#80H
        MOV     DPL,#00H
        MOV     A,#11000000B
        MOVX    @DPTR,A
        LCALL   BUSY
TAN2:   CJNE    R3,#21H,TAN1
        CALL    DELAY
        CALL    DELAY
        CALL    DELAY
        CALL    CLEAR
        CALL    BUSY
        CALL    KEYSCAN
        MOV     A,KEYBUF
        CJNE    A,#45H,XX

        MOV     KP0,#00H
        MOV     KP1,#00H
        MOV     KP2,#00H
        MOV     KP3,#00H
        MOV     KI0,#00H
        MOV     KI1,#00H
        MOV     KI2,#00H
        MOV     KI3,#00H
        MOV     KD0,#00H
        MOV     KD1,#00H
        MOV     KD2,#00H
        MOV     KD3,#00H

        MOV     KT0,#00H
        MOV     KT1,#00H
        MOV     KT2,#00H
        MOV     KT3,#00H
        MOV     KT4,#00H
        MOV     TKH,#00H
        MOV     TKL,#00H
        MOV     P1,#00H
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!               YAPILAN İŞLEMLER:              !
!  BU ANDAN İTİBAREN DURMA MODUNA GİRİLİR VE   !
!          TUŞA BASMA TARAMASI YAPILIR         !
!   M TUŞUNA BASILIRSA MANUEL AYARLAMA         !
!   A TUŞUNA BASILIRSA KENDİNDEN AYARLAMA      !
!   R TUŞUNA BASILIRSA ÇALIŞMA MODUNA GEÇİLİR !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

STRT:      CALL     CLEAR
CAB:       CALL     BUSY
           CALL     KEYSCAN
           MOV      A,KEYBUF
           CJNE     A,#4DH,MTUSU1
;****************************************
;***     MANUEL SETTING MODE         ***
;****************************************

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!              MANUEL AYARLAMA MODU            !
!                YAPILAN İŞLEMLER:             !
!    Kp , Ki VE Kd KATSAYILARI VE ÖRNEKLEME    !
!     ZAMANI Ts TUŞ TAKIMINDAN TEKER TEKER     !
!                    VERİLİR                   !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

KPVER:     CALL     CLEAR
           CALL     YAZK
           CALL     PRINT
           CALL     BUSY
           CALL     YAZP
           CALL     PRINT
           CALL     BUSY
           CALL     YAZESIT
           CALL     PRINT
           CALL     BUSY
           CALL     YAZKP0
           CALL     PRINT
           CALL     BUSY
           CALL     YAZNOKTA
           CALL     PRINT
           CALL     BUSY
           CALL     YAZKP1
           CALL     PRINT
           CALL     BUSY
           CALL     YAZKP2
           CALL     PRINT
           CALL     BUSY
           CALL     YAZKP3
           CALL     PRINT
           CALL     BUSY
           JMP      KPOVER_B
MTUSU1:    JMP      MTUSU2
KPOVER_B:  CALL     BETWEEN
KPOVER:    CALL     YAZESIT
```

```
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KPO_0
KPO_0:      JC      KPO_1
            JMP     KPO_BYK
KPO_1:      CJNE    A,#2FH,KPO_2
KPO_2:      JNC     KPO_3
            JMP     KPOVER
KPO_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KPO,A
            CALL    YAZKPO
            CALL    PRINT
            CALL    BUSY
            JMP     XOO
KPO_BYK:    CJNE    A,#3CH,KPO_A1
            JMP     KPOVER
KPO_A1:     CJNE    A,#3EH,KPO_A2
XOO:        CALL    ADJUST
            JMP     KP1VER_B
KPO_A2:     CJNE    A,#45H,KPOVER
            JMP     KIVER
KP1VER_B:   CALL    BETWEEN
KP1VER:     CALL    YAZNOKTA
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KP1_0
KP1_0:      JC      KP1_1
            JMP     KP1_BYK
KP1_1:      CJNE    A,#2FH,KP1_2
KP1_2:      JNC     KP1_3
            JMP     KP1VER
KP1_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KP1,A
            CALL    YAZKP1
            CALL    PRINT
            CALL    BUSY
            JMP     XO1
KP1_BYK:    CJNE    A,#3CH,KP1_A1
            CALL    ADJUST
            JMP     KPOVER_B
KP1_A1:     CJNE    A,#3EH,KP1_A2
XO1:        CALL    ADJUST
            JMP     KP2VER_B
KP1_A2:     CJNE    A,#45H,KP1VER
            JMP     KIVER
KP2VER_B:   CALL    BETWEEN
KP2VER:     CALL    YAZKP1
            CALL    PRINT
            CALL    KEYSCAN
```

```
            MOV     A,KEYBUF
            CJNE    A,#3AH,KP2_0
KP2_0:      JC      KP2_1
            JMP     KP2_BYK
KP2_1:      CJNE    A,#2FH,KP2_2
KP2_2:      JNC     KP2_3
            JMP     KP2VER
KP2_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KP2,A
            CALL    YAZKP2
            CALL    PRINT
            CALL    BUSY
            JMP     XO2
KP2_BYK:    CJNE    A,#3CH,KP2_A1
            CALL    ADJUST
            JMP     KP1VER_B
KP2_A1:     CJNE    A,#3EH,KP2_A2
XO2:        CALL    ADJUST
            JMP     KP3VER_B
KP2_A2:     CJNE    A,#45H,KP2VER
            JMP     KIVER
KP3VER_B:   CALL    BETWEEN
KP3VER:     CALL    YAZKP2
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KP3_0
KP3_0:      JC      KP3_1
            JMP     KP3_BYK
KP3_1:      CJNE    A,#2FH,KP3_2
KP3_2:      JNC     KP3_3
            JMP     KP3VER
KP3_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KP3,A
            CALL    YAZKP3
            CALL    PRINT
            CALL    BUSY
            JMP     XO3
KP3_BYK:    CJNE    A,#3CH,KP3_A1
            CALL    ADJUST
            JMP     KP2VER_B
KP3_A1:     CJNE    A,#3EH,KP3_A2
XO3:        CALL    ADJUST
            JMP     KP3VER_B
KP3_A2:     CJNE    A,#45H,KP3VER
            JMP     KIVER
KIVER:      CALL    CLEAR
            CALL    YAZK
            CALL    PRINT
            CALL    BUSY
            CALL    YAZI
```

```
            CALL    PRINT
            CALL    BUSY
            CALL    YAZESIT
            CALL    PRINT
            CALL    BUSY
            CALL    YAZKIO
            CALL    PRINT
            CALL    BUSY
            CALL    YAZNOKTA
            CALL    PRINT
            CALL    BUSY
            CALL    YAZKI1
            CALL    PRINT
            CALL    BUSY
            CALL    YAZKI2
            CALL    PRINT
            CALL    BUSY
            CALL    YAZKI3
            CALL    PRINT
            CALL    BUSY
            CALL    ADJUST
KIOVER_B:   CALL    BETWEEN
KIOVER:     CALL    YAZESIT
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KIO_0
KIO_0:      JC      KIO_1
            JMP     KIO_BYK
KIO_1:      CJNE    A,#2FH,KIO_2
KIO_2:      JNC     KIO_3
            JMP     KIOVER
KIO_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KIO,A
            CALL    YAZKIO
            CALL    PRINT
            CALL    BUSY
            JMP     X10
KIO_BYK:    CJNE    A,#3CH,KIO_A1
            JMP     KIOVER
KIO_A1:     CJNE    A,#3EH,KIO_A2
X10:        CALL    ADJUST
            JMP     KI1VER_B
KIO_A2:     CJNE    A,#45H,KIOVER
            JMP     KDVER
KI1VER_B:   CALL    BETWEEN
KI1VER:     CALL    YAZNOKTA
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KI1_0
KI1_0:      JC      KI1_1
            JMP     KI1_BYK
```

```
KI1_1:      CJNE    A,#2FH,KI1_2
KI1_2:      JNC     KI1_3
            JMP     KI1VER
KI1_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#3OH
            MOV     KI1,A
            CALL    YAZKI1
            CALL    PRINT
            CALL    BUSY
            JMP     X11
KI1_BYK:    CJNE    A,#3CH,KI1_A1
            CALL    ADJUST
            JMP     KIOVER_B
KI1_A1:     CJNE    A,#3EH,KI1_A2
X11:        CALL    ADJUST
            JMP     KI2VER_B
KI1_A2:     CJNE    A,#45H,KI1VER
            JMP     KDVER
KI2VER_B:   CALL    BETWEEN
KI2VER:     CALL    YAZKI1
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KI2_O
KI2_O:      JC      KI2_1
            JMP     KI2_BYK
KI2_1:      CJNE    A,#2FH,KI2_2
KI2_2:      JNC     KI2_3
            JMP     KI2VER
KI2_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#3OH
            MOV     KI2,A
            CALL    YAZKI2
            CALL    PRINT
            CALL    BUSY
            JMP     X12
KI2_BYK:    CJNE    A,#3CH,KI2_A1
            CALL    ADJUST
            JMP     KI1VER_B
KI2_A1:     CJNE    A,#3EH,KI2_A2
X12:        CALL    ADJUST
            JMP     KI3VER_B
KI2_A2:     CJNE    A,#45H,KI2VER
            JMP     KDVER

