Tezin Başlığı: **PIC Mikrodenetleyici ile Kapalı Ortamda Sıcaklık ve Nem Kontrolü**

Tezi Hazırlayan: **Nihat YÜKLÜ**

Sınav Tarihi: 20 Ekim 2006

Yukarıda adı geçen tez jürimizce değerlendirilerek Elektrik Elektronik Mühendisliği Anabilim Dalında Yüksek Lisans Tezi olarak kabul edilmiştir.

Sınav Jürisi Üyeleri

Doç. Dr. Nusret TAN (İnönü Üniversitesi) ……

Prof. Dr Hafız ALİSOY (İnönü Üniversitesi) ……

Yrd. Doç. Dr. Müslüm ARKAN (İnönü Üniversitesi) ……

……

Prof. Dr Hafız ALİSOY
Tez Danışmanı

İnönü Üniversitesi Fen Bilimleri Enstitüsü Onayı

……

Prof. Dr. Ali ŞAHİN

Enstitü Müdürü

TC
İNÖNÜ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

PIC MİKRODENETLEYİCİ İLE KAPALI ORTAMDA SICAKLIK VE NEM KONTROLÜ

Nihat YÜKLÜ

YÜKSEK LİSANS TEZİ
ELEKTRİK ELEKTRONİK ANABİLİM DALI

MALATYA
2006

ÖZET

Yüksek Lisans Tezi

PIC MİKRODENETLEYİCİ İLE KAPALI ORTAMDA SICAKLIK VE NEM KONTROLÜ

Nihat Yüklü

İnönü Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik – Elektronik Anabilim Dalı

133+xi sayfa
2006

Danışman: Prof. Dr. Hafız Alisoy

Mikroelektronik teknolojisindeki son gelişmelerden dolayı mikrodenetleyiciler endüstride kontrol amaçlı olarak yaygın olarak kullanılmaktadır. Mikrodenetleyicilerin kullanım alanları içerisinde ev ve ofis makinaları ile endüstride sistemlerin kontrol işlemlerini gösterebiliriz.

PIC mikrodenetleyiciler programlamada RISC teknolojisi adı verilen bir yöntem kullanır böylece az sayıda komuta sahip olmalarına rağmen, hızlı ve programlama esnekliği sağlarlar. Proseslerin kontrolü mikrodenetleyicilerle yapıldığında, yapılan işlerdeki proses değişkenleri üzerinde işlem yapabilme olanağı ve yapılan işlemi PIC programına bağlı olarak değiştirebilme avantajı sağlarlar.

Bu tezin amacı kapalı bir ortamda sıcaklık ve nemin istek doğrultusunda değişimini gerçekleştiren kontrolü PIC mikrodenetleyici kullanarak gerçekleştirmektir. Bu amaca ulaşmak için sıcaklık ve nem sensörleri incelendi. Sıcaklık ve nemin kontrolü işleminde bu değerleri dijital olarak ölçmek ve kontrol etmek için PIC mikrodenetleyicisinin kullanıldığı donanımlar geliştirildi. Sistemin kontrolünü sağlamak için de çeşitli yazılımlar geliştirildi.

Sıcaklık ve nemi algılayan sensörler sıcaklık veya nemin istenilen değerlerde olup olmadıklarını belirlemek için kullanılmıştır. Elektronik teknolojisindeki son gelişmeleri göz önüne alarak sıcaklığı ölçmek için LM35 entegresinin kullanıldığı bir dijital termometre geliştirildi. Kapalı ortamdaki sıcaklık ve nemi kontrol etmek için bu iki işlem değişkenini algılayan SHT11 sensörü PIC mikrodenetleyicisi ile birlikte deneysel kontrol işleminde kullanıldı. Kontrol işlemi geri beslemeli kapalı döngü kontrol sistemi ve ON/OFF kontrol olarak yapıldı.

Yapılan kontrol işlemindeki programda sıcaklık ve nem ölçülmekte, istenilen değerlere ulaşmak için ısıtma ve nemlendirme işlemi PIC mikroişlemcisine yazılan programa bağlı olarak gerçekleştirilmektedir.

Gerçekleştirilen dijital termometrenin simülasyon çalışması ISIS programı ile yapıldı. Çalışmalarda geliştirilen uygulamaların pratik uygulamalarda ve bu yöndeki çalışmalarda kullanılma potansiyeli vardır.

ANAHTAR KELİMELER: Mikrodenetleyici sistem, PIC, Sıcaklık, Nem, Mikrodenetleyici kontrol, donanım, yazılım, sıcaklık değişimi, nem değişimi, sensör.

ABSTRACT
M.S. Thesis

TEMPERATURE AND HUMIDITY CONTROL WITH PIC MICROCONTROLLER IN A CLOSED ENVIRONMENT

Nihat Yüklü

İnönü University
Graduate School of Natural and Applied Sciences
Department of Electrical and Electronics Engineering

133+xi pages

2006

Supervisor: Prof.Dr. Hafız Alisoy

Microcontrollers are used extensively in the industry for the control purposes due to the recent development in microelectronic technology. We can denote within the microprocessor usage area home and office machines with control processing of industrial systems.

PIC (Peripheral Interface Controller) microcontrollers are used programming method called RISC (Reduced Instruction Set Computer) technology. Although they have a little number of commands, they provide speed and programming elasticity. Controlling the processes with microcontrollers provides calculation ability on process variables and gives advantage of changing the processes depending on the software of the PIC.

The aim of this thesis is to control the change of temperature and humidity in a closed environment within the desired direction using the PIC microcontrollers. Temperature and humidity sensors are investigated to approach this purpose. In the temperature and humidity control processing to measure and control these values as digital, hardware are developed by using the PIC microcontrollers. Also to provide the system control various software are developed.

Sensors which sense the temperature and humidity are used to determine whether the temperature or humidity is at the desired value or not. Investigating the recent development of electronic technology to measure the temperature, a digital termometer is developed using the LM35 integrated circuit. Sensing these two process variables to control the temprature and humidity in a closed environment at the experimental control process, SHT11 sensor is used together with PIC microcontroller. Control processing is done via closed loop control with feedback and ON/OFF control.

In the control processing program which is done temperature and humidity is measured and to approach the desired values heating and humidifying process is realized depending on the program which is written on the PIC microcontroller.

Simulation studies of the realized digital termometer are done on the ISIS program. The applications developed within this research study have potential features for practical applications and for future work within this direction.

KEY WORDS: Microcontroller system, PIC, Temperature, Humidity, Microcontroller control, hardware, software, temperature change, humidity change, sensor.

TEŞEKKÜR

Mikrodenetleyici kontrollü sıcaklık ve nemin kontrolü adı altında uygulamaya yönelik bir tez konusu almamı sağlayan ve tez çalışmamda bana konu ile ilgili kontrol işleminde, işlemin gerçekleştirilmesi için gerekli adımları ve işlemin kontrolü için olası durumları göz önüne alarak kontrol algoritmasının geliştirilmesinde bana yardımcı olan tez danışmanım Prof. Dr. Hafız ALİSOY' a yardımlarından ve katkılarından dolayı teşekkürü bir borç bilirim.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

SİMGELER DİZİNİ

$r(t)$	Zaman domeni giriş sinyali
$y(t)$	Zaman domeni çıkış sinyali
$g(t)$	Zaman domeni sistem gösterimi
$r(s)$	Laplace domeni giriş sinyali
$y(s)$	Laplace domeni çıkış sinyali
$G(s)$	Laplace domeni sistem transfer fonksiyonu
$C(s)$	Kontrolörün Laplace domeni gösterimi
A_R	çatının alanı = $9m^2$
A_{w1}	duvarın alanı (Doğu, Batı) = $9m^2$
A_{w2}	duvarın alanı (Güney, Kuzey) = $9m^2$
C_{ah}	Hava işleme ünitesinin toplam ısıl kapasitansı = 4.5 kJ/C
C_d	kanal mazemesinin özgül ısısı = 4187 kJ/kg ^{0}C
C_h	nemlendiricinin toplam ısıl kapasitansı = 0.63 kJ/^{0}C
C_{pa}	Havanın özgül ısısı = 1.005 kJ/kg ^{0}C
C_{pw}	Suyun özgül ısısı = 4.1868 kJ/kg ^{0}C
C_R	Çatının toplam ısıl kapasitansı = 80 kJ/C
C_{w1}	duvarın toplam ısıl kapasitansı (Doğu, Batı)=70 kJ/C
C_{w2}	duvarın toplam ısıl kapasitansı (Güney, Kuzey) = 60 kJ/C
C_z	bölgenin toplam ısıl kapasitansı = 47.1 kJ/C
$e(t)$	hata
f_{sa}	hava kaynağının hacimsel akış hızı = 0.192 m3/s
f_{sw}	suyun akış hızı = $8.02X10^{-5}$ m3/s
$h(t)$	nemendiricinin üretmiş olduğu nemli hava oranı
hi	kanal içerisindeki ısı transfer katsayısı =8.33 W/m^2 ^{o}C
ho	ortamın ısı transfer katsayısı =16.6 W/m^2 ^{o}C
M_d	kanal modelinin kütlesi = 6.404 kg/m
m_a	hava akışının kütle akış hızı = 0.24 kg/s
m_m	karışım havasının toplam kütle akış hızı = 0.24 kg/s
m_o	dış kısımdaki havanın kütle akış hızı = 0.12 kg/s
m_r	tekrar hesaplanan havanın kütle akış hızı = 0.12 kg/s
m_t	tüp materyalinin kütlesi kg/m
$P(t)$	işgal edenlerin buharlaşma hızı = 0.08 kg/h
$q(t)$	işgal edenler ve ışıktan dolayı oluşan ısı kazancı (W)
T_{co}	bobinden sonraki havanın sıcaklığı (^{o}C)
T_h	sağlanan havanın sıcaklığı (nemlendirici içerisindeki) (^{o}C)
T_{in}	kanal içerisine verilen sıcaklık
T_m	karışım kutusunun dışında havanın sıcaklığı (^{o}C)
T_{me}	ölçülen sıcaklık (^{o}C)
T_o	dış kısımdaki sıcaklık = 32 ^{o}C (Yaz) = 5 ^{o}C (Kış)
T_{out}	kanaldan sonraki sıcaklık
T_r	tekrar hesaplanan havanın sıcaklığı (^{o}C)
T_s	ısıtma bobininin sağladığı sıcaklık
T_{sa}	hava kaynağının sıcaklığı (^{o}C)
T_{se}	sensördeki çıkış sıcaklığı (^{o}C)
T_{si}	hava kaynağının sıcaklığı (nemlendiriciye doğru) (^{o}C)
$T_{t,o}$	tüpün yüzey sıcaklığı (^{o}C)
T_{wo}	dönüş suyunun sıcaklığı = 10 (^{o}C)
T_{wi}	su kaynağının sıcaklığı (^{o}C)

T_{w1} duvarın sıcaklığı (Doğu, Batı) (^oC)
T_{w1} duvarın sıcaklığı (Güney, Kuzey) (^oC)
T_z bölgenin sıcaklığı (^oC)
U_{w1} duvarların toplam ısı transfer katsayısı (Doğu, Batı)= 2 $W/m^2\ ^oC$
U_{w2} duvarların toplam ısı transfer katsayısı (Güney, Kuzey)= 2 $W/m^2\ ^oC$
U_R çatının toplam ısı transfer katsayısı = 1 $W/m^2\ ^oC$
$(UA)_{ah}$ hava işleme ünitesinin toplam iletim alanı faktörü = 0.04 kJ/sC
V_a hava işleme ünitesinin hacmi = 2.88 m^3
V_h nemlendiricinin hacmi = 1.44 m^3
V_z bölgenin hacmi = 36 m^3
W_{co} bobinden çıkan havanın nem oranı (kg/kg kuru hava)
W_h hava kaynağının nem oranı (nemlendiricide) kg/kg(kuru hava)
W_m karışım kutusundan ayrılan havanın nem oranı (kg/kg kuru hava)
W_o dışarıdaki nem oranı = 0.02744 kg/kg (kuru hava) (Yaz) = 0.002 kg/kg (kuru hava)(Kış)
W_{sa} hava kaynağının nem oranı kg/kg (kuru hava)
W_{si} hava kaynağının nem oranı (nemlendiriciye) kg/kg (kuru hava)
W_z bölgenin nem oranı kg/kg (kuru hava)

Alt indisler

a hava
ah hava işleme ünitesi
ai içeri doğru hava
ao dışarı doğru hava
co bobinden dışarı
d kanal
h nemlendirici
in içeri
m karışmış
me ölçülen
o dış
R çatı
r tekrar dönüşümde olan
s kaynak
sa hava kaynağı
se sensör
sw su kaynağı
w su
w_1 Doğu ve Batı duvarları
w_2 Kuzey ve Güney duvarları
w_i su içeri doğru
w_o su dışarı doğru
z bölge

Latin harfleri
$\alpha_h(UA)_h$ nemlendiricinin toplam iletim alanı faktörü = 0.0183 kJ/s ^oC
t_{se} sensörün zaman sabiti (saniye)
r_a havanın yoğunluğu = 1.25 kg/m^3
r_w suyun yoğunluğu = 998 kg/m^3

1. GİRİŞ

Sıcaklık ve nem işlem değişkeni olarak endüstriyel kontrolde sıkça karşılaşılan, imalat işleminde çoğunlukla kontrolü zorunlu olan parametrelerdir. Son zamanlarda kontrol stratejileri enerji tüketiminin azaltılması ve en elverişli ortam koşullarının belirlenmesi ve bu koşullara karşılık gelen dinamik modellerin tasarımı ve optimizasyonu güncel problemler arasında yer almaktadır.