KI3VER_B:   CALL    BETWEEN
KI3VER:     CALL    YAZKI2
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KI3_O
KI3_O:      JC      KI3_1
            JMP     KI3_BYK
```

```
KI3_1:      CJNE    A,#2FH,KI3_2
KI3_2:      JNC     KI3_3
            JMP     KI3VER
KI3_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KI3,A
            CALL    YAZKI3
            CALL    PRINT
            CALL    BUSY
            JMP     X13
KI3_BYK:    CJNE    A,#3CH,KI3_A1
            CALL    ADJUST
            JMP     KI2VER_B
KI3_A1:     CJNE    A,#3EH,KI3_A2
X13:        CALL    ADJUST
            JMP     KI3VER_B
KI3_A2:     CJNE    A,#45H,KI3VER
            JMP     KDVER

MTUSU2:     JMP     MTUSU3
KDVER:      CALL    CLEAR
            CALL    YAZK
            CALL    PRINT
            CALL    BUSY
            CALL    YAZD
            CALL    PRINT
            CALL    BUSY
            CALL    YAZESIT
            CALL    PRINT
            CALL    BUSY
            CALL    YAZKDO
            CALL    PRINT
            CALL    BUSY
            CALL    YAZNOKTA
            CALL    PRINT
            CALL    BUSY
            CALL    YAZKD1
            CALL    PRINT
            CALL    BUSY
            CALL    YAZKD2
            CALL    PRINT
            CALL    BUSY
            CALL    YAZKD3
            CALL    PRINT
            CALL    BUSY
            CALL    ADJUST
KDOVER_B:   CALL    BETWEEN
KDOVER:     CALL    YAZESIT
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KDO_0
KDO_0:      JC      KDO_1
            JMP     KDO_BYK
```

```
KDO_1:      CJNE    A,#2FH,KDO_2
KDO_2:      JNC     KDO_3
            JMP     KDOVER
KDO_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KDO,A
            CALL    YAZKDO
            CALL    PRINT
            CALL    BUSY
            JMP     X20
KDO_BYK:    CJNE    A,#3CH,KDO_A1
            JMP     KDOVER
KDO_A1:     CJNE    A,#3EH,KDO_A2
X20:        CALL    ADJUST
            JMP     KD1VER_B
KDO_A2:     CJNE    A,#45H,KDOVER
            JMP     KDSONU
KD1VER_B:   CALL    BETWEEN
KD1VER:     CALL    YAZNOKTA
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KD1_0
KD1_0:      JC      KD1_1
            JMP     KD1_BYK
KD1_1:      CJNE    A,#2FH,KD1_2
KD1_2:      JNC     KD1_3
            JMP     KD1VER
KD1_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KD1,A
            CALL    YAZKD1
            CALL    PRINT
            CALL    BUSY
            JMP     X21
KD1_BYK:    CJNE    A,#3CH,KD1_A1
            CALL    ADJUST
            JMP     KDOVER_B
KD1_A1:     CJNE    A,#3EH,KD1_A2
X21:        CALL    ADJUST
            JMP     KD2VER_B
KD1_A2:     CJNE    A,#45H,KD1VER
            JMP     KDSONU
KD2VER_B:   CALL    BETWEEN
KD2VER:     CALL    YAZKD1
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KD2_0
KD2_0:      JC      KD2_1
            JMP     KD2_BYK
KD2_1:      CJNE    A,#2FH,KD2_2
KD2_2:      JNC     KD2_3
```

```
                JMP     KD2VER
KD2_3:          MOV     A,KEYBUF
                CLR     C
                SUBB    A,#3OH
                MOV     KD2,A
                CALL    YAZKD2
                CALL    PRINT
                CALL    BUSY
                JMP     X22
KD2_BYK:        CJNE    A,#3CH,KD2_A1
                CALL    ADJUST
                JMP     KD1VER_B
KD2_A1:         CJNE    A,#3EH,KD2_A2
X22:            CALL    ADJUST
                JMP     KD3VER_B
KD2_A2:         CJNE    A,#45H,KD2VER
                JMP     KDSONU
KD3VER_B:       CALL    BETWEEN
KD3VER:         CALL    YAZKD2
                CALL    PRINT
                CALL    KEYSCAN
                MOV     A,KEYBUF
                CJNE    A,#3AH,KD3_O
KD3_O:          JC      KD3_1
                JMP     KD3_BYK
KD3_1:          CJNE    A,#2FH,KD3_2
KD3_2:          JNC     KD3_3
                JMP     KD3VER
KD3_3:          MOV     A,KEYBUF
                CLR     C
                SUBB    A,#3OH
                MOV     KD3,A
                CALL    YAZKD3
                CALL    PRINT
                CALL    BUSY
                JMP     X23
KD3_BYK:        CJNE    A,#3CH,KD3_A1
                CALL    ADJUST
                JMP     KD2VER_B
KD3_A1:         CJNE    A,#3EH,KD3_A2
X23:            CALL    ADJUST
                JMP     KD3VER_B
KD3_A2:         CJNE    A,#45H,KD3VER
KDSONU:         CALL    CLEAR
                CALL    YAZT
                CALL    PRINT
                CALL    BUSY
                CALL    YAZS
                CALL    PRINT
                CALL    BUSY
                CALL    YAZESIT
                CALL    PRINT
                CALL    BUSY
                CALL    YAZKTO
                CALL    PRINT
```

```
                CALL    BUSY
                CALL    YAZKT1
                CALL    PRINT
                CALL    BUSY
                CALL    YAZNOKTT
                CALL    PRINT
                CALL    BUSY
                CALL    YAZKT2
                CALL    PRINT
                CALL    BUSY
                CALL    YAZKT3
                CALL    PRINT
                CALL    BUSY
                CALL    YAZKT4
                CALL    PRINT
                CALL    BUSY
                CALL    ADJUST
KTOVER_B:       CALL    BETWEEN
KTOVER:         CALL    YAZESIT
                CALL    PRINT
                CALL    BUSY
                CALL    KEYSCAN
                MOV     A,KEYBUF
                CJNE    A,#3AH,KTO_0
KTO_0:          JC      KTO_1
                JMP     KTO_BYK
KTO_1:          CJNE    A,#2FH,KTO_2
KTO_2:          JNC     KTO_3
                JMP     KTOVER
KTO_3:          MOV     A,KEYBUF
                CJNE    A,#35H,KTT1
KTT1:           JNC     KTOVER
                CLR     C
                SUBB    A,#30H
                MOV     KTO,A
                CALL    YAZKTO
                CALL    PRINT
                CALL    BUSY
                JMP     X30
KTO_BYK:        CJNE    A,#3CH,KTO_A1
                JMP     KTOVER
KTO_A1:         CJNE    A,#3EH,KTO_A2
X30:            CALL    ADJUST
                JMP     KT1VER_B
KTO_A2:         CJNE    A,#45H,KTOVER
                JMP     KTSONU
KT1VER_B:       CALL    BETWEEN
KT1VER:         CALL    YAZKTO
                CALL    PRINT
                CALL    KEYSCAN
                MOV     A,KEYBUF
                CJNE    A,#3AH,KT1_0
KT1_0:          JC      KT1_1
                JMP     KT1_BYK
KT1_1:          CJNE    A,#2FH,KT1_2
```

```
KT1_2:      JNC       KT1_3
            JMP       KT1VER
KT1_3:      MOV       A,KEYBUF
            CLR       C
            SUBB      A,#30H
            MOV       KT1,A
            CALL      YAZKT1
            CALL      PRINT
            CALL      BUSY
            JMP       X31
KT1_BYK:    CJNE      A,#3CH,KT1_A1
            CALL      ADJUST
            JMP       KTOVER_B
KT1_A1:     CJNE      A,#3EH,KT1_A2
X31:        CALL      ADJUST
            JMP       KT2VER_B
KT1_A2:     CJNE      A,#45H,KT1VER
            JMP       KTSONU
KT2VER_B:   CALL      BETWEEN
KT2VER:     CALL      YAZNOKTT
            CALL      PRINT
            CALL      KEYSCAN
            MOV       A,KEYBUF
            CJNE      A,#3AH,KT2_0
KT2_0:      JC        KT2_1
            JMP       KT2_BYK
KT2_1:      CJNE      A,#2FH,KT2_2
KT2_2:      JNC       KT2_3
            JMP       KT2VER
KT2_3:      MOV       A,KEYBUF
            CLR       C
            SUBB      A,#30H
            MOV       KT2,A
            CALL      YAZKT2
            CALL      PRINT
            CALL      BUSY
            JMP       X32
KT2_BYK:    CJNE      A,#3CH,KT2_A1
            CALL      ADJUST
            JMP       KT1VER_B
KT2_A1:     CJNE      A,#3EH,KT2_A2
X32:        CALL      ADJUST
            JMP       KT3VER_B
KT2_A2:     CJNE      A,#45H,KT2VER
            JMP       KTSONU
KT3VER_B:   CALL      BETWEEN
KT3VER:     CALL      YAZKT2
            CALL      PRINT
            CALL      KEYSCAN
            MOV       A,KEYBUF
            CJNE      A,#3AH,KT3_0
KT3_0:      JC        KT3_1
            JMP       KT3_BYK
KT3_1:      CJNE      A,#2FH,KT3_2
KT3_2:      JNC       KT3_3
```

```
            JMP     KT3VER
KT3_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KT3,A
            CALL    YAZKT3
            CALL    PRINT
            CALL    BUSY
            JMP     X33
KT3_BYK:    CJNE    A,#3CH,KT3_A1
            CALL    ADJUST
            JMP     KT2VER_B
KT3_A1:     CJNE    A,#3EH,KT3_A2
X33:        CALL    ADJUST
            JMP     KT4VER_B
KT3_A2:     CJNE    A,#45H,KT3VER
KT4VER_B:   CALL    BETWEEN
KT4VER:     CALL    YAZKT3
            CALL    PRINT
            CALL    KEYSCAN
            MOV     A,KEYBUF
            CJNE    A,#3AH,KT4_0
KT4_0:      JC      KT4_1
            JMP     KT4_BYK
KT4_1:      CJNE    A,#2FH,KT4_2
KT4_2:      JNC     KT4_3
            JMP     KT4VER
KT4_3:      MOV     A,KEYBUF
            CLR     C
            SUBB    A,#30H
            MOV     KT4,A
            CALL    YAZKT4
            CALL    PRINT
            CALL    BUSY
            JMP     X53
KT4_BYK:    CJNE    A,#3CH,KT4_A1
            CALL    ADJUST
            JMP     KT3VER_B
KT4_A1:     CJNE    A,#3EH,KT4_A2
X53:        CALL    ADJUST
            JMP     KT4VER_B
KT4_A2:     CJNE    A,#45H,KT4VER
KTSONU:     MOV     A,KT0
            CJNE    A,#00H,ROUTE
            MOV     A,KT1
            CJNE    A,#00H,ROUTE
            MOV     A,KT2
            CJNE    A,#00H,ROUTE
            MOV     A,KT3
            CJNE    A,#00H,ROUTE
            MOV     A,KT4
            CJNE    A,#05H,SUBRO
SUBRO:      JNC     ROUTE
            JMP     KDSONU
ROUTE:      CALL    DELAY
```

```
        CALL    CLEAR
        CALL    DELAY
EE:     MOV     TKP_H,#00H
        MOV     TKP_L,#00H
        MOV     TKI_H,#00H
        MOV     TKI_L,#00H
        MOV     TKD_H,#00H
        MOV     TKD_L,#00H
        MOV     TKT_H,#00H
        MOV     TKT_L,#00H
        MOV     A,KP3
        ADD     A,TKP_L
        MOV     TKP_L,A
        MOV     A,KP2
        MOV     B,#0AH
        MUL     AB
        ADD     A,TKP_L
        MOV     TKP_L,A
        MOV     KSAYI,KP1
        CALL    ATA_100
        CALL    MUL_16
        MOV     TKH,TKP_H
        MOV     TKL,TKP_L
        CLR     C
        CALL    ADDING
        MOV     TKP_L,TKL
        MOV     TKP_H,TKH
        MOV     KSAYI,KP0
        CALL    ATA_1000
        CALL    MUL_16
        MOV     TKH,TKP_H
        MOV     TKL,TKP_L
        CALL    ADDING
        MOV     TKP_L,TKL
        MOV     TKP_H,TKH
        MOV     A,KI3
        ADD     A,TKI_L
        MOV     TKI_L,A
        MOV     A,KI2
        MOV     B,#0AH
        MUL     AB
        ADD     A,TKI_L
        MOV     TKI_L,A
        MOV     KSAYI,KI1
        CALL    ATA_100
        CALL    MUL_16
        MOV     TKH,TKI_H
        MOV     TKL,TKI_L
        CLR     C
        CALL    ADDING
        MOV     TKI_L,TKL
        MOV     TKI_H,TKH
        MOV     KSAYI,KI0
        CALL    ATA_1000
        CALL    MUL_16
```

```
MOV     TKH,TKI_H
MOV     TKL,TKI_L
CALL    ADDING
MOV     TKI_L,TKL
MOV     TKI_H,TKH
MOV     A,KD3
ADD     A,TKD_L
MOV     TKD_L,A
MOV     A,KD2
MOV     B,#0AH
MUL     AB
ADD     A,TKD_L
MOV     TKD_L,A
MOV     KSAYI,KD1
CALL    ATA_100
CALL    MUL_16
MOV     TKH,TKD_H
MOV     TKL,TKD_L
CLR     C
CALL    ADDING
MOV     TKD_L,TKL
MOV     TKD_H,TKH
MOV     KSAYI,KD0
CALL    ATA_1000
CALL    MUL_16
MOV     TKH,TKD_H
MOV     TKL,TKD_L
CALL    ADDING
MOV     TKD_L,TKL
MOV     TKD_H,TKH
MOV     A,KT4
ADD     A,TKT_L
MOV     TKT_L,A
MOV     A,KT3
MOV     B,#0AH
MUL     AB
ADD     A,TKT_L
MOV     TKT_L,A
MOV     KSAYI,KT2
CALL    ATA_100
CALL    MUL_16
MOV     TKH,TKT_H
MOV     TKL,TKT_L
CLR     C
CALL    ADDING
MOV     TKT_L,TKL
MOV     TKT_H,TKH
MOV     KSAYI,KT1
CALL    ATA_1000
CALL    MUL_16
MOV     TKH,TKT_H
MOV     TKL,TKT_L
CALL    ADDING
MOV     TKT_L,TKL
MOV     TKT_H,TKH
```

```
                MOV     KSAYI,KTO
                CALL    ATA_10000
                CALL    MUL_16
                MOV     TKH,TKT_H
                MOV     TKL,TKT_L
                CLR     C
                CALL    ADDING
                MOV     TKT_L,TKL
                MOV     TKT_H,TKH
                MOV     OP3,#00H
                MOV     OP2,#00H
                MOV     OP1,TKT_H
                MOV     OP0,TKT_L
                MOV     DIV0,#00H
                MOV     DIV1,#05H
                CALL    DIV_16
                MOV     TKT_HH,OP1
                MOV     TKT_LL,OP0
                INC     TKT_HH
                JMP     DD
EE1:            JMP     EE
DD:             MOV     PRCHAR,TKP_H
                MOV     ADDRESS,#07H
                CALL    PRINT
                CALL    BUSY
                MOV     PRCHAR,TKP_L
                MOV     ADDRESS,#08H
                CALL    PRINT
                CALL    BUSY
                MOV     PRCHAR,TKI_H
                MOV     ADDRESS,#0AH
                CALL    PRINT
                CALL    BUSY
                MOV     PRCHAR,TKI_L
                MOV     ADDRESS,#0BH
                CALL    PRINT
                CALL    BUSY
                MOV     PRCHAR,TKD_H
                MOV     ADDRESS,#0DH
                CALL    PRINT
                CALL    BUSY
                MOV     PRCHAR,TKD_L
                MOV     ADDRESS,#0EH
                CALL    PRINT
                CALL    BUSY
                MOV     PRCHAR,TKT_H
                MOV     ADDRESS,#4DH
                CALL    PRINT
                CALL    BUSY
                MOV     PRCHAR,TKT_L
                MOV     ADDRESS,#4EH
                CALL    PRINT
                CALL    BUSY
                CALL    KEYSCAN
                MOV     A,KEYBUF
```

```
            CJNE    A,#3EH,EE1
            CALL    CLEAR
            CALL    DELAY
MTUSU3:     CJNE    A,#41H,ATSON
            CLR     ETO
            CLR     FLAG1
            MOV     TC_H,#OOH
            MOV     TC_L,#OOH
            MOV     COUNT,#OOH
            MOV     R4,#OOH
            MOV     R5,#OOH
            MOV     TL1,#O
            MOV     TH1,#O
            MOV     TUNEMAX,#OOH
            MOV     TUNEMIN,#OFFH
            JMP     AUTOTUNE
ATSON:      JMP     ATSONU

;*****************************************************
;***              AUTO-TUNING MODE                 ***
;*****************************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!                KENDİNDEN AYARLAMA MODU             !
!                   YAPILAN İŞLEMLER:                !
!HİSTEREZİSLİ RÖLE KARAKTERİSTİĞİYLE BİR OSİLASYON  !
!OLUŞTURULUR.  BU SALINIMIN PERİYODU VE GENLİĞİNDEN!
!    Kc=4.(Umax-Umin)/pi.(TUNEMAX-TUNEMIN) , Tc     !
!  BELİRLENİR. BURADAN  Kp=0,6.Kc , Ki=2/Tc ,       !
!                  Kd=0,12.Tc BULUNUR.              !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

AUTOTUNE:   MOV     P1,#OFFH
            CALL    ADCCONV
            MOV     A,ANADAT
            CPL     A
            ANL     A,#O1111111B
            CJNE    A,#3EH,AUTOHIGH
AUTOHIGH:   JC      AUTOTUNE
AUTO:       MOV     P1,#OOH
            CALL    ADCCONV
            MOV     A,ANADAT
            CPL     A
            ANL     A,#O1111111B
            CJNE    A,#35H,AUTOLOW
AUTOLOW:    JNC     AUTO
            INC     COUNT
            MOV     A,COUNT
            CJNE    A,#OCH,REANI
REANI:      JC      AUTOTUNE
            CJNE    A,#OCH,REKAR
ATUNE:      MOV     P1,#OFFH
            CALL    ADCCONV
            MOV     A,ANADAT
            CPL     A
            ANL     A,#O1111111B
```