Bir HVAC sisteminin dinamik modelinin türetilmesindeki işlemler Tashtoush vd. [1] tarafından yapılan çalışmada anlatılmaktadır. Bu çalışmada HVAC sistem bir kapalı bölge, ısıtıcı bobin, soğutucu ve nem alıcı bobin, nemlendirici, kanal, ve vantilatör içermektedir.Yapılan çalışmada özellikle ilgi iç kısımdaki ortamın kalitesinin iyileştirilmesi ve enerji tüketiminin azaltılması üzerine kontrol stratejileri geliştirmekte odaklanmıştır. Dinamik model özellikle HVAC sistemlerde dinamik karakteristiklerin bilindiği kontrol sistemleri için faydalıdır.

Isı kontrolünde sensör pozisyonunun etkisini göstermek için Riederer vd. [3] bir oda modeli geliştirmiştir. Bir oda sıcaklık kontrol edicisinin sensörü tarafından ölçülen sıcaklık onun oda içerisindeki pozisyonuna bağlıdır. Oda içerisindeki şartlar çok homojen değilse sensörlerin ölçmüş oldukları sıcaklıklar " ortalama hava sıcaklığından " farklı olacaktır. Bu çalışmada yapılan analiz ile oda modelleri geliştirmek için detaylı bir kriter listesi oluşturulmuştur. Bu kriterler yeni oda modelleri geliştirmek için kullanılmıştır.

Odanın iç kısmındaki sıcaklık dağılımı ve hava akımı odadaki ısıl konforun bir yerden diğer bir yere göre değişimine sebep olduğu Peng ve Paassen [4] tarafından belirtilmiştir. Odadaki eşyaların bulunduğu ve çalıştığı bir bölgede ısıl şartların önceden belirlenmesi ve daha iyi kontrolü için iç kısımdaki sıcaklık dağılımı dinamiğinin detaylı ve hızlı bir modeline gereksinim vardır, Peng ve Paassen [4] hava akımının kompleksliğinden dolayı çok az bir makalede bu tip modeller üzerinde çalışıldığını vurgulayarak, bu makalede hesaplanabilir akışkan dinamiğinden türetilen bölge modellerini tartışmaktadır. Model deneysel sonuçlarla geçerlilik kazanmıştır. Kontrol sistemlerini daha iyi tasarlamak için, bölge modeli durum uzay gösterimi formuna dönüştürülmüştür. Durum uzay modeli ile sıcaklığın önceden belirlenmesi ve daha kesin bir şekilde kontrolünün yapıldığı bir örnekte verilmiştir.

Fiziksel kanunlarla bir hava şartlandırma odasının dinamik modelini türetmek için Kasahara vd. [5] gerekli işlemleri tanımlamak için kontrol edilen odayı beş bölgeye

bölmüştür. Her bir bölgenin dinamiği toplanmış kapasite modeli ile açıklanmaktadır, toplam olarak 15 lineer diferansiyel denklem elde edilmiştir. Bu işlemden türetilen model parametreler, odanın tüm ısı transferi katsayıları ile nümerik olarak ilişkilendirilerek, oda ile ilgili önemli değişik zaman sabitleri de birleştirilmiştir. Sistemin zaman sabiti ve kazanç sabitlerinin tahmin edilmesiyle oransal - artı – integral (PI) kontrol ediciler tasarlanmıştır. Odanın iç kısmındaki sıcaklık ve nem değeri bir hava işleme ünitesinde PI kontrol kullanılarak set noktası değerleri sağlanmıştır.

Binayı oluşturan elemanların ve ısıtma, havalandırma ve hava şartlandırma cihazlarının yapısından kaynaklanan performans sınırlamaları, binaların enerji verimli çalışmasını iyileştirmek için çalışma stratejilerinin incelenmesini gerekli kılmaktadır. Elektronik kontrol edicilerin montajının kolay ve üstün yetenekleri, bununla birlikte bu kontrol edicilerin programlanabilir olmasını Morteza vd. [6] özellikle ilgi çekici bulmaktadır, çalışmasında eski tip ve yeni tip binaların ekonomik olarak en verimli çalışma stratejilerini belirlemiştir.

Bütün ısıtma, havalandırma ve hava şartlandırma sistemlerinin evrensel optimizasyonu Lu Lu vd. [7] tarafından sunulmuştur. Evrensel optimizasyonun nesnel fonksiyonu ve sınırlamaları ana bileşenlerin matematiksel modellerine bağlı olarak bu çalışmada formüle edilmiştir. Bu modellerin tamamı güç tüketim elemanları ve ısı değiştiricileri ile ilişkilendirilmiştir. Bütün ana bileşenlerin karakteristikleri modellerle kabaca sunularak, problemin karmaşıklığını göstermek için birbirleriyle olan etkileşim analiz edilerek tartışılmıştır. Çalışan parçaların karakteristiklerinden dolayı bütün HVAC sistemlerin karmaşık başlangıç optimizasyon problemleri, optimizasyona hazır olması için basitleştirilmiş birleşik bir forma dönüştürülmüştür.

İç kısmında hava ve soğutucu su döngüleri bulunan merkezi bir ısıtma, havalandırma ve hava şartlandırma sisteminin (HVAC) bina içindeki kısmını optimize etmek için gerekli pratik metotların sunumu Lu Lu vd. [8] tarafından çalışılmıştır. Bu işlem için öncelikle elemanların karakteristik analizinden gidilerek, soğutucu yüklerin, ısı dönüştürücülerin enerji tüketimlerinin ve enerji tüketici elemanların matematiksel modelleri oluşturulmuştur. Bu makalenin en önemli özelliği her bir bileşen için olmayıp, sistemin tamamı için geçerli olan enerji tüketiminin optimizasyonu sistematik bir yaklaşımla ele alınmış olmasıdır.

Parametreleri çalışma şartları ile değişen bir seranın sıcaklık kontrolü için yeni bir uygulamalı teknik K.G. Arvanitis vd. [9] tarafından önerilmiştir. Simülasyon kullanarak

HVAC sistemlerin enerji analizi ve performans geçerliliği [10] Tim Salsbury, Rick Diamond tarafından incelenmiştir.

PID (oransal – integral – türevsel) kontrolediciler yalnızca üç parametreye sahip olmalarına karşılık, Skogestad [11] sistematik bir işlem olmaksızın, bu parametreler için iyi değerler (ayar değerleri) bulmanın zor olduğunu belirterek, gerçekte de bir fabrika ziyaret edildiğinde PID kontroledicilerin büyük çoğunluğunun oldukça zayıf bir şekilde ayarlandığını (kontrol ettiğini) ifade etmektedir.

Fanger [12] bina iç kısımlarındaki hava kalitesinin daha iyi oluşunun üretimi artıracağını belirterek, ısıl ortamın bireysel kontrol olarak da sağlanması gerektiğini bu nedenle binaların yapım aşamasında bu hususun dikkate alınmasını önermektedir.

1.1 Tezin Amacı

İmalat teknolojisinde kapalı bir ortamda sıkça karşılaşılan sıcaklık ve nem parametrelerinin nominal teknolojik işlem taleplerine uygun değişimini gerçekleştiren PIC mikrodenetleyici tabanlı ünitenin hazırlanmasıdır.

1.2 Tezin Kapsamı

Tez altı bölüm, sonuç, kaynaklar, 27 adet şekiller dizini, 3 adet çizelgeler dizini ve deneysel çalışma için yazılan program ekinden olup, toplam sayfadan ibarettir.
Tezin giriş bölümünde konunun güncelliği ile ilgili son yapılan literatür çalışmalarının analizi verilerek, tezin amacı ve bu amaç doğrultusunda yapılması öngörülen problemler belirlenir.

Tezin ikinci bölümünde mikrodenetleyicilere giriş yapılarak, günümüzde eğitim amaçlı geniş kullanım alanına sahip PIC 16F84 mikrodenetleyicisine ilişkin temel kavramlar verilir. PIC mikrodenetleyicilerinin tercih nedenleri açıklanır, PIC' in kullanımı için gerekli aşamalar açıklanarak, PIC mikrodenetleyicilerinin donanımsal incelemesi yapılır, ayrıca bu bölümde PIC mikrodenetleyicilerin iç yapısı incelenerek, Program hafıza organizasyonu ve data hafıza organizasyonu açıklanarak, mikodenetleyici içerisindeki INTCON kaydedicisi ve durum kaydedicisi ile ilgili teorik bilgiler verilmiştir. PIC için yazılan programlarda kullanılan kesme ve yığın kavramları açıklanarak, CPU' nun diğer özellikleri olan osilatör tipleri, reset, watchdog zamanlayıcısı, uyku modu, uykudan uyandırma gibi kavramlar açıklanmıştır.

Tezin üçüncü bölümünde sıcaklık değişkeninin ölçülmesi için sıcaklıkla ilgili sensörler incelenmiştir. Bu amaçla dokunma yoluyla sıcaklığın ölçülmesi ve dokunmaksızın sıcaklığın ölçülmesi incelenerek sensör tipleri ve teknolojileri araştırılmıştır. sensör tipleri olarak elektromekanik sensörler, bi - metal termostatlar, elektronik silikon sensörler, thermokupıllar, thermistörler ve RTD' ler hakkında bilgi verilmiştir. Dijital termometrenin deneysel çalışmasında katı hal yarıiletken teknolojisi ile üretilen LM35 santigrad doğruluklu sıcaklık sensörü kullanıldığından bu sensörün özellikleri, tipik uygulamaları, bağlantı diyagramları ve tipik performans karakteristiği verilerek bu sensörün kullanıldığı uygulamalar belirtilmiştir.

Tezin dördüncü bölümünde nemin ne olduğu, niçin önemli olduğu nem sensörü tipleri ve teknolojileri açıklanmıştır. Nem sensörleri ile ilgili gerekli bilgileri vermek amacı ile kapasitif RH sensörler, rezistif nem sensörleri, ısıl iletkenliğe bağlı olarak nem ölçen sensörler hakkında bilgi verilmiştir. Deneysel çalışmaya yardımcı olacak nem sensörlerinin seçimi ve belirlenmesi konusu araştırılarak, sıcaklığın nem üzerine etkisi incelenmiştir. Deneysel çalışmada nem sensörü olarak SHT11 sensörü kullanıldığından bu sensor hakkında özet bilgi, sensörün uygulamaları, blok diyagramı, seri ara birimlemesi, sensor üzerindeki seri saat sinyali girişi, seri veri haberleşmeyi sağlayan veri hattı açıklanmıştır. Sensörden programlama ile bir komutun gönderilmesi bilgi olarak verilerek, sıcaklığın sensör üzerindeki etkisi belirtilmiştir.

Tezin beşinci bölümünde kontrol sistemi incelenerek, günümüzde bilgisayar teknolojisinin gelişimi, mikroelektronik teknolojisindeki gelişmelere paralel olarak mikrodenetleyici kontrollü sistemlerin yaygın olarak kullanımı dolayısıyla bilgisayarlı kontrol sistemi açıklanmıştır. Kontrol sistemlerinin tanımlanmasında kullanılan bir kontrol sistemindeki değişken isimleri açıklanmıştır. Kontrol sistemlerinin sınıflandırılması yapılarak, kontrol sistemlerinin blok diyagramlarla gösterimi ve transfer fonksiyonlarının hesaplanması bilgileri verilmiştir. Deneysel kontrol işlemi geri beslemeli kontrol sistemi ile yapılacağından geri beslemeli kontrol sistemi tanıtılmıştır. Kontrol sistemleri geri beslemeye bağlı olarak iki guruba ayrıldığından bunlardan açık döngü kontrol ve kapalı döngü kontrol kavramları açıklanarak, kapalı döngü kontrolde negatif ve pozitif geri beslemenin çıkışa etkisi ve transfer fonksiyonları incelenmiş, otomatik kontrolün faydaları belirtilmiştir.

Tezin altıncı bölümünde bu bölüme kadar yapılan çalışmaları desteklemek amacı ile PIC 16F877 entegresinin ve LM35 sıcaklık sensörünün kullanıldığı deneysel bir dijital termometre devresi verilerek, bu devrenin ISIS' te simülasyonu yapılmıştır. Devrenin

simülasyonundan alınan görüntüler şekil olarak eklenmiştir. Teorik olarak yapılan tez çalışmasını desteklemek amacı ile sıcaklık ve nem sensörü olarak kullanılan SHT11 entegresi ve PIC 16F877 entegresi kullanılarak, kapalı bir ortamda sıcaklık ve nem değişimini istek doğrultusunda kontrol eden bir kontrol ünitesi deneysel olarak hazırlanmıştır.

Tezin sonuç kısmında mevcut çalışmalarda sıcaklık ve nemin ölçümü ile kontrolünün analog yapılmasına karşılık, tez çalışmasında sıcaklık ve nemin yarı iletken teknolojisindeki gelişmelerin dikkate alınarak SHT11 entegresi kullanılarak tek bir yongayla ölçme işleminin gerçekleştirildiği ve kontrol işleminin de dijital mikrodenetleyici kontrollü olarak yapıldığı belirtilip, programlanabilir kontrolün avantajları, yetenekleri ve elastikiyeti açıklanmıştır.