```
            CJNE    A,TUNEMIN,DDD2
DDD2:       JNC     SSS2
            MOV     TUNEMIN,A
SSS2:       CJNE    A,#3EH,AUTOH
AUTOH:      JC      ATUNE
AUT:        MOV     P1,#OOH
            CALL    ADCCONV
            MOV     A,ANADAT
            CPL     A
            ANL     A,#O1111111B
            CJNE    A,TUNEMAX,DDD3
DDD3:       JC      ROLL
            MOV     TUNEMAX,A
ROLL:       CJNE    A,#35H,AULOW
AULOW:      JNC     AUT
REKAR:      MOV     P1,#OFFH
            CALL    ADCCONV
            MOV     A,ANADAT
            CPL     A
            ANL     A,#O1111111B
            CJNE    A,#3EH,AHIGH
AHIGH:      JC      REKAR
            JNB     FLAG1,GLR
            CLR     TR1
            MOV     TC_H,R4
            MOV     TC_L,TH1
            CLR     FLAG1
            JMP     EEEE
GLR:        SETB    TR1
            SETB    FLAG1
AUTTO:      MOV     P1,#OOH
            CALL    ADCCONV
            MOV     A,ANADAT
            CPL     A
            ANL     A,#O1111111B
            CJNE    A,#35H,ALOW
ALOW:       JNC     AUTTO
            JMP     REKAR
EEEE:       MOV     P1,#OOH
            MOV     PRCHAR,#54H
            MOV     ADDRESS,#43H
            CALL    PRINT
            CALL    BUSY
            MOV     PRCHAR,#55H
            MOV     ADDRESS,#44H
            CALL    PRINT
            CALL    BUSY
            MOV     PRCHAR,#4EH
            MOV     ADDRESS,#45H
            CALL    PRINT
            CALL    BUSY
            MOV     PRCHAR,#45H
            MOV     ADDRESS,#46H
            CALL    PRINT
            CALL    BUSY
```

```
        MOV     PRCHAR,#44H
        MOV     ADDRESS,#47H
        CALL    PRINT
        CALL    BUSY
        MOV     PRCHAR,#4FH
        MOV     ADDRESS,#49H
        CALL    PRINT
        CALL    BUSY
        MOV     PRCHAR,#4BH
        MOV     ADDRESS,#4AH
        CALL    PRINT
        CALL    BUSY
        MOV     PRCHAR,#21H
        MOV     ADDRESS,#4BH
        CALL    PRINT
        CALL    BUSY
LOOP:   CALL    KEYSCAN
        MOV     A,KEYBUF
        CJNE    A,#45H,LOOP
        CALL    CLEAR
        MOV     A,#00001111B
        MOVX    @DPTR,A
        CALL    DELAY
        CALL    CLEAR
        SETB    ET
        MOV     R2,#00H
        MOV     R3,#00H
        MOV     R4,#00H
        MOV     R5,#00H
        MOV     R6,#00H
        MOV     R7,#00H
        MOV     OP3,#00H
        MOV     OP2,#00H
        MOV     OP1,#0BFH
        MOV     OP0,#015H
        MOV     A,TUNEMAX
        CLR     C
        SUBB    A,TUNEMIN
        MOV     DIV0,#00H
        MOV     DIV1,A
        CALL    DIV_16
        MOV     OP3,#00H
        CALL    BINBCDCON
        MOV     KP0,R5
        MOV     KP1,R4
        MOV     KP2,R3
        MOV     KP3,R2
        MOV     OP3,#00H
        MOV     OP2,#00H
        MOV     OP1,TC_H
        MOV     OP0,TC_L
        MOV     DIV0,#00H
        MOV     DIV1,#0C8H
        CALL    DIV_16
        MOV     OP3,#00H
```

```
                CALL    BINBCDCON
                MOV     KIO,R5
                MOV     KI1,R4
                MOV     KI2,R3
                MOV     KI3,R2
                MOV     OP3,#OOH
                MOV     OP2,#OOH
                MOV     OP1,TC_H
                MOV     OPO,TC_L
                MOV     MULO,#OOH
                MOV     MUL1,#06H
                CALL    MUL_16
                MOV     DIVO,#27H
                MOV     DIV1,#1OH
                CALL    DIV_16
                MOV     OP3,#OOH
                CALL    BINBCDCON
                MOV     KDO,R5
                MOV     KD1,R4
                MOV     KD2,R3
                MOV     KD3,R2
                JMP     CAB
ATSONU:         CJNE    A,#52H,RT
                SETB    TRO
                MOV     R3,TKT_HH
                MOV     R2,TKT_LL
                JMP     RTUSU
RT:             LJMP    RTSONU

;*****************************************************
;***                    RUN MODE                   ***
;*****************************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!                     ÇALIŞMA  MODU                   !
!                   YAPILAN İŞLEMLER:                 !
!      HER 5ms'DE BİR ZAMANLAYICI  KESMESİ İLE        !
!            PID ALGORİTMASINA DALLANILIR.            !
!ÖRNEKLEME ZAMANI 5ms İSE PID PROGRAMI ÇALIŞTIRILIR!
!         5ms'DEN BÜYÜK ÖRNEKLEME ZAMANI İÇİN         !
! (T-5ms) SURESİ KADAR BİR BEKLEME YARATILIR VE PID!
!   PROGRAMI BU SÜRE DOLDUKTAN SONRA ÇALIŞTIRILIR. !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

RTUSU:          CALL    CLEAR
                LCALL   BUSY
                MOV     A,#00001100B
                MOVX    @DPTR,A
                MOV     DOI_L,#OOH
                MOV     DOD_L,#OOH
                MOV     DND_L,#OOH
                MOV     ANADAT,#O
                MOV     ANADAT_O,#O
                MOV     ANADAT_N,#O
                CALL    KEYSCAN
                MOV     A,KEYBUF
```

```
            CJNE    A,#53H,RTUS
            MOV     P1,#00H
            CLR     TR0
            MOV     A,#00001111B
            MOVX    @DPTR,A
            CALL    DELAY
            CALL    CLEAR
            JMP     RTSONU
RTUS:       JMP     RTUSU
RTSONU:     LJMP    CAB

;************************************************
;*  DETERMINING THE PERIOD OF THE OSSILATION  *
;************************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!       SALINIMIN PERIYODUNUN BELİRLENMESİ      !
!              YAPILAN İŞLEMLER:                !
!ZAMANLAYICI ÇALIŞTIRILARAK, HER 1µs'DE TL0'IN !
!  HER 256µs'DE TH0'IN AKTİF OLMASI SAĞLANIR.  !
!  TH0 AKTİF OLDUĞUNDA R4 REGISTER'I İÇERİĞİ   !
!              BİR ARTAR.                       !
!   BURADAKİ SÜRE SALINIMIN PERİYODUNU VERİR.   !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

TUNEPER:    CJNE    R4,#0FFH,NEXT
            MOV     R4,#00H
            INC     R5
            JMP     NEXTSTEP
NEXT:       INC     R4
NEXTSTEP:   MOV     TL1,#0
            MOV     TH1,#0
            RETI

;************************************************
;***              PID ALGORITHM               ***
;***            RUNNING BY MEANS OF           ***
;***              TIMER INTERRUPT             ***
;************************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!               PID ALGORİTMASI                 !
!              YAPILAN İŞLEMLER:                !
!    ÇALIŞMA MODUNDA BELİRTİLDİĞİ ŞEKİLDE      !
!          TRAPEZOİDAL YAPIDAKİ PID             !
!        KONTROLÖRÜNÜN, HATA İŞARETİNİ          !
!      YORUMLAYIP , YENİ KONTROL İŞARETİ        !
!            BELİRLEDİĞİ PROGRAM.               !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

PID:        CALL    ADCCONV
            DJNZ    R2,TT
            MOV     R2,TKT_LL
            DJNZ    R3,TT
            MOV     R3,TKT_HH
            JMP     PIDD
TT:         JMP     TTT
```

```
PIDD:     MOV     TKH,TKP_H
          MOV     TKL,TKP_L
          MOV     A,ANADAT
          CLR     C
          SUBB    A,#80H
          JNC     TAPA
          MOV     ANADAT_O,#00H
          JMP     TAPAN
TAPA:     MOV     ANADAT_O,A
TAPAN:    CALL    KNOWN
          CALL    MUL_16
          CALL    THOUSAND
          CALL    DIV_16
          MOV     A,OP1
          CJNE    A,#00H,TAP1
          MOV     OUTP_L,OP0
          JMP     INTEGRA
TAP1:     MOV     OUTP_L,#0FFH
          MOV     OUTP_H,#00H

INTEGRA:  MOV     TKH,TKI_H
          MOV     TKL,TKI_L
          MOV     A,ANADAT
          CLR     C
          CJNE    A,#80H,NEQL128
          MOV     OUTI_L,OUTI_L
          JMP     DIFER
NEQL128:  JC      ASMALL
          MOV     ANADAT_N,ANADAT
          MOV     A,ANADAT_N
          CLR     C
          SUBB    A,#80H
          MOV     ANADAT_O,A
          CALL    KNOWN
          CALL    MUL_16
          CALL    THOUSAND
          CALL    DIV_16
          MOV     A,OP0
          CLR     C
          ADD     A,DOI_L
          JNC     RESULT
          MOV     A,#0FFH
RESULT:   MOV     DOI_L,A
          MOV     OUTI_L,A
          JMP     DIFER
ASMALL:   MOV     ANADAT_N,ANADAT
          MOV     A,#80H
          CLR     C
          SUBB    A,ANADAT_N
          MOV     ANADAT_O,ANADAT_N
          CALL    KNOWN
          CALL    MUL_16
          CALL    THOUSAND
          CALL    DIV_16
          MOV     A,DOI_L
```

```
            CLR     C
            CJNE    A,OPO,NEG2
            JMP     NEG3
NEG2:       JNC     KBIG
            JMP     NEG3
KBIG:       CLR     C
            SUBB    A,OPO    ;(ANADAT_N-ANADAT)*KI/1000
            JNC     RESU
NEG3:       MOV     A,#00H
RESU:       MOV     DOI_L,A
            MOV     OUTI_L,A

DIFER:      MOV     A,ANADAT
            MOV     DND_L,ANADAT
            CJNE    A,DOD_L,DIFA
            MOV     DOD_L,DND_L
            MOV     OUTD_L,#00H
            JMP     EQL
DIFA:       JC      ADNK
            SUBB    A,DOD_L
            MOV     ANADAT_O,A
            MOV     DOD_L,DND_L
            MOV     TKH,TKD_H
            MOV     TKL,TKD_L
            CALL    KNOWN
            CALL    MUL_16
            CALL    THOUSAND
            CALL    DIV_16
            MOV     A,OP1
            CJNE    A,#00H,DIFDIF
            JMP     DIFEQ
DIFDIF:     MOV     OPO,#0FFH
DIFEQ:      MOV     OUTD_L,OPO
EQL:        MOV     A,OUTP_L
            CLR     C
            ADD     A,OUTI_L
            JNC     SSS
            CLR     C
            MOV     A,#0FFH
SSS:        ADD     A,OUTD_L
            JNC     SSSS
            MOV     A,#0FFH
SSSS:       MOV     OUTALL_L,A
            MOV     P1,OUTALL_L
            JMP     TTT
ADNK:       MOV     ANADAT_O,A
            MOV     A,DOD_L
            MOV     DOD_L,ANADAT_O
            CLR     C
            SUBB    A,DOD_L
            MOV     ANADAT_O,A
            MOV     DOD_L,DND_L
            MOV     TKH,TKD_H
            MOV     TKL,TKD_L
            CALL    KNOWN
```

```
            CALL      MUL_16
            CALL      THOUSAND
            CALL      DIV_16
            MOV       A,OP1
            CJNE      A,#OOH,DIFDIFF
            JMP       DIFEQQ
DIFDIFF:    MOV       OPO,#OFFH
DIFEQQ:     MOV       OUTD_L,OPO
            MOV       A,OUTP_L
            CLR       C
            ADD       A,OUTI_L
            JNC       WWW
            CLR       C
            MOV       A,#OFFH
WWW:        SUBB      A,OUTD_L
            JNC       WWWW
            MOV       A,#OOH
WWWW:       MOV       OUTALL_L,A
            MOV       P1,OUTALL_L
TTT:        MOV       TLO,#O
            MOV       THO,#-20
            RETI

;*********************************
;*          D E L A Y            *
;*********************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!      GECİKME ALTPROGRAMI      !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

DELAY:      MOV       RO,#OFFH
UBD:        MOV       R1,#OFFH
            NOP
SUB:        NOP
            DJNZ      R1,SUB
            NOP
            DJNZ      RO,UBD
            RET
;*********************************
;*             BUSY              *
;*********************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! GÖSTERGEDE YÜRÜTÜLEN İŞLEMİN !
!  TAMAMLANMASI İÇİN BEKLEME   !
!         ALTPROGRAMI           !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

BUSY:       MOV       DPH,#8OH
            MOV       DPL,#O2H
BUSY1:      MOVX      A,@DPTR
            ANL       A,#10000000B
            NOP
            CJNE      A,#OOH,BUSY1
            MOV       DPL,#OOH
            RET
```

```
;*********************************
;*       SCANNING OF KEYBOARD    *
;*        OUTPUTS TO KEYBUF      *
;*   REGISTER BANK USED IS 01    *
;*********************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!TUŞ TAKIMINI TARAYAN VE BASILAN!
!  TUŞU BELİRLEYEN ALTPROGRAM   !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

KEYSCAN:  SETB     P3.0
          SETB     P3.4 ; RESETS 4017
          CALL     SELAY
          CLR      P3.4
          SETB     P3.1 ; LOW DECADE OF KEYBUF
          CALL     SELAY
          MOV      R0,#0
KEYAG:    JNB      P3.0,KEYOUT
          INC      R0
          CALL     SELAY
          SETB     P3.5  ; SINGLE CLOCK PULSE TO 4017
          CALL     SELAY
          CLR      P3.5
          CJNE     R0,#10,KEYAG
          CLR      P3.1
          SETB     P3.4
          CALL     SELAY
          CLR      P3.4
          CALL     SELAY
          MOV      R0,#0
KEYAG2:   JNB      P3.0,KEYOUT
          INC      R0
          SETB     P3.5
          CALL     SELAY
          CLR      P3.5
          CJNE     R0,#10,KEYAG2
          JMP      OUT
KEYOUT:   MOV      DPTR,#2000H
          MOV      A,R0
          JB       P3.1,SECDEC
          MOV      DPTR,#2010H
SECDEC:   MOVC     A,@A+DPTR
          MOV      KEYBUF,A
OUT:      MOV      DPTR,#8000H
          RET

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!      KÜÇÜK GECİKME YARATAN    !
!          ALTPROGRAM           !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
SELAY:    MOV      R7,#40H
KON:      NOP
          DJNZ     R7,KON
          RET
```

```
;****************************************
;*         CLEAR    DISPLAY             *
;****************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!     GÖSTERGEDE BULUNAN VERİLERİ       !
!    TEMİZLEMEYE YARAYAN ALTPROGRAM     !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

CLEAR:     PUSH     PSW
           PUSH     ACC
           PUSH     DPL
           PUSH     DPH
           MOV      DPTR,#8000H
           MOV      A,#01
           MOVX     @DPTR,A
           POP      DPH
           POP      DPL
           POP      ACC
           POP      PSW
           RET

;****************************************
;*           PRINT A CHARACTER          *
;*      PRINTS  PRCHAR INTO ADDRESS     *
;****************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! GÖSTERGEDE VERİLEN ADRESE BELİRLENEN  !
!    KARAKTERİN YAZILMASINI SAĞLAYAN    !
!            ALTPROGRAM                 !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

PRINT:     PUSH     PSW
           PUSH     ACC
           PUSH     DPL
           PUSH     DPH
           MOV      DPTR,#8000H
           MOV      A,ADDRESS
           ORL      A,#10000000B
           MOVX     @DPTR,A
           MOV      DPL,#02
BU1:       MOVX     A,@DPTR
           ANL      A,#10000000B
           CJNE     A,#00H,BU1
           MOV      DPL,#01H
           MOV      A,PRCHAR
           MOVX     @DPTR,A
           MOV      DPL,#02
BU2:       MOVX     A,@DPTR
           ANL      A,#10000000B
           CJNE     A,#00H,BU2
           POP      DPH
           POP      DPL
           POP      ACC
           POP      PSW
           RET
```

```
;************************************
;*         ADC  CONVERSION          *
;************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!     ANALOG-DİJİTAL ÇEVİRİCİNİN    !
!      DÖNÜŞTÜRME ALTPROGRAMI       !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

ADCCONV:  CLR      P3.3
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          CLR      P3.2
          MOV      R1,#OFFH
          MOVX     A,@R1
          MOV      ANADAT,A
          SETB     P3.2
          NOP
          SETB     P3.3
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          RET
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!  GÖSTERGEYE YAZILACAK KARAKTERLER !
!     VE ADRESLERİNİN BELİRLENDİĞİ   !
!            ALTPROGRAMLAR           !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
YAZK:       MOV       PRCHAR,#4BH
            MOV       ADDRESS,#00
            RET
YAZT:       MOV       PRCHAR,#54H
            MOV       ADDRESS,#00
            RET
YAZP:       MOV       PRCHAR,#0F0H
            MOV       ADDRESS,#01
            RET
YAZI:       MOV       PRCHAR,#69H
            MOV       ADDRESS,#01
            RET
YAZD:       MOV       PRCHAR,#64H
            MOV       ADDRESS,#01
            RET
YAZS:       MOV       PRCHAR,#73H
            MOV       ADDRESS,#01
            RET
YAZESIT:    MOV       PRCHAR,#3DH
            MOV       ADDRESS,#02
            RET
YAZKPO:     MOV       A,KPO
            ADD       A,#30H
            MOV       PRCHAR,A
            MOV       ADDRESS,#03
            RET
YAZKIO:     MOV       A,KIO
            ADD       A,#30H
            MOV       PRCHAR,A
            MOV       ADDRESS,#03
            RET
YAZKDO:     MOV       A,KDO
            ADD       A,#30H
            MOV       PRCHAR,A
            MOV       ADDRESS,#03
            RET
YAZKTO:     MOV       A,KTO
            ADD       A,#30H
            MOV       PRCHAR,A
            MOV       ADDRESS,#03
            RET
YAZKT1:     MOV       A,KT1
            ADD       A,#30H
            MOV       PRCHAR,A
            MOV       ADDRESS,#04
            RET
YAZNOKTT:   MOV       PRCHAR,#2EH
            MOV       ADDRESS,#05
            RET
YAZNOKTA:   MOV       PRCHAR,#2EH
```