Tezin kaynaklar kısmında tez konusunun güncelliği ile ilgili literatür çalışmalarında yaralanılan makaleler ve tezin hazırlanmasında başvurulan kitap ve dokümanlar, tezde yapılan atıf önceliğine bağlı olarak sıralanarak verilmiştir.

Tezin özgeçmiş bölümünde ise eğitim ve mesleki hayatımla ilgili olarak kısa özgeçmişim verilmiştir.

2. MİKRODENETLEYİCİLER

2.1 Giriş

Günümüzde hızla gelişen teknoloji bilgisayarlarla kontrol edilen cihazları bizlere çok yaklaştırdı. Öyle ki, cebimizde taşıdığımız telefon, büromuzdaki faks makinesi, evimizdeki çamaşır makinesi gibi cihazlar artık çok küçük, adına mikrodenetleyici denilen elektronik elemanlar yardımıyla kontrol edilir hale geldi. Aynı şeyler, 1990 öncesinde mikroişlemci (CPU) kontrol elemanlarıyla yapılmaktaydı. CPU ile bir cihazı kontrol etmek için Giriş / Çıkış (I/O) elemanları, RAM gibi ek devreler gerekmekteydi. Bunlar hem maliyeti artırıyor, hem de programlamayı zorlaştırıyordu. İşte Microchip firmasının PIC adını verdiği mikrodenetleyicilerle bu sorunlar ortadan kaldırıldı.

Fiyatları 5-6 $ olan bir PIC16F84 yongayla, RAM ve Giriş / Çıkış portu gibi birimlere ihtiyaç duymadan istediğimiz devreyi kurabilir, ücretsiz olarak edinebileceğimiz PIC ASSEMLY adı verilen program sayesinde PIC'i kolaylıkla programlayabiliriz.

Bilgisayar denetimi gerektiren bir uygulamayı geliştirirken seçilecek mikrodenetleyicinin ilk olarak tüm isteklerinizi yerine getirip getirmeyeceğine, daha sonra da maliyetinin düşüklüğüne bakmalıyız. Ayrıca yapacağımız uygulamanın devresini kurmadan önce seçtiğimiz mikrodenetleyicinin desteklediği bir program üzerinde simülasyonunu yapıp yapamayacağımızı dikkate almalıyız. [13]

İşte bütün bu özellikleri göz önüne aldığımızda PIC'leri kullanmanın akılcı bir yol olduğu görülmektedir. PIC16F84 mikrodenetleyicisinin program belleği flash teknolojisiyle üretilmiştir. Flash hafıza teknolojisi ile üretilen bir belleğe yüklenen program, yongaya uygulanan enerji kesilse bile silinmez. Yine bu tip bir belleğe istenirse yeniden yazılabilir. Flash hafızalar bu özellikleri ile EEPROM bellekler ile aynı görülmektedir. Gerçekte de flash ile EEPROM bellek aynı şeylerdir.

Flash belleğe sahip olan PIC16F84'ü programladıktan ve deneylerde kullandıktan sonra, silip yeniden program yazmak, PIC ile çalışmaya başlayanlar için büyük kolaylıklar sağlar. Böylece işe yeni başlayanlar yaptıkları programlama hataları nedeniyle yongayı atmak zorunda kalmayacaktır. Gerçekte EPROM program hafızası olan yongalara da yeniden yazmak mümkündür ama, bu durumda bir EPROM silici

cihaza ihtiyaç vardır. Silici cihazı bulunsa bile programı bellekten silmek için en az 10-15 dakika beklenmek zorunda kalınacaktır. İşte PIC16F84'ün bu özellikleri, mikrodenetleyici kullanmaya yeni başlayanlar için ve mikrodenetleyici temel kavramlarının anlatılabilmesi için ideal bir seçenektir [13]. Bu bölümde mikrodenetleyici kavramları anlatılırken PIC16F84 mikrodenetleyicisi üzerinden anlatılacaktır.

2.2 PIC 16F84 Mikrodenetleyici

2.2.1 Mikroişlemci temel kavramları

Mikroişlemci: Ön belleğine yazılan programı işleterek istenilen çıkışlara yönlendiren birimdir. Mikroişlemci veya sayısal bilgisayarlar üç temel kısımdan ve bunlara ek olarak bazı destek devrelerden oluşur.

i) CPU – Merkezi işlem ünitesi
ii) Giriş / Çıkış
iii) Bellek

Her bir temel kısım en basitten en karmaşığa kadar çeşitlilik gösterir.

Giriş / Çıkış (Input / Output): Sayısal, analog ve özel fonksiyonlardan oluşur. Mikroişlemcinin dış dünya ile ilişkisini sağlar. Mikroişlemciye verilen ve işlemlerden alınan veriler bu hat üzerinden sağlanır.

CPU (Central Proccesing Unit – Merkezi İşlem Birimi): Sistemin kalbidir. Bu birim hesapları yapmak ve verileri idare etmek için 4, 8 veya 16 bitlik veri formatında çalışır. Bir mikroişlemcide temelde kullanılan üç sinyal yolu vardır;

a) Veri sinyal yolu: Bu yol, işlemci, bellek ve çevre birimleri arasında veri iletmek için kullanılır.

b) Adres sinyal yolu: Bu yol, işlemcinin program komutlarına ve veri saklama alanlarına erişimi sağlayan bellek adreslerini, ROM ve RAM belleklerine göndermek için kullanılır.

c) Kontrol sinyal yolu: Bu yol, RAM belleğine veri yazıldığı veya ondan veri okunduğuna dair bilgi vermek gibi, denetim amaçları için kullanılır. Bu yol aynı zamanda kesmelerin kullanımına olanak tanıyan bağlantıları da içerir. Tipik bir mikroişlemci komutunun yürütülmesi her üç sinyal yolunun da kullanımını gerektirebilir. Böylelikle kullanılan ek devreler artarak maliyet yükselir ve tasarım çok karmaşık hal alır. İşlemci ilk olarak komuta, komut adresini adres sinyal yoluna koyarak erişir. İkili kodlardan oluşan bu adres, buna karşılık gelen bellek konumu tarafından tanınır ve bu konum istenen komutu veri yolundan işlemciye gönderir. Örneğin eğer bu komut, verinin işlemciden gönderilmesini ve bir RAM konumunda saklanmasını gerektiriyorsa işlemci, adres sinyal yoluna istenen konumu belirtmek, veri yoluna veriyi iletmek ve kontrol sinyal yoluna da RAM'a yazıyor olduğunu belirtmek için kullanılır.

2.2.2 Mikrodenetleyiciler

Mikrodenetleyici: Bir yazılım olmadan hiçbir işe yaramayan bir plastik, metal ve temizlenmiş kum yığınıdır. Mikrodenetleyiciyi kontrol eden bir yazılım olduğundaysa neredeyse sınırsız bir uygulamaya sahiptir.

Mikrodenetleyicinin Avantajları: Mikrodenetleyicilerin mikroişlemcilere olan üstünlükleri oldukça fazladır. Örneğin mikroişlemcili bir sistem yapıldığında mikroişlemcinin yanı sıra hafızalar (RAM, ROM veya EPROM), Giriş / Çıkış birimi ve buna benzer birçok sistem kullanılmaktadır. Bu karışık sistemin hem tasarlanması hem de yapımı zordur, aynı zamanda da maliyeti oldukça yüksektir. Mikrodenetleyicilerde ise bir sistemin çalıştırılabilmesi için yalnızca bir mikrodenetleyici ve osilatör devresi yeterli gelmektedir. Sistemde gerekli olan ön bellek ve Giriş / Çıkış birimi mikrodenetleyiciler içinde bir yonga halindedir.

2.3 PIC Mikrodenetleyicilerine Giriş

PIC'in kelime anlamı Peripheral Interface Controller, Çevre Arabirimlerini Kontroledici anlamındadır. İlk olarak 1994 yılında 16 bitlik ve 32 bitlik büyük işlemcilerin, giriş ve çıkışlarındaki yükü azaltmak ve denetlemek amacıyla çok hızlı bir çözüme ihtiyaç duyulduğu için geliştirilmiştir.

2.3.1 PIC mikrodenetleyicilerinin tercih sebepleri

a) Lojik uygulamalarının hızlı olması

b) Fiyatının oldukça ucuz olması

c) 8 bitlik mikrodenetleyiciler olması ve bellek ile veri için ayrı yerleşik Bus'ların kullanılması

d) Veri ve belleğe hızlı olarak erişiminin sağlanması

e) PIC'e göre diğer mikrodenetleyicilerde veri ve programı taşıyan bir tek bus bulunması, PIC'lerde ise veri ve bellek için ayrı bus'ların olması nedeniyle PIC'in diğer mikrodenetleyicilerden iki kat daha hızlı olması

f) Herhangi bir ek bellek veya Giriş / Çıkış elemanı gerektirmeden sadece 2 kondansatör ve bir direnç ile çalışabilmeleri

g) Yüksek frekanslarda çalışabilme özelliği

h) Stand-by durumunda çok düşük akım çekmesi

i) Kesme kapasitesi ve 14 bit komut işleme hafızası

Kod sıkıştırma özelliği ile aynı anda birçok işlem gerçekleştirebilmesi, PIC mikrodenetleyicileri çeşitli özelliklerine göre PIC16C6X, 16C7X, 16C5X gibi gruplara ayrılırlar.

2.4 PIC'in Kullanımı için Gerekli Aşamalar

Giriş / Çıkış : Mikrodenetleyicinin dış dünya ile ilişkisini sağlayan, giriş ve çıkış şeklinde ayarlanabilen bağlantı pinleridir. Giriş / Çıkış çoğunlukla mikrodenetleyicinin iletişim kurmasına, kontrol etmesine veya bilgi okumasına izin verir.

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

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

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

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

Programlayıcı: Yazılımın mikrodenetleyici belleğinde programlanmasını ve böylece ICE'ın yardımı olmadan çalışmasını sağlayan bir birimdir. Çoğunlukla seri porta (örneğin PICSTART, PROMASTER) bağlanan bu birimler çok çeşitli biçim, ebat ve fiyatlara sahiptir.

Kaynak Dosyası: Hem Assembler'in, hem de tasarımcısının anlayabileceği dilde yazılmış bir programdır. Kaynak dosyasının mikrodenetleyici tarafından anlaşılabilmesi için önceden assemble edilmiş olmalıdır.

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

Nesne Dosyası (Object File): Assembler tarafından üretilen bu dosya, programcı, simülatör veya ICE'nin anlayabilecekleri ve böylelikle dosyanın işlevlerinin çalışmasını sağlayabilecekleri bir dosyadır. Dosya uzantısı assemble edicinin emirlerine bağlı olarak, .OBJ veya .HEX olur.

2.4.1 PIC mikrodenetleyicilerinin özellikleri

Güvenirlik: PIC komutları bellekte çok az yer kaplarlar. Dolayısıyla bu komutlar 12 veya 14 bitlik bir program bellek yerine yazılabilirler. Harvard mimarisi teknolojisi kullanılmayan mikrodenetleyicilerde de yazılım, programının veri kısmına atlama yaparak bu verilerin komut gibi çalıştırılmasını sağlamaktadır. Buda büyük hatalara yol açmaktadır. PIC'ler de bu durum engellenmiştir.

Hız: PIC oldukça hızlı bir mikrodenetleyicidir. Her bir komut döngüsü 1µsn' dir. Örneğin 5 milyon komutluk bir programın 20 Mhz'lik bir kristale işletilmesi yalnızca 1sn sürer. Bu süre 386SX33 hızının yaklaşık 2 katıdır. Ayrıca RISC mimarisi işlemcisi olmasının hıza etkisi oldukça büyüktür.

Komut Seti: PIC'in 16C5X ailesinde bir yazılım yapmak için 33 komuta ihtiyaç duyulurken 16CXX araçları için bu sayı 35'tir. PIC tarafından kullanılan komutların hepsi kaydedici temellidir. Komutlar 16C5X ailesinde 12 bit, 16CXX ailesinde ise 14 bit uzunluğundadır. PIC'te CALL, GOTO ve bit test eden BTFSS ve INCFSZ gibi komutlar dışında diğer komutlar 1 saykıl çeker. Belirtilen komutlar ise 2 saykıl çeker.

Statik İşlem: PIC tamamıyla statik bir işlemcidir. Yani saat durdurulduğunda da tüm kaydedici içeriğini korunur. Pratikte bunu tam olarak gerçekleştirebilmek mümkün değildir. PIC mikrodenetleyicisi program işletilmediği zaman uyuma moduna geçerek microdenetleyicinin çok düşük akım çekmesini sağlar. PIC uyuma moduna geçirildiğinde, saat durur ve PIC uyuma işleminden önce hangi işleminden önce ve hangi durumda olduğunu çeşitli bayraklarla ifade eder (elde bayrağı, sonuç sıfır bayrağı vs.) PIC uyuma modunda 1µA'den küçük değerlerde akım çeker (stand-by akımı).

Sürme Özelliği (Sürücü Kapasitesi): PIC yüksek bir çıktı kapasitesine sahiptir. Tek bacaktan 40 mA akım çekebilmekte ve entegre toplamı olarak 150 mA akım akıtma kapasitesine sahiptir. Entegrenin 4MHz osilatör frekansında çektiği akım çalışırken 2mA, stand-by durumunda ise 2µA kadardır.