```
            MOV     ADDRESS,#04
            RET
YAZKP1:     MOV     A,KP1
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#05
            RET
YAZKI1:     MOV     A,KI1
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#05
            RET
YAZKD1:     MOV     A,KD1
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#05
            RET
YAZKP2:     MOV     A,KP2
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#06
            RET
YAZKI2:     MOV     A,KI2
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#06
            RET
YAZKD2:     MOV     A,KD2
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#06
            RET
YAZKT2:     MOV     A,KT2
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#06
            RET
YAZKP3:     MOV     A,KP3
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#07
            RET
YAZKI3:     MOV     A,KI3
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#07
            RET
YAZKD3:     MOV     A,KD3
            ADD     A,#30H
            MOV     PRCHAR,A
            MOV     ADDRESS,#07
            RET
YAZKT3:     MOV     A,KT3
            ADD     A,#30H
            MOV     PRCHAR,A
```

```
            MOV       ADDRESS,#07
            RET
YAZKT4:     MOV       A,KT4
            ADD       A,#30H
            MOV       PRCHAR,A
            MOV       ADDRESS,#08
            RET

;****************************************
;***       BETWEEN CORRECT VALUES     ***
;****************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!   KONTROLÖR KATSAYILARI VERİLİRKEN   !
!GEREKLİ TUŞLARA BASILIP BASILMADIĞINI!
!     ANLAMAYA YARAYAN ALTPROGRAM      !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

BETWEEN:    CALL      KEYSCAN
            MOV       A,KEYBUF
            CJNE      A,#3EH,A1
            JMP       BETWEEN
A1:         CJNE      A,#3CH,A2
            JMP       BETWEEN
A2:         CJNE      A,#45H,A3
            JMP       BETWEEN
A3:         RET

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! TUŞ TAKIMINDA HANGİ TUŞA BASILDIĞI   !
!         BİLGİSİNİ SAKLAYAN           !
!        KEYBUF  DEĞİŞKENİNİN          !
!    İÇERİĞİNİ TEMİZLEYEN ALTPROGRAM   !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

ADJUST:     MOV       A,KEYBUF
            CLR       A
            MOV       KEYBUF,A
            CALL      DELAY
            CALL      DELAY
            RET

;****************************************
;***              ADDING              ***
;****************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!          TOPLAMA ALTPROGRAMI         !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

ADDING:     MOV       A,OP0
            ADD       A,TKL
            MOV       TKL,A
            MOV       A,OP1
            ADDC      A,TKH
            MOV       TKH,A
            RET
```

```
;*************************************
;***          KSAYI*10000          ***
;*************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!      16 BİTLİK ÇARPMADA KATSAYI    !
!        DEĞİŞKENİNİ 10000,1000      !
!     VE 100 İLE ÇARPMADAN ÖNCE      !
!  DEĞİŞKEN  ATAMA ALTPROGRAMLARI    !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

ATA_10000:MOV       OP3,#00H
          MOV       OP2,#00H
          MOV       OP1,#27H
          MOV       OP0,#10H
          MOV       MUL0,#00H
          MOV       MUL1,KSAYI
          RET

;*************************************
;***          KSAYI*1000           ***
;*************************************
ATA_1000: MOV       OP3,#00H
          MOV       OP2,#00H
          MOV       OP1,#03H
          MOV       OP0,#0E8H
          MOV       MUL0,#00H
          MOV       MUL1,KSAYI
          RET

;*************************************
;***          KSAYI*100            ***
;*************************************
ATA_100:  MOV       OP3,#00H
          MOV       OP2,#00H
          MOV       OP1,#00H
          MOV       OP0,#64H
          MOV       MUL0,#00H
          MOV       MUL1,KSAYI
          RET

;*************************************
;*** SETTING UP  THE KNOWN VALUES ***
;*************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! BİLİNEN PARAMETRELERİ KULLANILAN  !
!    DEĞİŞKENE ATAYAN ALTPROGRAM    !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

KNOWN:    MOV   OP3,#00H
          MOV   OP2,#00H
          MOV   OP1,TKH
          MOV   OP0,TKL
          MOV   MUL0,#00H
          MOV   MUL1,ANADAT_0
          RET
```

```
;*************************************
;**INFORMING THE VALUE OF THOUSAND **
;*************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! BİLİNEN PARAMETRELERİ KULLANILAN  !
!    DEĞİŞKENE ATAYAN ALTPROGRAM    !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

THOUSAND: PUSH      PSW
          PUSH      ACC
          PUSH      DPL
          PUSH      DPH
          MOV       DIVO,#03H
          MOV       DIV1,#0E8H
          POP       DPH
          POP       DPL
          POP       ACC
          POP       PSW
          RET

;*************************************************
;*                 16 BITS MULTIPLICATION        *
;* INPUT : [OP3][OP2][OP1][OPO]*[MULO][MUL1]     *
;* OUTPUT: [OP3][OP2][OP1][OPO]                  *
;*************************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!         16 BİTLİK ÇARPMA ALTPROGRAMI          !
!    *BU PROGRAMIN ALGORİTMASI INTEL YAZILIM    !
!   KÜTÜPHANESİNDEN ALINMIŞ VE ASSEMBLY DİLİYLE !
!            YENİDEN YAZILMIŞTIR*               !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

MUL_16:   MOV       SAB3,#0   ; CLEAR UPPER 16 BITS
          MOV       SAB2,#0   ; GENERATE THE LSB
                                  OF THE RESULT
          MOV       B,OPO
          MOV       A,MUL1
          MUL       AB
          MOV       SABO,A    ; LOW-ORDER RESULT
          MOV       SAB1,B    ; HIGH-ORDER RESULT
          MOV       B,OP1     ; GENERATE NEXT RESULT
          MOV       A,MUL1
          MUL       AB
          ADD       A,SAB1    ; LOW-ORDER RESULT
          MOV       SAB1,A    ; SAVE
          MOV       A,B       ; GET HIGH-ORDER RESULT
          ADDC      A,SAB2    ; CARRY FROM PREVIOUS OP
          MOV       SAB2,A    ; SAVE
          JNC       MUL_LOOP1
          INC       SAB3      ; CARRY INTO SAB3
MUL_LOOP1:MOV       B,OPO
          MOV       A,MULO
          MUL       AB
          ADD       A,SAB1    ; LOW-ORDER RESULT
          MOV       SAB1,A    ; SAVE
```

```
            MOV     A,B         ; GET HIGH-ORDER RESULT
            ADDC    A,SAB2      ; CARRY FROM PREV. OP.
            MOV     SAB2,A      ; SAVE
            JNC     MUL_LOOP2
            INC     SAB3        ; CARRY INTO SAB3
MUL_LOOP2:  MOV     B,OP2
            MOV     A,MUL1
            MUL     AB
            ADD     A,SAB2
            MOV     SAB2,A
            MOV     A,B
            ADDC    A,SAB3
            MOV     SAB3,A
            MOV     B,OP1
            MOV     A,MULO
            MUL     AB
            ADD     A,SAB2
            MOV     SAB2,A
            MOV     A,B
            ADDC    A,SAB3
            MOV     SAB3,A
            MOV     B,OP3
            MOV     A,MUL1
            MUL     AB
            ADD     A,SAB3
            MOV     SAB3,A
            MOV     B,OP2
            MOV     A,MULO
            MUL     AB
            ADD     A,SAB3
            MOV     SAB3,A
            MOV     OPO,SABO
            MOV     OP1,SAB1
            MOV     OP2,SAB2
            MOV     OP3,SAB3
            RET

;*************************************************
;*              16 BITS DIVISION                 *
;*  INPUT : [OP3][OP2][OP1][OPO]/[DIVO][DIV1]    *
;*  OUTPUT : [OP3][OP2][OP1][OPO]                *
;*************************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!          16 BİTLİK BÖLME ALTPROGRAMI           !
!    *BU PROGRAMIN ALGORİTMASI INTEL YAZILIM     !
!   KÜTÜPHANESİNDEN ALINMIŞ VE ASSEMBLY DİLİ İLE !
!              YENİDEN YAZILMIŞTIR *              !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
DIV_16:     MOV     R7,#0
            MOV     R6,#0
            MOV     SABO,#0
            MOV     SAB1,#0
            MOV     SAB2,#0
            MOV     SAB3,#0
            MOV     R1,DIVO
```

```
            MOV     R0,DIV1
            MOV     R5,#32
DIV_LOOP:   CALL    SHIFT_D
            MOV     A,R6
            RLC     A
            MOV     R6,A
            MOV     A,R7
            RLC     A
            MOV     R7,A
            JC      CAN_SUB
            CLR     C
            MOV     A,R7
            SUBB    A,R1
            JC      CANT_SUB
            JNZ     CAN_SUB
            CLR     C
            MOV     A,R6
            SUBB    A,R0
            JC      CANT_SUB
CAN_SUB:    CLR     C
            MOV     A,R6
            SUBB    A,R0
            MOV     R6,A
            MOV     A,R7
            SUBB    A,R1
            MOV     R7,A
            SETB    C
            JMP     QUOT
CANT_SUB:   CLR     C
QUOT:       CALL    SHIFT_Q
            DJNZ    R5,DIV_LOOP
            MOV     OP0,SAB0
            MOV     OP1,SAB1
            MOV     OP2,SAB2
            MOV     OP3,SAB3
            RET

SHIFT_D:    CLR     C
            MOV     A,OP0
            RLC     A
            MOV     OP0,A
            MOV     A,OP1
            RLC     A
            MOV     OP1,A
            MOV     A,OP2
            RLC     A
            MOV     OP2,A
            MOV     A,OP3
            RLC     A
            MOV     OP3,A
            RET
SHIFT_Q:    MOV     A,SAB0
            RLC     A
            MOV     SAB0,A
            MOV     A,SAB1
```

```
            RLC     A
            MOV     SAB1,A
            MOV     A,SAB2
            RLC     A
            MOV     SAB2,A
            MOV     A,SAB3
            RLC     A
            MOV     SAB3,A
            RET

;*****************************************************
;*                  32 BITS ADDITION                 *
;*INPUT[OP3][OP2][OP1][OPO]+[ADO][AD1][AD2][AD3][AD4]*
;*          OUTPUT:[OP3][OP2][OP1][OPO]              *
;*****************************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!           32 BİTLİK TOPLAMA ALTPROGRAMI           !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

ADD_32:     CLR     C
            MOV     A,OPO
            ADDC    A,AD3
            MOV     OPO,A
            MOV     A,OP1
            ADDC    A,AD2
            MOV     OP1,A
            MOV     A,OP2
            ADDC    A,AD1
            MOV     OP2,A
            MOV     A,OP3
            ADDC    A,ADO
            MOV     OP3,A
            RET
;*****************************************************
;*          BINARY TO BCD CONVERSION                 *
;*          INPUT : [OP2][OP1][OPO]                  *
;*         OUTPUT : 72H..70H  PACKED                 *
;*            R7..R2    UNPACKED                     *
;*****************************************************
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!     BINARY OLARAK VERİLEN DEĞERİ BCD'YE ÇEVİREN   !
!                   ALTPROGRAM                      !
!       *BU PROGRAMIN ALGORİTMASI INTEL YAZILIM     !
!   KÜTÜPHANESİNDEN ALINMIŞ VE ASSEMBLY DİLİ İLE    !
!               YENİDEN YAZILMIŞTIR *               !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

BINBCDCON:  MOV     R2,OP1
            MOV     A,OPO
            MOV     RO,#7OH
            MOV     R6,OP2
            XCH     A,RO
            MOV     R1,A
            XCH     A,RO
            MOV     R4,#03
```

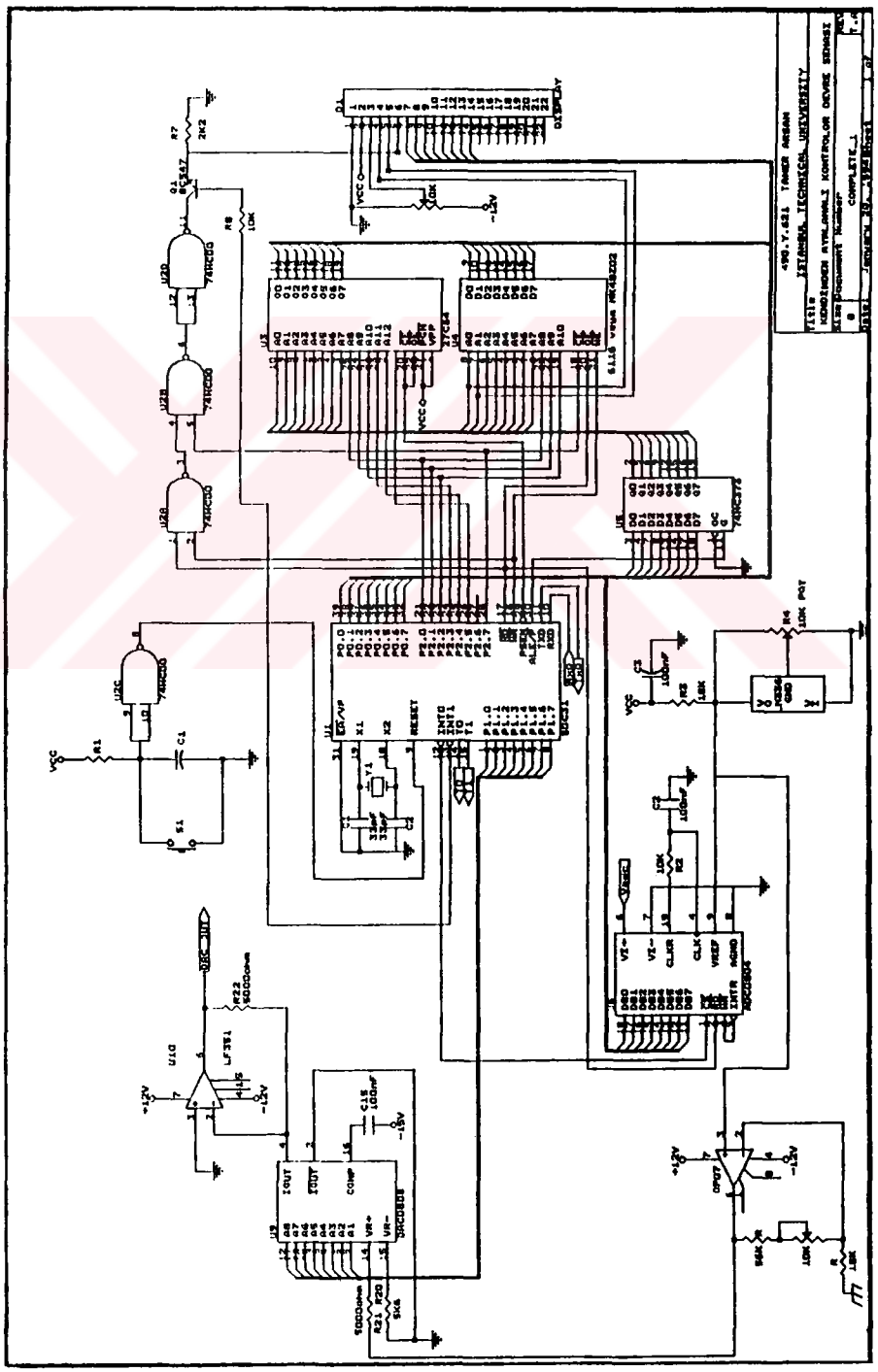
```
BCDCOA:   MOV     @R1,#00
          INC     R1
          DJNZ    R4,BCDCOA
          MOV     R3,#24
BCDCOB:   CLR     C           ; BIN=BIN*2
          RLC     A
          XCH     A,R2
          RLC     A
          XCH     A,R2
          XCH     A,R6
          RLC     A
          XCH     A,R6
          XCH     A,R0        ; BCD=BCD*2+CARRY
          MOV     R1,A
          XCH     A,R0
          MOV     R4,#03
          MOV     R5,A
BCDOC:    MOV     A,@R1
          ADDC    A,@R1
          DA      A
          MOV     @R1,A
          INC     R1
          DJNZ    R4,BCDOC
          MOV     A,R5
          JC      BCDCOD
          DJNZ    R3,BCDCOB ; IF PRECISION <>16 BACK
          CLR     C
BCDCOD:   MOV     A,70H       ; UPCKG LAST 5 DIGITS
          ANL     A,#00001111B
          MOV     R2,A
          MOV     A,70H
          ANL     A,#11110000B
          SWAP    A
          MOV     R3,A
          MOV     A,71H
          ANL     A,#00001111B
          MOV     R4,A
          MOV     A,71H
          ANL     A,#11110000B
          SWAP    A
          MOV     R5,A
          MOV     A,72H
          ANL     A,#00001111B
          MOV     R6,A
          MOV     A,72H
          ANL     A,#11110000B
          SWAP    A
          MOV     R7,A
          RET
```

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!    AÇILIŞ LOGOSUNUN BELİRLENMESİ    !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

          ORG       1900H
          DB        '*  AUTOTUNING  *.'
          DB        '*PID CONTROLLER*!'

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!       TUŞ TAKIMINDAKİ TUŞLARIN      !
!       KARŞILIKLARININ ATANMASI      !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

           ORG      2000H
           DB       32H
           DB       3FH
           DB       35H
           DB       4DH
           DB       38H
           DB       41H
           DB       30H
           DB       3FH
           DB       53H
           DB       45H
           DB       00H
           DB       00H
           DB       00H
           DB       00H
           DB       00H
           DB       00H
           DB       31H
           DB       33H
           DB       34H
           DB       36H
           DB       37H
           DB       39H
           DB       3EH
           DB       3CH
           DB       3FH
           DB       52H

           END
```