Seçenekler: PIC ailesinde her türlü ihtiyaçların karşılanacağı çeşitli hız, sıcaklık, kılıf, Giriş / Çıkış hatları, zamanlama (timer) fonksiyonları, seri iletişim portları, A/D ve bellek kapasitesi seçenekleri bulunur.

Çok Yönlülük: PIC çok yönlü mikrodenetleyicidir ve ürünün içinde yer darlığı durumunda birkaç mantık kapısının yerini değiştirmek için düşük maliyetli bir çözüm bulunur.

Güvenlik: PIC endüstride en üstünler arasında yer alan bir kod koruma özelliğine sahiptir. Koruma bitinin programlanmasından itibaren, program belleğinin içeriği, program kodunun yeniden yapılandırılmasına olanak verecek şekilde okunmaz.

Geliştirme: PIC program geliştirme amacıyla programlanabilip tekrar silinebilme özelliğine sahiptir (EPROM, EEPROM). Aynı zamanda seri üretim amacıyla bir kere programlanabilir (OTP) özelliklerine sahip olanları da vardır.

Liste Dosyası: Assembler tarafından oluşturulan ve kaynak dosyadaki tüm komutları, hexadesimal sistemdeki değerleri ve tasarımcının yazmış olduğu yorumlarıyla birlikte içeren bir dosyadır. Bir programı hatalardan arındırırken araştırılacak en yararlı dosya budur. Çünkü bu dosyayı izleyerek yazılımlarda neler olup bittiğini anlama şansı kaynak dosyasından daha fazladır. Dosya uzantısı .LST'dir.

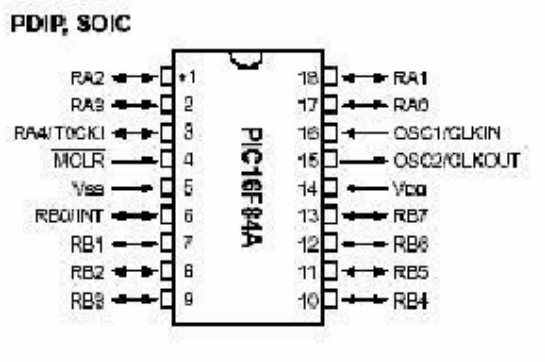
Diğer Dosyalar: Hata dosyası (error file: uzantısı: .ERR) hataların bir listesini içerir ancak bunların kaynağı hakkında hiçbir bilgi vermez. Uzantısı .COD olan dosyalar emülatör tarafından kullanılırlar.

Hatalar: Tasarımcının farkında olmadan yaptığı hatalardır. Bu hatalar, basit yazılım hatalarından , yazılım dilinin yanlış kullanımına kadar uzanır. Hataların çoğu derleyici tarafından bulunur ve bir .LST dosyasında görüntülenir. Kalan hataları bulmak ve düzeltmekte geliştiriciye düşer.

2.5 PIC Mikrodenetleyicilerinin Donanımsal İncelenmesi

2.5.1 PIC16F84'ün bacak bağlantısı

PIC16F84 mikrodenetleyicisinin bacak bağlantısı (pin uçları) Şekil 2.1' de gösterilmiştir [13, 14].



Şekil 2.1 PIC16F84' ün bacak bağlantısı

2.5.2 PIC16F84'ün genel özellikleri

. DC 10 MHz çalışma hızı

. 14 bit genişliğinde veri yolu

. 8 bit genişliğinde veri yolu

. 8 seviye yığın derinliği

. Doğrudan, dolaylı ve göreceli adresleme metodları

. EEPROM'da 40 yıl veri tutma

. EEPROM veri belleğine 1.000.000 kez veri yazma – silme

. 13 adet tek tek kontrol edilebilen giriş – çıkış pini

. 4 kesilim kaynağı

. 8 bit zamanlayıcı / sayıcı, 8 bit programlanabilir ön bölücü

. Tamamen statik tasarım

. Düşük güçlü, yüksek hızlı CMOS Flash / EEPROM teknolojisi

. Geniş çalışma voltaj aralığı (Ticari = 2.0 V - 6.0 V)

. Düşük güç tüketimi (5V - 4 Mhz'de tipik 2 mA'den düşük, 2V – 32 KHz'de tipik 15µA, 2V'ta standby akımı 1 µA'den düşük)

. Kod koruma

. Uyarma modunda güç tasarrufu

. 1024 kelime güncellenebilir

. 68 bayt veri belleği (RAM – rasgele erişimli bellek)

. 64 bayt veri belleği (EEPROM – elektrikle silinebilir programlanabilir yalnız okunabilir bellek)

. Yalnızca 35 adet tek kelime komut öğrenme

. Çalışma sıcaklığı (Ticari = 0 ˚C ile +70 ˚C arasında, Endüstriyel = -40 ˚C ile +85˚C arasında)

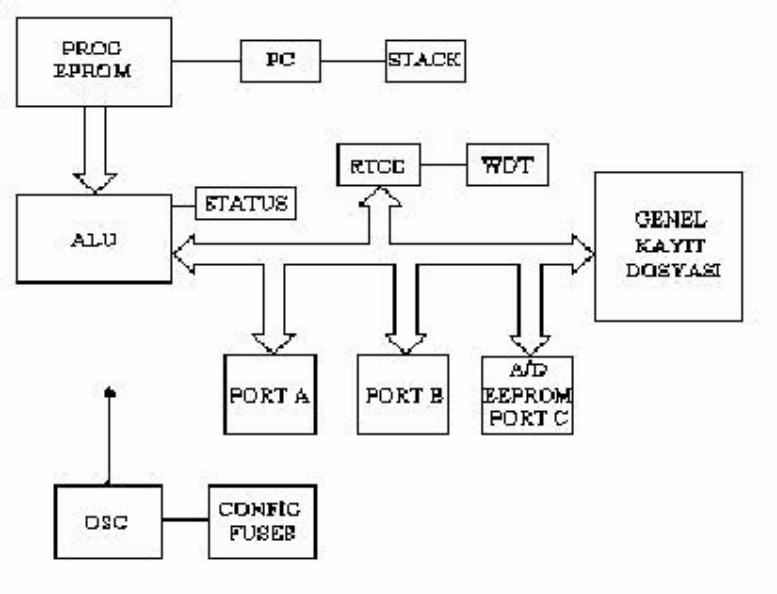
2.5.3 PIC mikrodenetleyicilerinin iç yapısı

CPU bölgesinin kalbi ALU'dur. (Aritmetic Logic Unit - Aritmetik Mantık Birimi). ALU, W (Working – Çalışan) adında bir kaydedici içerir. PIC, diğer mikroişlemcilerden, aritmetik ve mantık işlemleri için bir tek ana kaydediciye sahip oluşuyla farklılaşır. W kaydedicisi 8 bit genişliğindedir ve CPU'daki herhangi bir veriyi transfer etmek üzere kullanır.

CPU alanında ayrıca iki kategoriye ayırabileceğimiz veri kaydedici dosyaları (Data Register Files) bulunur. Bu veri kaydedici dosyalarından biri Girş / Çıkış ve kontrol işlemlerinde kullanılırken, diğeri RAM olarak kullanılır.

PIC'ler de Harward mimarisi kullanılır [14]. Harward mimarisi mikrodenetleyicilerde veri akış miktarını hızlandırmak ve yazılım güvenliğini arttırmak amacıyla kullanılır. Ayrı bus'ların kullanımıyla veri ve program belleğine hızlı bir

şekilde erişim sağlanır. Şekil 2.2' de PIC mikrodenetleyicisinin blok diyagramı görülmektedir.



Şekil 2.2 PIC mikrodenetleyicisinin blok diyagramı

PIC mikrodenetleyicilerini donanımsal olarak incelerken PIC16F84 üzerinde durarak bu PIC'i temel alıp donanım incelenecektir. Bellek ve bazı küçük farklılıklar dışında burada anlatılanlar bütün PIC'ler için geçerlidir.

2.6 Genel Tanımlama

PIC16F84 veya …F84, düşük maliyetli, yüksek performanslı, CMOS, tamamen-statik, 8 bit mikrodenetleyicilerdir.

Tüm PIC 16/17 mikrodenetleyiciler RISC mimarisini kullanmaktadır. PIC16CXX mikrodenetleyicileri birçok esas özelliklere sahiptir. 8 seviyeli, derin küme ve çoklu iç ve dış kesme kaynaklarına sahiptir. Harward mimarisinin ayrı komut ve veri taşıyıcısıyla ayrı 8 bitlik geniş veri taşıyıcılı, 14 bitlik geniş komut kelimesine imkan vermektedir. İki aşamalı komut hattı tüm komutların tek bir saykılla (çevrimle) işlenmesini sağlamaktadır. Yalnızca bazı özel komutlar 2 saykıl çekerler. Bu komutlar dallanma komutlarıdır.

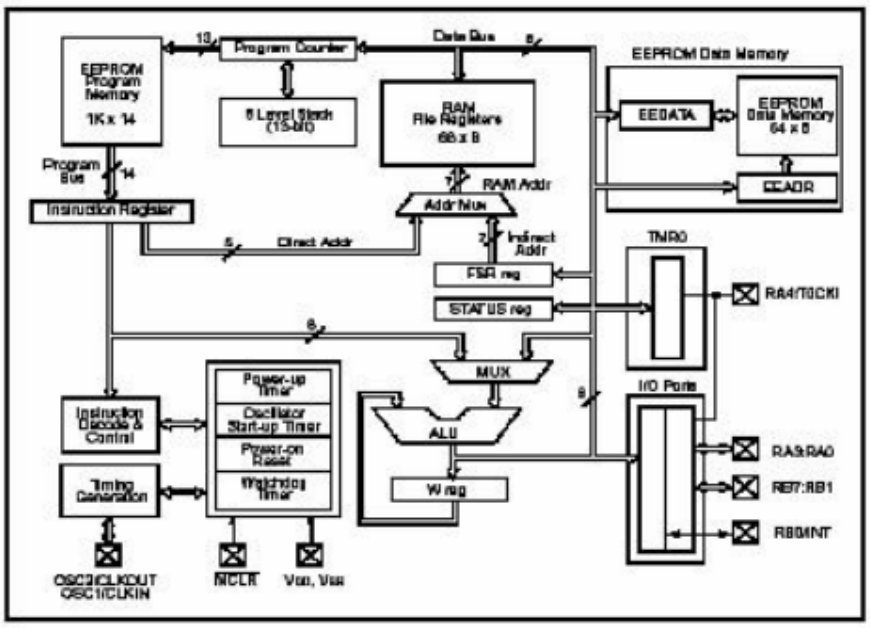
PIC16CXX mikrodenetleyicileri tipik olarak 2:1 oranında kod sıkıştırmasına erişmektedir ve sınıflarındaki 8 bit mikrodenetleyicilerden 2:1 oranında hız artırılmasına olanak sağlanmaktadır.(10MHz)

PIC16F84 Mikroyongası 36 bitlik RAM belleğine, 64 bayt EEPROM belleğine ve 13 Giriş / Çıkış pinine sahiptir. Bunun yanı sıra, timer ve sayaçta mevcuttur.

PIC16CXX ailesi dış elemanları azaltacak spesifik özelliklere sahiptir ve böylece maliyet minimuma inerken, sistemin güvenirliği artmakta, enerji sarfiyatı ise azalmaktadır. Bunun yanı sıra tüm PIC'ler de 4 adet osilatör seçeneği mevcuttur. Bunlarda tek pinli RC osilatör , düşük maliyet çözümünü sağlamakta (4MHz). LP osilatör (kristal veya seramik rezonatör), enerji sarfiyatını minimize etmekte (asgari akım) (40 KHz). XT kristal veya seramik rezonatör osilatörü standart hızlı ve HS kristal veya seramik rezonatörlü osilatör çok yüksek hıza sahiposilatör (20 MHz).

PIC mikrodenetleyicilerinin en büyük özelliği uyku modu özelliğidir. Bu mod ile PIC işlem yapılmadığı durumlarda uyuma moduna geçerek çok düşük akım çeker (5 mA). Kullanıcı birkaç iç ve dış kesmelerle PIC'i uyuma modundan çıkarabilmektedir. Yüksek güvenirlikli Watchdog timer kendi bünyesindeki yonga üstü RC osilatörü ile yazılımı kilitlemeye karşı korumaktadır.

PIC16F84 EEPROM program belleği, aynı aygıt paketinin orjinali ve üretimi için kullanılmasına olanak vermektedir. Yeniden programlanabilirliği mikrodenetleyiciyi uygulamanın sonunda çıkarmadan kodu güncelleştirmeye izin vermektedir. Bu aygıtın kolayca erişilemediği, fakat prototipin kod güncelleştirmesi gerekli olduğu durumlarda, bir çok uygulamanın geliştirilmesinde yararlıdır. Bunun yanı sıra bu kodun güncelleştirilmesi diğer ayrı uygulamalarda da yararlıdır. Şekil 2.3' de PIC'in basitleştirilmiş iç yapısı görülmektedir [14].



Şekil 2.3 PIC16F84'ün basitleştirilmiş iç yapısı

PIC'ler özellikle de PIC16F84 yüksek hızlı otomobillerden, motor kontrolü uygulamaları, düşük enerji sarfiyatlı uzaktan çalışan sensörler, elektronik kilitler, güvenlik aygıtları ve akıllı kartlara kadar bir çok uygulamalarda kullanılırlar. EEPROM teknolojisi uygulama programların (transmitter kodları, motor hızları, alıcı frekansları, güvenlik kodları vb.) uygulanmasını son derece hızlı ve uygun hale getirmektedir. Küçük boyutlarıyla bu mikrodenetleyiciler alan sınırlaması bulunan uygulamalarda kusursuzdurlar. Düşük maliyet, düşük enerji sarfiyatı, yüksek performans, kullanım kolaylığı ve Giriş / Çıkış esnekliği özellikle de PIC16F84 mikrodenetleyicisinin daha önce kullanılması hiç düşünülmeyen alanlarda kullanılmasını sağlamaktadır (bunlar; timer fonksiyonları, seri haberleşme, PWM fonksiyonları ve birlikte işlemci uygulamaları gibi).

Seri sistem içi programlama özelliği (iki pinin üzerinden) ürünün tamamen toplanması ve test edilmesinden sonra ürünün alıştırılması esnekliği imkanı

vermektedir. Bu özellik sayesinde ürün serileştirilebilmekte ve veriler saklanabilmektedir.

2.7 Gelişme Desteği

PIC16CXX sınıfı tam özellikli mikrobirleştirici, yazılım, simülator, devre içi emülatör, düşük maliyetli program geliştirme ve tam özellikli programlayıcı ile desteklenmiştir. PIC16F84, PIC16C5X mikrodenetleyicilerin geliştirilmiş halidir. PIC16C5X için yapılan devrelerde kolaylıkla PIC16F84 kullanılabilir.

2.8 Hafıza Organizasyonu

PIC içersinde iki hafıza bloğu bulunur. Bunlar program hafızası ve data hafızasıdır. Her bir bloğa geçişin kendi bacakları vardır. Data hafızası genel amaçlı RAM ve özel fonksiyon kaydedicilerini içerir. Data hafıza alanı ayrıca EEPROM hafızasını içermektedir. Data EEPROM hafızasının yazılması ve okunması için adresi tayin eden, dolaylı adres işaret edicisidir. Data EEPROM hafızasında 64 bayt'ın adresi 00-3HF arasındadır.

2.8.1 Program hafıza organizasyonu

PIC16F84 8K×14 program hafıza alanını adresleme yeteneğinde olan 13-bitlik program sayıcısına sahiptir. Bunun sadece 1K×14' ü (0000-03FH) fiziksel olarak gerçekleştirilir. Fiziksel olarak gerçekleştirilmiş adres üzerinde bir bölgede geçiş kapsamaya sebep olacaktır. Örneğin, bölgeler 20H, 420H, 820H, C20H, 1020H, 1820H ve 1C20H aynı değerde olacaktır. Reset vektörü 0000H'de ve kesme vektörü ise 0004H adresindedir.

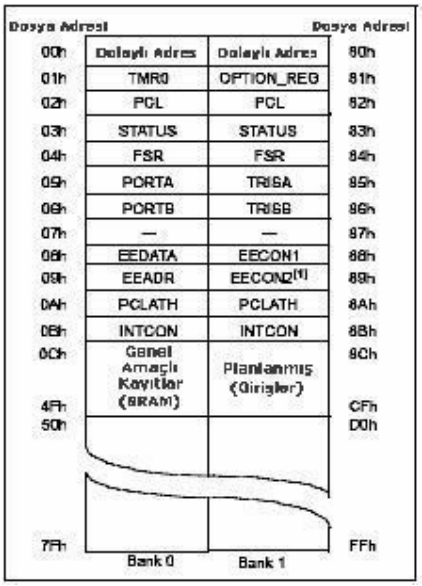
2.8.2 Data hafıza organizasyonu

Data hafızası iki alana bölünmüştür. Birincisi özel fonksiyon kaydedici (SFR) alanı; ikincisi özel amaçlı kaydedici (GPR) alanıdır. SFR'ler PIC'in operasyonunu kontrol eder.

Data hafızasının parçaları banka konulur. Bu hem SFR hem de GPR alanı içindir. GPR alanı genel amaçlı RAM'in 96 bayt'dan büyük olması için banka konulur. SFR alanı banka konulmuş çevresel fonksiyonları kontrol eden kaydediciler içindir. Banka konulma bank seçiminin kontrolü ile gerçekleştirilir. Bu kontrol bitleri durum (status) kaydedicilerine yerleştirilmiştir.

Komutlar MOVWF ve MOVF, değerleri W kaydedicisinden kaydedici dosyasındaki ('F') herhangi bir bölgeye karşılık olarak kaydırılır.

Dolaylı adreslemede data hafızasının banklarına geçiş için RPO ve RP1 bitlerinin değerleri kullanılır. Şekil 2.4 Program Hafıza Organizasyonunu [14-16] göstermektedir.



Şekil 2.4 Program Hafıza Organizasyonu

2.8.3 Genel amaçlı kaydedici (GPR)

GPR 8 bit genişliğinde olup, bütün PIC mikrodenetleyiciler belirli miktarda GPR bölgesine sahiptir. Bu kısım RAM (Veri depolama alanı) olarak kullanılır.

2.8.4 Özel fonksiyon kaydedicileri

SFR'ler PIC operasyonu kontrol etmek amacıyla CPU ve çevresel fonksiyonlar tarafından kontrol edilir. Bu kaydediciler statik RAM'lerdir.

Özel kaydediciler iki sette sınıflandırılır. Çekirdek fonksiyon ile birleştirilmiş olanlar bu bölümde tanımlanmaktadır. Çevresel özelliklerin operasyonu ile bağlantılı olanlar spesifik özellik bölümünde tanımlanmaktadır. Çizelge 2.1' de PIC16F84' ün kaydedici dosyaları bank 0 ve bank 1 dikkate alınarak verilmiştir [14] .

Adresler	Dosya Adı	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Bank 0									
00h	INDF	FSR Kaydedicisi tarafından kullanılır. Fiziksel bir kaydedici değildir.							
01h	TMR0	8-Bit gerçek zamanlı darda sayacı							
02h	PCL	PC' nin düşük seviyeli bitleri							
03h	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
04h	FSR	Dolaylı hafıza adres göstericisi							
05h	PORTA	—	—	—	RA4/T0CKI	RA3	RA2	RA1	RA0
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT
07h	—	Tanımlanmamış konum							
08h	EEDATA	EEPROM veri kaydedicisi							
09h	EEADR	EEPROM adres kaydedicisi							
0Ah	PCLATH	PC'nin yüksek byte'larını bulunduran kaydedici							
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
Bank 1									
80h	INDF	FSR Kaydedicisi tarafından kullanılır. Fiziksel bir kaydedici değildir.							
81h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
82h	PCL	PC'nin düşük seviyeli bitleri							
83h	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
84h	FSR	Dolaylı hafıza adres göstericisi							
85h	TRISA	—	—	—	PORTA (I/O)'nın data yönetim kaydedicisi				
86h	TRISB	PORTB (I/O)'nın data yönetim kaydedicisi							
87h	—	Tanımlanmamış konum							
88h	EECON1	—	—	—	EEIF	WRERR	WREN	WR	RD
89h	EECON2	EEPROM kontrol kaydedicisi 2 - Fiziksel bir kaydedici değildir							
0Ah	PCLATH	—	—	—	PC'nin yüksek byte'larını tutan kaydedici				
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF

Çizelge 2.1 PIC16F84' ün kaydedici dosyaları

2.8.5 INTCON kaydedicisi

INTCON Kaydedici tüm kesme kaynaklarının değişik bitlerini aktif yapan, okunabilir ve yazılabilir kaydedicidir.

Karşılık gelen kesmeyi mümkün kılan bit, yani GIE biti temizlense bile TOIF, INTF veya RBIF bitleri belirtilmiş durum tarafından yerleştirilecektir. Çizelge 2.2 INTCON kaydedicisi ve bu kaydedici içerisindeki bitlerin görevleri tanımlanmıştır.

Bit No	*Kaydedici Bitler*	*Açıklama*
BIT0 R/W	RBIF	RB port değişim kesmesi. RB <7:4> girişleri değiştiği zaman set edilir
BIT1 R/W	INTF	INT kesme bayrağı. INT kesmesi meydana geldiğinde set edilir
BIT2 R/W	RTIF	RTCC taşma bayrağı. RTCC de taşma olduğu zaman set edilir
BIT3 R/W	RBIE	RBIF kesme aktif biti. RBIE=0 ise pasif, RBIE=1 ise aktif
BIT4 R/W	INTE	INTF kesme aktif biti. INTE=0 ise pasif, INTE=1 ise aktif
BIT5 R/W	RTIE	RTCC kesme aktif biti. RTIE=0 ise pasif, RTIE=1 ise aktif
BIT6 R/W	ADIE	A/D kesme aktif biti. ADIE=0 ise pasif, ADIE=1 ise aktif
BIT7 R/W	GIE	Global kesme aktif biti. ICIE=0 ise pasif, GIE=1 ise aktif

Çizelge 2.2 INTCON kaydedicisi

2.8.6 Durum kaydedicisi

Durum kaydedicisi, data hafızası için ALU, RESET durumları ve bank seçimlerinin aritmetik durumlarını içerir. Durum kaydedicisi Z, DC veya C bitlerini etkileyen komut için gidilecek yer ise bu üç bite yazım engellenir. Bu bitler birlenir veya PIC mantığına göre sıfırlanır. Ayrıca TO ve PD bitlerine yazılamaz. Bu yüzden varış olarak durum kaydedicisi komutu verilecek olursa sonuç tasarlanandan farklı olabilir. Çizelge 2.3' de durum kaydedicisi ve bitlerinin fonksiyonları [14] verilmiştir.

Bit 0	C	**Elde/ Borç Bit:** ADDWF, SUBWF, ADDLW ve SUBLW komutlarında en çok ağırlıklı bit bir elde bitine sebep oluyorsa, bu bit set edilebilir. Çıkarma işlemi, ikiler tamlayan metoduna göre yapılır.Döndürme komutlarından (RRF, RLF) komutlarında kaynak kaydedicisinin en yüksek ağırlıklı veya en düşük ağırlıklı biti buraya yüklenir.
Bit 1	DC	**Digit elde/ Borç bit:** ADDWF, SUBWF, ADDLW ve SUBLW komutları için sonuçta elde oluşuyorsa bu bit set edilir.
Bit 2	Z	**Sıfır Bit:** Aritmetik veya mantık işlemlerinin sonucu sıfıra eşit ise set edilir. Diğer durumlarda reset edilir.
Bit 3	PD	**Uyku bit:** CLRWDT komutu ile veya güç verme işlemi sonucunda set edilir.Watchdog timer zamanlayıcı bitimi sonucunda reset edilir.
Bit 4	T0	**Zaman aşımı bit:** CLRWDT, SLEEP komutu ile veya güç verme işlemi sonucunda set edilir. Watchdog timer zamanlayıcı bitimi sonucunda reset edilir.
Bit 5	RP0	**Doğrudan adresleme için yazmaç sayfa seçimi:** RP1, RP0 00: Sayfa 0 01: Sayfa1 10: Sayfa 2 11: Sayfa3 Her sayfa 128 byte' dan oluşur. PIC 16F84 sadece RP0' ı kullanır. RP1 biti genel amaçlı oku/ yaz biti olarak kullanılabilir.
Bit 6	RP1	**Doğrudan adresleme için yazmaç sayfa seçimi:**
Bit 7	IRP	**Dolaylı adresleme için yazmaç sayfa seçimi:** IRP 0: Sayfa 0,1 1: Sayfa 2,3 IRP biti PIC 16F8X tarafından kullanılmaz. IRP sıfırlanmış olmalıdır.

Çizelge 2.3 Durum (Status) kaydedicisi

2.8.7 PCL ve PCLATH

Program sayıcı (PC) 13 bit genişliğindedir. Alçak dereceli byte, PCL, okunabilir ve yazılabilir, PC'nin yüksek dereceli byte'ı ise (PCH) direkt olarak okunup yazılamaz. CALL veya GOTO komutları ile PC' ye yeni bir değer yüklendiği zaman PC'nin yüksek bitleri PCLATH' ten yüklenirler.

2.8.8 Kesmeler

PIC16F84 ailesi 4 adet kaynak kesmesine sahiptir:

i) Harici kesme bacağı RB0/INT
ii) TMR0 taşma kesmesi
iii) PORTB de değişim durumunda kesme
iv) EEPROM'a yazma işlemi tamamlanması durumunda kesme işlemi

Genel kesme biti GIE (INTCON < 7 >) eğer birlenmiş (set) ise bütün kesme maskeleri düşmekte veya bütün kesmeler yeteneksiz kılınmaktadır. GIE reset durumunda sıfırlanır.

2.8.9 Yığın (Stack)

PIC16F84 8 adet 13 bit genişliğinde donanım yığınına sahiptir. Yığın alanı, ne data alanının ne de programın parçası değildir. Yığın işaret edicisi okunamaz ve yazılamaz [14]. PC' nin 13 bitlik değeri, RETURN, RETLW veya RETFIE komutlarının işlenmesi veya CALL komutunun gerçekleştirildiği durumda, yığın üzerine itilir. PCLATH itme veya çekme işlemi sırasında etkilenmez.