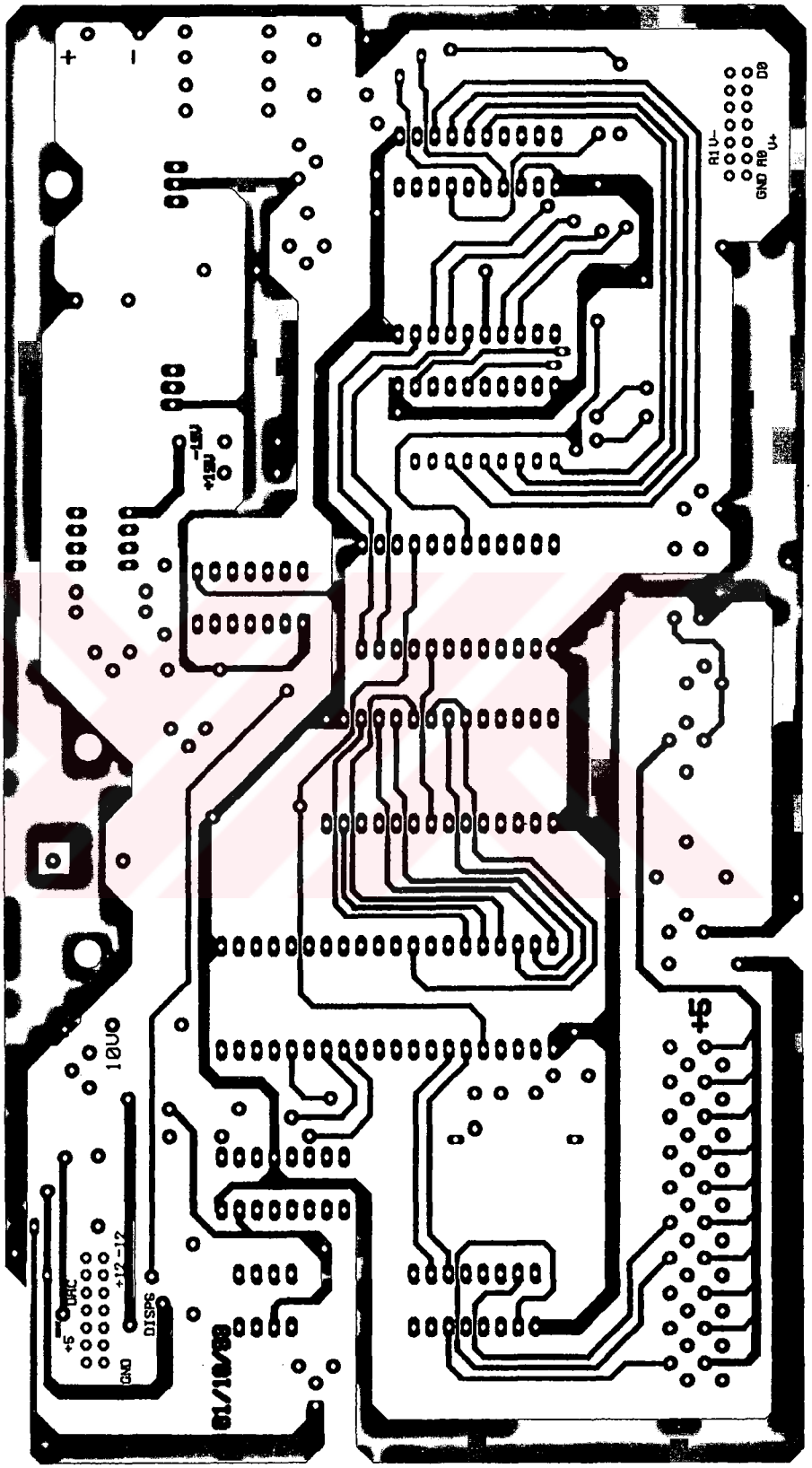
T.C. YÜKSEKÖĞRETİM KURULU
DOKÜMANTASYON MERKEZİ

EK B

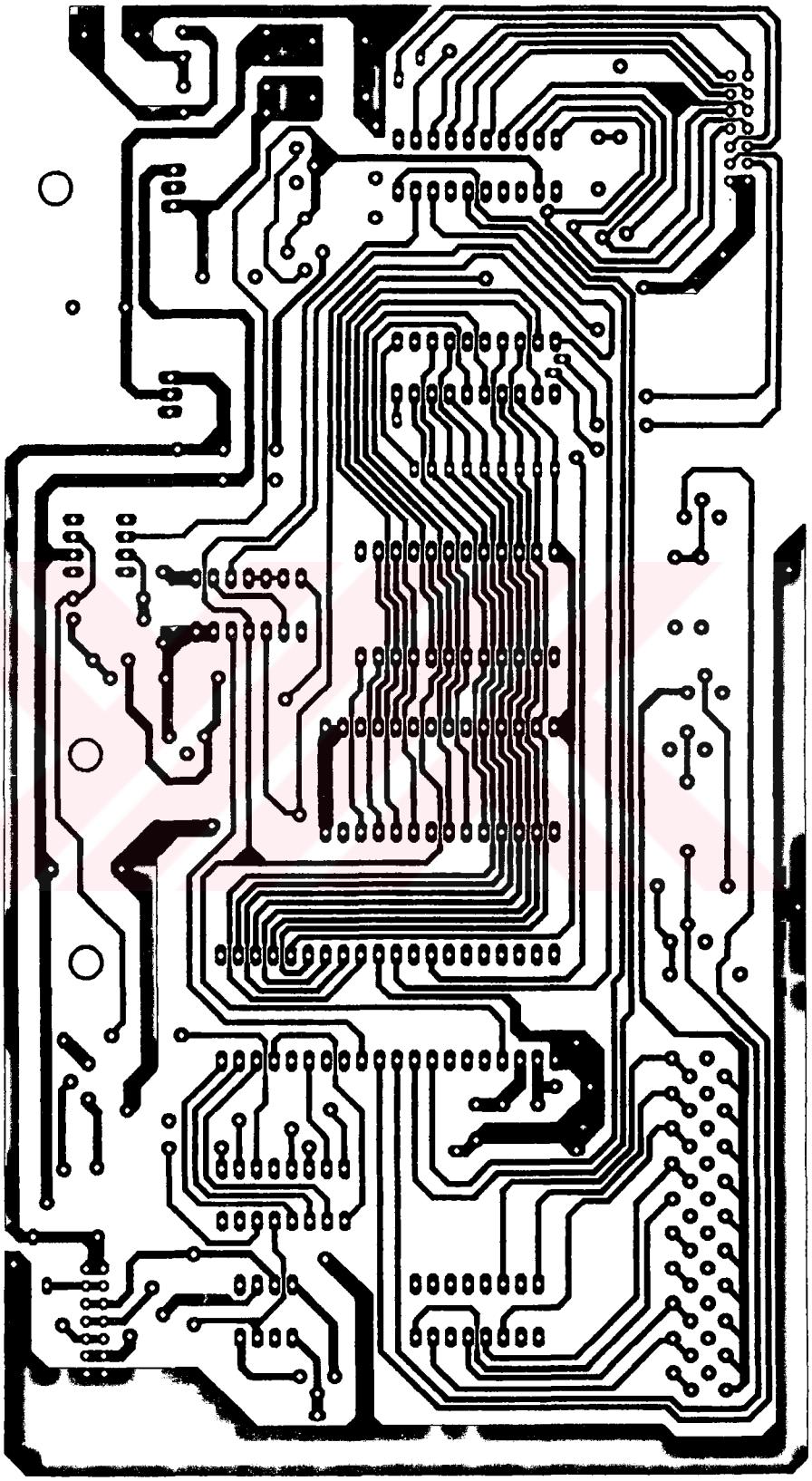
GERÇEKLENEN MİKROKONTROLÖR KARTININ DEVRE ŞEMASI VE BASKILI DEVRE YERLEŞİM DÜZENLERİ



Şekil B.1 - Kendinden Ayarlamalı Kontrolör Devre Şeması.



Şekil B.2 - Gerçeklenen karta ait baskılı devre plaketinin eleman yüzü.



Şekil B.3 - Gerçeklenen karta ait baskılı devre plaketinin lehim yüzü.

Şekil B.4-Gerçeklenen karta ait baskılı devre plaketinde elemanların yerleşimi.

T.C. YÜKSEKÖĞRETİM KURULU
DOKÜMANTASYON MERKEZİ

ÖZGEÇMİŞ

Taner ARSAN, 1968 yılında Erzincan'da doğdu. İlk-öğrenimini Balıkesir'de, ortaöğrenimini İstanbul Gazi Osman Paşa Ortaokulunda tamamladı. 1985 yılında İstanbul Kabataş Erkek Lisesinden mezun oldu ve aynı yıl İstanbul Teknik Üniversitesi, Elektrik-Elektronik Fakültesi, Elektronik ve Haberleşme Mühendisliği bölümünü kazandı. 1990 yılında bu bölümden mezun olup, İ.T.Ü, Fen Bilimleri Enstitüsü, Kontrol ve Bilgisayar Mühendisliği Ana Bilim Dalı, Kontrol ve Bilgisayar Mühendisliği programında Yüksek Lisans öğrenimine başladı. 1991 yılı Temmuz ayında İ.T.Ü Elektrik-Elektronik Fakültesi Kontrol ve Bilgisayar Mühendisliği Bölümü, Kontrol ve Kumanda Sistemleri Ana Bilim Dalında Araştırma Görevlisi oldu. Halen bu bölümde görevini sürdürmektedir.