2.9 CPU' ya ait Diğer Özellikler

PIC16F84'de, sitemde maksimum güvenlik sağlamak için çok sayıda nitelik tasarlanmıştır. Bunları harici eleman sayısını en aza indirmek, maliyeti düşürmek, işlem modunda güç tüketiminde tutumlu olmak,kod koruma sağlamak [14] şeklinde sıralayabiliriz.

PIC entegrelerinin nitelikleri

- OSC seçimine imkan verebilmeleri
- Reset
- Güç-on reset (POR)
- Güç-verme timer (PWRT)
- Osilatör start-verme timer (OST)
- Kesme
- WDT
- Uyku
- Kod koruma
- Seri olarak devrenin programlanması

Güç verince gerekli gecikmeyi sağlamak için iki adet zamanlayıcı vardır. Birincisi Osilatör Start-verme Timer (OST) olup kristal osilatör kararlı olana kadar yongayı resette tutar. İkincisi ise güç-verme timer (PWRT) güçte kararlılık sağlamak için devreyi resette tutuyor. Yonga bu iki zamanlayıcı vasıtasıyla harici reset devresine ihtiyaç duymuyor.

Uyku modunda, güç azalımından dolayı çok küçük akım harcıyor. WDT çıkışı harici reset veya kesme sayesinde kullanıcı uyku modundan uyandırabilir. Birkaç osilatör seçenek şartlarından uygun olanına izin verilmelidir. RC osilatör maliyeti korumakta, LC kristal osilatörü ise güç korumaktadır. Konfigürasyon bitleri içinde kullanıcı uygun olanı seçecektir.

2.10 Osilatör Tipi

PIC16F84 4 farklı osilatör moduna sahiptir. Kullanıcı bu 4 osilatör modunun birini seçebilir [14].

i) LP düşük güçlü kristal
ii) XT kristal
iii) HS yüksek hızlı kristal
iv) Rezistör / kapasitör

2.11 Reset

Reset durumu yongaya güç uygulayarak, MCLR giriş düşük duruma getirilerek veya bir WDT zaman aşımı ile gerçekleştirilebilir. Osilatör başlatıcı zamanlayıcı aktif olduğu zaman veya MCLR girişi alçak seviyeli olduğu sürece devre reset durumunda kalır. MCLR girişi yüksek seviyeye ayarlanır ayarlanmaz osilatörü başlatan zamanlayıcı aktif hale gelir. MCLR Vdd'ye bağlanması ile güç kesildiğinde bu uygulamalar OST gücü ile çalışmaya başlar. WDT zaman aşımı durumunda zaman aşımının sonunda çalışmaya başlar.

2.12 Watchdog Zamanlayıcısı (WDT)

WDT, yonga üzerinde herhangi bir harici elemana gerek duymadan serbest çalışan RC osilatörüdür. Bu WDT'ın devrenin OSC1 / OSC2 bacaklarındaki Clock olursa da çalışacağı anlamına gelir [14, 17]. Örneğin uyku komutu işlemide WDT zaman aşımı durumu devresi üretir.

2.13 Uyku Modu (Powerdown Mode)

Uyku modu SLEEP komutunun işletilmesi için seçilir [14]. WDT temizlenir ancak çalışmaya devam eder. Durum kaydedicisindeki PD biti sıfırlanırken T0 biti set edilir ve osilatör sürücüsü kapanır. GİRİŞ / ÇIKIŞ pinleri SLEEP komutu işletilmeden önceki durumlarını korurlar.

2.14 Uykudan Uyandırma

Aşağıdaki olaylardan herhangi birisi izlenildiğinde mikrodenetleyici uyku modundan uyanabilir [14] .

1. Harici reset girişi MCLR pininin kapalı (ON) durumunda
2. WDT zaman aşımı resetlendiğinde
3. RB portu değiştiğinde veya EEPROM bilgi yazma tamamlandığında

Devre, WDT veya harici olarak MCLR pinindeki alçak seviyeli darbeyle uyandırılırlar. Her iki durumda program tekrar işlemeye başlamadan önce entegre bir OST peryodu süresince reset modunda kalacaktır.

Durum kaydedicisindeki PD bilgisi işlemciye güç verilip verilmediğini veya uyku modunda uyku durumuna geçilip geçilmediğini anlamak maksadıyla kullanılabilir. Güç verilme durumunda bu bit set olurken SLEEP komutuyla sıfırlanır. Durum kaydedicisindeki T0 biti uyanmaya WDT'ın mı yoksa MCLR pinindeki sinyalin mi sebep olduğunu anlamak için kullanılır.

3. SICAKLIK

Sıcaklık belirli bir skalaya göre referanslanmış sıcaklık veya soğukluk derecesi olarak tanımlanır. Sıcaklık aynı zamanda bir nesne veya sistemin ısı enerjisi olarak da tanımlanabilir. Isı enerjisi doğrudan moleküler enerji (bir molekül içerisindeki parçacıkların titreşim, sürtünme ve salınma hareketleri) ile ilişkilidir. Isı enerjisi yüksek ise moleküler enerji de yüksektir.

Sıcaklık otomasyonda, tüketici eşyalarında ve imalat endüstrisinde onlarca yıldır en önemli işlem değişkenlerinden biri olarak görüldü. Dirençli termometreler ve thermokupıllarla sıcaklığın ölçümü, gerçekte 100 yıldan daha eskidir. İşlemlerin sürekli optimizasyonu termometrelerden daha yüksek beklentilerin gerçekleşmesine yol açtılar. Bu beklentileri sıcaklığın daha hızlı, daha kesin ve bir süre boyunca tekrarlanabilirliği daha iyi olan termometrelerin gerekliğidir

Sıcaklık sensörleri bir sıcaklık değişimine karşılık gelen direnç veya çıkış voltajı gibi bir fiziksel parametrenin değişimini algılarlar. Temelde iki tip sıcaklık algılayıcı elemanlar vardır [18]

3.1 Dokunma Yoluyla Sıcaklığın Ölçülmesi

Dokunma yoluyla sıcaklık algılayıcılarda sensör sıcaklığı algılanacak ortam veya nesne ile doğrudan fiziksel olarak temas halindedir, bu metotla katıların, sıvıların ve gazların oldukça geniş bir genlikte sıcaklıklarını ölçebiliriz [18].

Dirençli termometreler bir elektriksel iletkenin sıcaklıkla elektriksel direncinin değişimi ilkesi ile çalışırlar. Pozitif sıcaklık katsayılı (PTC) ve Negatif sıcaklık katsayılı (NTC) sensörler arasındaki fark, Pozitif sıcaklık katsayılı sensörlerde direnç değeri artan sıcaklıkla artarken, negatif sıcaklık katsayılı sensörlerde direnç değeri artan sıcaklıkla azalır. PTC sensörleri metallik iletkenlerdir, yaygın olarak kullanılan metaller platinyum, iridyum, nikel ve bakırdır. NTC sensörler ise uygun metal oksitlerinden yapılır, NTC' ler bazen sıcak iletkenler olarak da isimlendirilir, çünkü bunlar yalnızca çok yüksek sıcaklıklarda iyi bir elektriksel iletkenlik sergilerler.

Genellikle sıcaklıkla elektriksel direncin değişimi doğrusal olmayıp, yüksek dereceli bir polinomla verilir. [19]

$$R(T) = R_0 (1 + AT + BT^2 + CT^3 + \dots)$$

Thermokuplın çalışması bir tel boyunca sıcaklık gradienti var ise telin iletkenliğine bağlı olarak yüklerin yer değişimi prensibine dayanır. Eğer farklı iletkenliklerdeki iki iletken bir noktada birleştirilmiş iseler, sıcaklık gradientinin büyüklüğüne bağlı olarak, farklı yük yer değişiminden dolayı bir termoelektrik emf ölçülebilir.

Dirençli termometrelerle karşılaştırdığımızda, birkaç bin derce santigrata kadar çıkan çok yüksek sıcaklıklarda açık bir avantaja sahiptirler. Bunlar genellikle fırınlarda, erimiş metallerin ve sıcaklığı 250 santigratın üzerindeki değerlerin ölçülmesinde kullanılır.

3.2 Dokunmaksızın Sıcaklığın Ölçülmesi

Dokunmaksızın sıcaklık ölçen algılayıcılar ısı kaynağından yayılan ısı enerjisini ölçerler. Bu metotla yansımanın olmadığı katı ve sıvıların sıcaklıklarını ölçebiliriz, fakat doğal olarak transparant olan gazlarda sıcaklık ölçümü için uygun değildir [18].

Bu kategoride hareketli nesnelerin ve ulaşılamayan nesnelerin sıcaklığı ölçülür. Örneğin fırındaki bir nesnenin sıcaklığının uzak bir yerden ölçümü, bu durumda ölçülen nesnedeki ısıl radyasyon ölçülen değer olarak kullanılır. Pirometre olarak adlandırılan dokunmasız sıcaklık ölçüm aletlerinin temel prensibi, sıcak bir cisimden yayılan ısıl radyasyonun bir optik sistemle termokupla ulaşması prensibine dayanır.

3.3 Sensör Tipleri ve Teknolojileri

Sıcaklık sensörleri başlıca üç guruba ayrılır, bunlar elektromekanik, elektronik ve dirençli sensörlerdir [18]. Aşağıdaki bölümde her bir sensör tipinin nasıl oluşturulduğu ve sıcaklığı ölçmek için nasıl kullanıldığından söz edilmiştir.

3.3.1 Elektromekanik sensörler, bi - metal termostatlar

Bi-metal termostatlar isimlerinin ifade ettiği gibi iki farklı metalin sıcaklık ve basınç altında tek bir materyal parçası olacak şekilde bağlanmalarıyla oluşturulur. İki materyalin farklı uzama oranlarından dolayı, ısı enerjisi elektromekanik harekete dönüştürülür [18].

3.3.2 Elektronik, silikon sensörler

Silikon sensörler yarıiletken materyallerin elektriksel direnç özelliklerini kullanırlar. Silikon sensörler, özellikle çok alçak sıcaklık değerlerinde, sıcaklığa karşı oldukça lineer bir direnç artışı sağlarlar. IC tipi elemanlar, doğrudan dijital bir sıcaklık değeri verebilirler, bu durumda bir A/D dönüştürücüye gereksinim olmaz [18].

3.3.3 Thermokupıllar

Thermokupıllar farklı metal veya alaşımlardan yapılmış iki elektriksel iletkenin bir devrede bir uçlarını birleştirilmeleri ile oluşturulurlar. Thermokupıllar çok yüksek sıcaklıkların ölçümünde kullanılabilir. Tipik olarak çıplak iletkenlerle oluşturulmalarına rağmen seramik ile izole edilmişlerdir. Bütün thermokapıllar bir sıcak bağlantıya (ölçme bağlantısı) ve bir de soğuk bağlantıya (referans bağlantısı) sahiptirler [18].

Thermokupılların uçları farklı sıcaklıkların altında olduğunda, sıcaklık farkıyla orantılı olan bir akım akacaktır. Ölçme noktasındaki sıcaklık, kullanılan thermokupılın tipinin, milivolt olarak üretilen potansiyelin değeri ve referans bağlantısının sıcaklığının bilinmesi ile belirlenebilir [18].

3.4 Direnç Elemanları

3.4.1 Thermistörler

Thermistörler veya ısıya duyarlı dirençler, elektriksel dirençleri sıcaklıkla değişen elemanlardır. Tipik olarak iki veya üç metal oksidin seramik bir materyal ile karışımı ile oluşturulmuş ve iki uç lehimlenmiştir. Thermistörler iki farklı tipte yapılmışlardır. Bunlar Pozitif sıcaklık katsayılı (PTC) ve Negatif sıcaklık katsayılı (NTC) termistörlerdir. Pozitif sıcaklık katsayılı termistörlerde direnç değeri artan sıcaklıkla

artarken, negatif sıcaklık katsayılı termistörlerde direnç değeri artan sıcaklıkla azalır [18].

RTD' lere benzer şekilde belirli bir sıcaklık değeri için thermistörün direnci ± bir tolerans değeri ile belirtilir.

3.4.2 RTD' ler

RTD' ler, thermistörler gibi sıcaklığı ölçmek veya kontrol etmek için elektriksel direncin değişimini kullanırlar. RTD' ler bir algılayıcı eleman, eleman ve ölçme cihazı arasında bağlantıyı sağlayan teller ve elemanı işlemde monte etmek için gerekli bir destek kısmına sahiptirler [18]. Metal algılayıcı eleman direnci sıcaklıkla değişen bir elektriksel dirençtir.

3.5 Arabirimleme ve Tasarım Bilgisi

Herhangi bir algılama tipi teknolojisi ile ilgili en önemli göz önünde bulundurulması gereken nokta algılayıcının yerinin tespit edilmesidir. Kontrol uygulamalarında sıcaklıktaki değişim oranı oldukça küçük ise algılayıcı mümkün olduğu kadar ısı kaynağına yakın yerleştirilmelidir [18]. Bu şekilde ısıl geri kalma minimum olacaktır.

3.6 LM35 Santigrad Doğruluklu Sıcaklık Sensörü

3.6.1 Genel tanımlama

LM35 serisi çıkış voltajı Santigrad sıcaklıkla lineer olarak orantılı olan entegre devre sıcaklık sensörleridir. LM35 ° Kelvin olarak kalibrasyon yapılan lineer sıcaklık sensörlerine göre avantajlıdır, çünkü kullanıcının uygun Santigrad skalası elde etmesi için, çıkıştan büyük bir sabit voltaj değeri çıkarması gerekmemektedir. LM35 oda sıcaklığında ±1⁄4°C' lik, −55 ile +150°C sıcaklık genliğinde ise ±3⁄4°C lik tipik doğruluk değerleri sağlaması için harici bir kalibrasyona gereksinim duymaz [20].

LM35 ucuz olup, bu imalat aşaması ile ilgilidir. LM35 alçak bir çıkış empedansına sahip olup, lineer çıkışı ve doğruluklu kalibrasyonu ile sıcaklığın doğrudan okunması

için arabirimlemede veya kontrol devresi ile kullanımda kolaylık sağlar. Bu eleman tek güç kaynağı veya artı eksi güç kaynakları ile birlikte kullanılabilir. Kaynaktan yalnızca 60 μA akım çektiğinden, hava içerisinde 0.1°C' dan daha küçük bir kendi kendine ısınma değerine sahiptir. LM35 −55° ile +150°C' lik bir anma sıcaklık genliğinde çalışacak şekilde olmasına karşılık, LM35C −40° ile +110°C' lik bir anma sıcaklık genliğinde çalışacak şekilde tasarlanmıştır.

LM35 serisi TO-46 transistör paketi tipinde imal edilmekle birlikte, LM35C, LM35CA ve LM35D plastik TO-92 transistör paketi tipinde de imal edilmektedir. LM35D 8 – uçlu küçük yüzey montajı paketi şeklinde ve plastik TO-220 paketi şeklinde imal edilmektedir [20].

3.6.2 Özellikleri

Doğrudan ° Celsius sıcaklık skalası ile kalibre edilmiştir.

Lineer olarak + 10.0 mV/°C' lik bir skala çarpanına sahiptir.

0.5°C' lik bir doğruluk değeri garanti edilebilir (+25°C' da)

−55° ile +150°C' lik bir anma sıcaklık genliğinde orantılı çıkış sağlar.

Uzaktan uygulamalar için uygundur.

Düşük fiyatlıdır.

4 ile 30 volt aralığında çalışır.

60 μA' den daha az bir akım çeker.

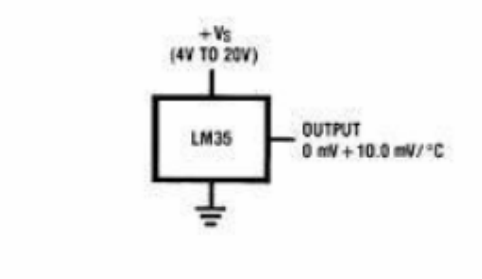
Küçük kendi kendine ısınma değeri (Hava içerisinde 0.08°C)

Tipik olarak yalnızca ±1⁄4°C sıcaklık aralığında lineer değildir.

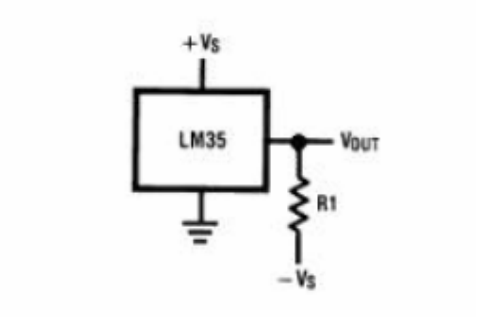
Alçak çıkış empedansı, 1 mA' lik yük için 0.1 W.

3.6.3 Tipik uygulamalar

LM35 sıcaklık sensörünün tipik uygulamaları Şekil 3.1 ve Şekil 3.2' de verilmiştir.



Şekil 3.1 Temel Santigrad sıcaklık sensörü, (+2°C ile +150°C)



Şekil 3.2 Santigrad sıcaklık sensörü tam genlikte

$R_1 = -V_S/50\ \mu A$ seçiniz.

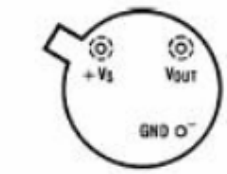
V_{OUT}=+1,500 mV, +150°C' de

= +250 mV, +25°C' de

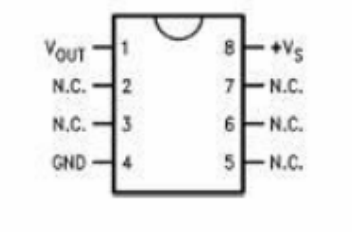
= −550 mV, −55°C' de

3.6.4 Bağlantı diyagramları

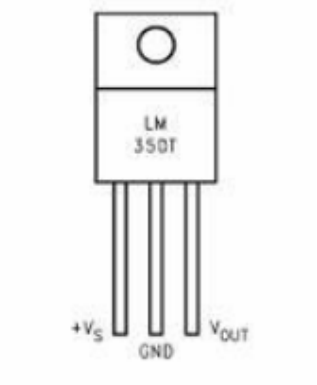
LM35 sıcaklık sensörünün değişik paket tipleri ve uç bağlantıları (Şekil 3.3a, b, c ve d) bulunmaktadır.



Şekil 3.3a Metal kutulu paket TO-46, alttan görünüşü



Şekil 3.3b Küçük yüzey montajı paketi şeklinde, SO-8, üstten görünüş

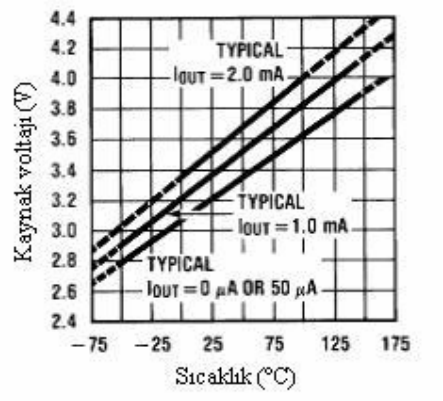


Şekil 3.3c Plastik paket, TO-220

Şekil 3.3d Plastik paket, TO-92, alttan görünüş

3.7 Tipik Performans Karakteristiği

Kaynak voltajındaki değişime karşılık çıkış voltajına karşılık gelen sıcaklık değerleri, çıkış akımının fonksiyonu olarak Şekil 3.4' de görülmektedir.



Şekil 3.4 Minimum kaynak voltajına karşılık sıcaklık

3.8 Uygulamalar

LM35 diğer entegre devre sıcaklık sensörlerine benzer şekilde kolayca uygulanabilir.Bir yüzeye tutturulabilir veya yapıştırılabilir, sıcaklığı yüzeyin sıcaklığının 0.01°C kadar bir sıcaklık değerinde olacaktır. Bu çevre hava sıcaklığının, yüzey sıcaklığı ile aynı olduğu anlamına gelir. Eğer hava sıcaklığı yüzey sıcaklığından çok daha yüksek veya alçak ise, LM35' in gerçek sıcaklığı yüzey sıcaklığı ve hava

sıcaklığının ortalarında bir değer alcaktır. Bu özellikle TO-92 plastik paket tipi için doğrudur, bu paket tipinde bakır uç ısıyı eleman içerisine taşıyan temel ısıl yoldur, bu nedenle bu elemanın sıcaklığı yüzey sıcaklığından çok hava sıcaklığına daha yakın olacaktır [20].

Bu problemi minimize etmek için LM35' i monte ederken elemanın ilgili yüzeyin sıcaklığı ile aynı sıcaklıkta olmasına dikkat edilmelidir. Bunun en basit yolu bu bağlantıları reçine ile kaplamaktır, bu bağlantı ve uçların yüzeyle aynı sıcaklıkta olmasını sağlar, bu da LM35' in sıcaklığının hava sıcaklığından etkilenmemesini sağlar.

4. NEM

Nem hava veya diğer gazlardaki su buharı içeriği olarak tanımlanır. Nem genellikle mutlak nem olarak ölçülür. Mutlak nem, su buharının kütlesinin hava veya gazın hacmine oranı olarak tanımlanır. Yoğuşma noktası bir gazın yoğunlaşarak sıvı hale geçmeye başladığı sıcaklık ve basınç değeri olarak tanımlanır. Bağıl nem, RH, ise aynı sıcaklık ve basınçta havadaki nem oranının doymuş nem oranına oranı olarak tanımlanır [18].

4.1 Nem Niçin Önemlidir

İç ortamda % 35-55 arasındaki bağıl nem oranı normal kabul edilir. %45 civarındaki bağıl nem idealdir. %35'in altındaki ortamlar "kuru"dur ve istenmez. %55'in üzerindeki ortamlar ise "yaş" olarak kabul edilir.

ASHRAE, iç ortam havasında mikroorganizmaların oluşmasına ve yayılmasına engel olacak maksimum bağıl nem oranını %60 olarak belirlemiştir. Evinizde ideal nemi kontrol etmek için burun kuruluğu ve statik elektrik anahtar olabilir. Uygun nemin kontrolü aynı zamanda bakterilerin, virüslerin, küf, toz ve akar oluşumlarının çoğalmasını engellemek için de önemlidir. Günümüzde ısı yalıtımını sağlamak için daha fazla sızdırmaz yapılan evlerde kış mevsiminde çok fazla nem sorun oluşturabilir. Örneğin, kışın aşırı nem camlarda ve kapı pervazlarında su damlamalarına ve terlemelere neden olur. Bu koşullarda pencerelerde ve kapılarda, köşelerde, bodrum döşemesinin yakınlarında zamanından önce bozulmalar ve küflenmeler oluşabilir.

4.2 Nem Sensörü Tipleri ve Teknolojileri

Yarıiletken teknolojisindeki son gelişmeler nem sensörlerini oldukça doğru bir şekilde ölçen, dayanıklı ve maliyet olarak da uygun duruma getirdi. Yaygın olarak kullanılan nem sensörleri kapasitif, direnç ve ısıl iletkenliğe bağlı olarak ölçüm yaparlar [18].

4.2.1 Kapasitif RH sensörler

Kapasitif RH sensörler endüstride ve ticari amaçlı olarak sıkça kullanılır. Sıcaklığın bu sensörlerin çalışmasına etkisi çok az olduğundan, oldukça büyük bir sıcaklık genliğinde sıcaklık kompanzasyonu yapılmadan kullanılabilir. Kapasitif bir RH sensörde dielektrik sabitindeki değişme ortamın bağıl nemi ile doğru orantılıdır [18].

4.2.2 Rezistif nem sensörleri

Rezistif nem sensörleri empedansdaki değişimi ölçerler, bu değişim genellikle nem ile üstel bir şekilde ters orantılıdır. Empedansı değişen ortam ise iletken polymer veya tuzdur.

4.2.3 Isıl iletkenlikle nem sensörleri

Isıl iletkenlikle nem sensörleri (Mutlak nem sensörleri olarak da bilinirler) kuru hava ve su buharı içeren havanın ısıl iletkenliği arasındaki farkı hesaplayarak mutlak nemi ölçerler. Bu sensörler iki negatif sıcaklık katsayılı termistör elemanının bir dc köprü devresinde kullanılması ile oluşturulur. Elemanlardan biri kuru nitrojen içerisinde olurken, diğeri nemi ölçülecek ortamda bulunur. Bu iki termistörün dirençleri arasındaki fark mutlak nem ile doğru orantılıdır [18].

4.3 Nem Sensörlerinin Seçimi ve Belirlenmesi

Aşağıdaki kısımda bir sensörü diğerinden farklı kılan sıcaklık, doğruluk ve değiştirilebilirlik gibi özellikler üzerinde durulmuş ve her bir sensörün avantaj ve dezavantajları da belirtilmiştir.

4.3.1 Nem sensörlerinin seçimi

Nem sensörlerinin seçiminde dikkate alınması gereken hususlar şunlardır [18].

a) Doğruluk
b) Değiştirilebilirlik
c) Tekrarlanabilirlik
d) Kararlılık
e) Büyüklüğü ve paket tipi

f) Fiyatının uygunluğu
g) Kalibrasyonu
h) Sensörün değiştirilme maliyeti

4.3.2 Kapasitif RH sensörlerin seçilmesi

Kapasitif RH Sensörlerin uygulaması geniş bir genlikte yer almaktadır. Bunalar arasında

a) Yazıcılar
b) Mikrodalga fırın, soğutucular ve elbise kurutucuları
c) HVAC
d) Meteoroloji istasyonları
e) Endüstri ve gıda işleme cihazları

Yarıiletken tasarımındaki son gelişmelerin sağladığı avantajlar dolayısı ile bir çok kapasitif sensörün uzun bir süre için histerisizi minimum değerdedir [18].

4.3.3 Dirençli nem sensörlerinin seçilmesi

Dirençli sensörler küçük, düşük maliyetli nem sensörleri olup, uzun bir zaman aralığında kararlı ve kolaylıkla değiştirilebilir olma gibi özelliklere sahiptirler. Bir çok endüstriyel, ticari ve konutla ilgili kontrol ve gösterge ürünleri uygulamaları için uygundurlar.

Dirençli sensörler nem değişikliklerine lineer olmayan bir cevap vermelerine karşılık, bu cevaplar analog ve dijital metotlarla lineer yapılabilir. Tipik olarak direnç değerleri birkaç kilo ohmdan 100 M Ω' a kadar değişir. Dirençli sensörlerin çalışma genliği –40°C ile 100°C arasındadır [18].

4.4 Arabirimleme ve Tasarım Bilgisi

4.4.1 Sıcaklık ve nemin etkisi

Absorbsiyon tabanlı bütün bütün nem sensörlerinin (kapasitif, rezistif) çıkışı hem sıcaklık hem de % RH' den etkilenir. Bu nedenle daha yüksek bir doğruluk veya daha geniş bir sıcaklık genliği için sıcaklık kompanzasyonu kullanmak gerekir.

Nem sensörlerini sıcaklığı kompanze edecek şekilde kalibrasyon ederken, en iyi yol sıcaklık ölçümünü, nem sensörünün aktif alanına oldukça yakın değerlerinde gerçekleştirmektir, örneğin aynı nem derecesi içerisindeki mikro ortamlarda [18].

4.4.2 Yoğuşma ve ıslaklık

Sensörün aktif alanının yüzey sıcaklığı onu çevreleyen ortamdaki gazın çiy noktasının altına düştüğü zaman yoğuşma başlar. Yüzey sıcaklığı anlık olarak da ortamın çiy noktasının altına düştüğü zaman sensör üzeride (veya herhangi bir yüzeyde) yoğuşma oluşur. Nem seviyesinin % 95' in üzerinde olduğu çalışma ortamlarında sensörün yakınlarındaki küçük sıcaklık değişimleri sebebi bilinmeyen bir şekilde yoğuşmaya sebep olur. Yoğuşmanın hızlı bir şekilde olacağı yüksek nemli ortamlarda, suyun buharlaşması yavaş olur (öyle ki sensörün yüzey sıcaklığı ortamın çiy noktasından biraz büyük değerdedir).

4.5 SHT11 Sıcaklık ve Nem Sensörü

Bağıl nem ve sıcaklık sensörü olan SHT11 aşağıdaki özellikler sahiptir.

. Çiy noktası

. Tamamen kalibrasyon edilmiş sayısal çıkış

. Oldukça uzun bir süre karalılık

. Harici dış elemanlar gerektirmez

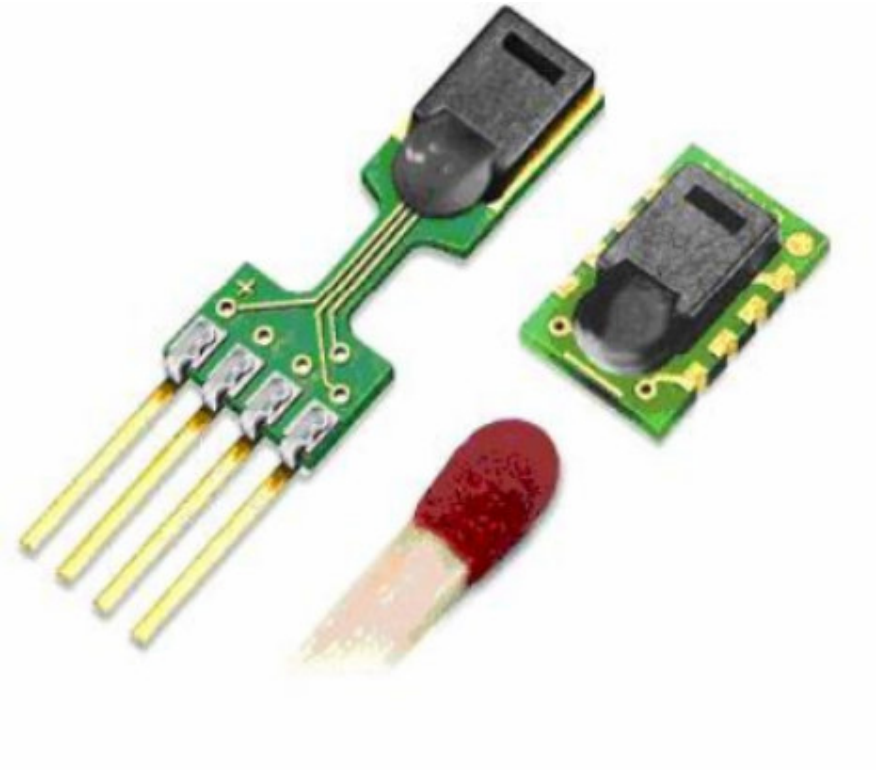
. Son derece çok az güç tüketimi

. Yüzeye monte edilebilir

. Küçük hacmimlidir

. Güç azatlımı otomatiktir

Şekil 4.1’ de SHT11 sensörünün dış görünüşü görülmektedir.



Şekil 4.1 SHT11 sensörünün dış görünüşü

4.5.1 SHT11 ürün özeti

SHT11 tek bir yongalık bağıl nem ve sıcaklık sensörü olup, kalibre edilmiş dijital çıkış vermektedir. Patentli mikro-makinalamalı (CMOSens teknolojisi) endüstriyel CMOS işleminin uygulanması, yüksek güvenilirlik ve mükemmel uzun süreli karalılık sağlar. Cihaz bağıl nem için kapasitif polimer algılayıcı eleman ve sıcaklık sensörü olarak da band geçişi içermektedir. Her ikisi de aynı yonga üzerinde 14 bitlik analog sayısal dönüştürücü ve seri bir ara birim devresi ile birleştirilmiştir. Bu işlemin sonucu

oldukça üstün bir sinyal kalitesi, hızlı bir cevap zamanı ve dıştaki bozucu etkilere karşı (EMC) duyarsızlığının çok rekabet edebilir bir fiyatla sunulmasıdır [21].

Her bir SHT11 entegresi kesin doğruluklu bir nem ortamında tek tek kalibre edilir. Kalibrasyon katsayıları OTP bellek içerisine programlanır. Bu katsayılar ölçüm süresince, sensörden gelen sinyalleri kalibre etmek için iç kısımda kullanılır [21].

İki hatlı seri arabirim ve içteki voltaj regülasyonu kolay ve hızlı bir şekilde sistem entegrasyonuna izin verir. Oldukça küçük bir hacimde oluşu ve çok az güç tüketimi onun bir çok uygulamalarda tek seçenek olmasını sağlar. Eleman ya yüzeye monte edilebilir LCC (Uçsuz yonga taşıyıcı) veya takılabilir tek hatlı 4 pinli paket olarak kullanıcılara sunulmuştur. İstek halinde müşterilerin belirleyeceği paketleme opsiyonu da olabilir.

4.5.2 Uygulamalar

. HVAC

. Otomobil

. Tüketici ürünleri

. Meteoroloji istasyonları

. Nemlendiriciler

. Nem alıcılar

. Test ve Ölçüm

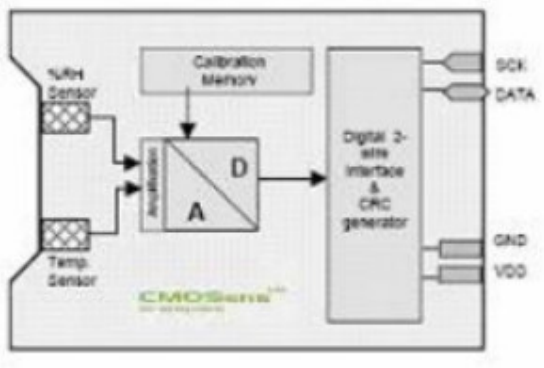
. Data Yükleme

. Otomasyon

. Beyaz eşya

4.5.3 Blok diyagramı

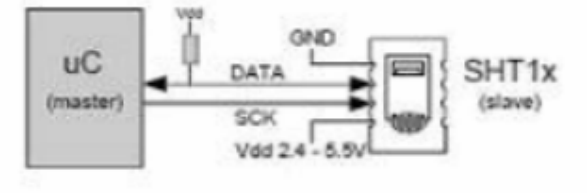
SHT11 sensörünün blok diyagramı Şekil 4.2' de görülmektedir [21].



Şekil 4.2 SHT11 sensörünün blok diyagramı

4.5.4 Ara birimlemenin belirlenmesi

Şekil 4.3' de SHT11 sensörünün mikrodenetleyici ile ara birimlendiği tipik bir uygulama devresi görülmektedir [21].



Şekil 4.3 SHT11 sensörünün ara birimlendiği tipik bir uygulama devresi

4.5.5 Güç uçları

SHT11 2.4 ile 5.5 V arasında bir kaynak voltajına gereksinim duyar. Eleman güç uygulandıktan sonra "uyku" durumuna ulaşması için 11 ms' ye ihtiyaç duyar. Bu zamandan önce hiçbir komut gönderilmeyecektir. Güç kaynağı uçları olan Vdd ve GND 100 nF' lık bir kondansatör ile dekuplaj edilebilir [21].

4.5.6 Seri ara birimleme

SHT11' in seri ara birimlemesi sensörün okuması ve güç tüketimi için en iyi duruma getirilmiştir, sensör I^2C ara birimleme ile uyumlu değildir.

4.5.7 Seri saat sinyali girişi, SCK

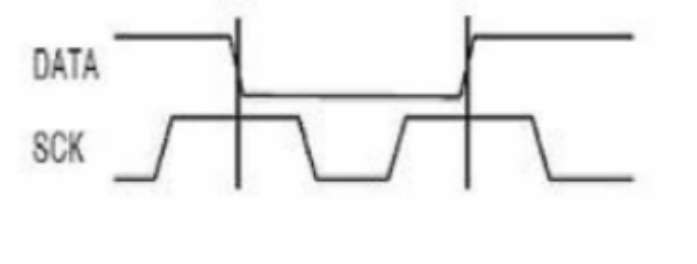
Seri saat girişi olan SCK mikrodenetleyici ile SHT11 arasındaki haberleşmeyi senkronize etmek için kullanılır. Ara birimleme tamamen statik mantık içerdiğinden, SCK frekansı için minimum bir değer yoktur.

4.5.8 Seri veri

Üç durumlu veri ucu, veriyi elemandan içeri veya elemandan dışarı transfer etmede kullanılır. Veri seri saat sinyalinin, SCK, azalan kenardan sonra değişir ve yükselen kenarında geçerli olur. İletim süresince SCK yüksek değerde iken veri hattı kararlı kalmalıdır. Sinyalin memnuniyetsizliğini önlemek için mikrodenetleyici yalnızca alçak veri değerlerini sürecektir. Sinyali yüksek değere çekmek için dışardan bir pull-up direnci (10KΩ) gereklidir (Şekil 4.3). Pull-up dirençleri mikrodenetleyicilerin I/O devrelerine çoğunlukla dahil edilir [21].

4.5.9 Bir komutun gönderilmesi

Bir iletimin başlatılabilmesi için, bir "iletim başla" dizisinin verilmesi gerekir. Bu dizi SCK' nın yüksek değerinde data hattının alçak değerde olması ve onu takip eden alçak bir SCK değeri ve tekrar SCK' nın yüksek değerinde datanın yükselen değer alması. Şekil 4.4' de "iletim başla" dizisi görülmektedir [21].



Şekil 4.4 "iletim başla" dizisi

4.5.10 Sıcaklığın etkisi

Bir gazın bağıl nemi onun sıcaklığına bağlıdır. Bu nedenle nem sensörlerini bağıl nemi ölçülen havanın sıcaklığı ile aynı sıcaklık değerinde tutmak gerekir. Eğer SHT11 elektronik elemanların ısı verdiği bir PCB üzerinde olması gerekiyorsa elemanlardan oldukça uzakta ve ısı kaynağının altında olmalı, bulunduğu bölüm oldukça iyi havalandırılmalıdır. SHT11 ile PCB' nin geri kalan kısımlarındaki bakır tabakalardaki ısı iletimini azaltmak için bu alanlar minimize edilmelidir. Şekil 4.5 SHT11' in PCB üzerine monte edilmesini göstermektedir [21].

Şekil 4.5 SHT11' in PCB üzerine monte edilmesi

4.5.11 Zar

Sensörü korumak ve kirlerin sensörün bulunduğu yuvaya girmesini önlemek için bir zar kullanılabilir. Bu zar aynı zamanda kimyasal buhar konsantrasyonlarının da tepe değerlerine ulaşmasını önleyecektir. Sensörün cevap zamanının optimum olabilmesi için zarın arkasında tutulan hava hacminin minimumda tutulması gerekir.

4.5.12 Işık

SHT11 ışığa duyarlı değildir. Uzun süreli doğrudan güneş ışığı altında kalması veya kuvvetli ultraviyole radyasyon altında kalması bulunduğu kabı yıpratır [21].

5. KONTROL SİSTEMİ

Bir kontrol sistemi sistemdeki diğer değişkenin değerini işleterek istenilen sonucu koruyan bileşenler grubudur. Bir kontrol sistemi istenilen değeri koruyan herhangi bir bileşenler grubudur, çok çeşitli bileşenler tek bir kontrol sisteminin parçası olabilirler, bu bileşenler elektrik, elektronik, mekanik, hidrolik, pnömatik, insan olabildiği gibi bunların herhangi bir kombinasyonu da olabilir.

Mikroelektronikteki gelişmelerin sonucu olarak, bugün pratik olarak bütün kontrol sistemleri mikroişlemci ve karmaşık mikrodenetleyici tabanlı olarak oluşturulmaktadır. Bilgisayar kontrollü sistemlerin kullanılması ile analog sistemlerden daha yüksek performansın ve bununla birlikte yeni fonksiyonerliğin elde edilebileceği K.J. Aström vd. [22] tarafından ifade edilmektedir.

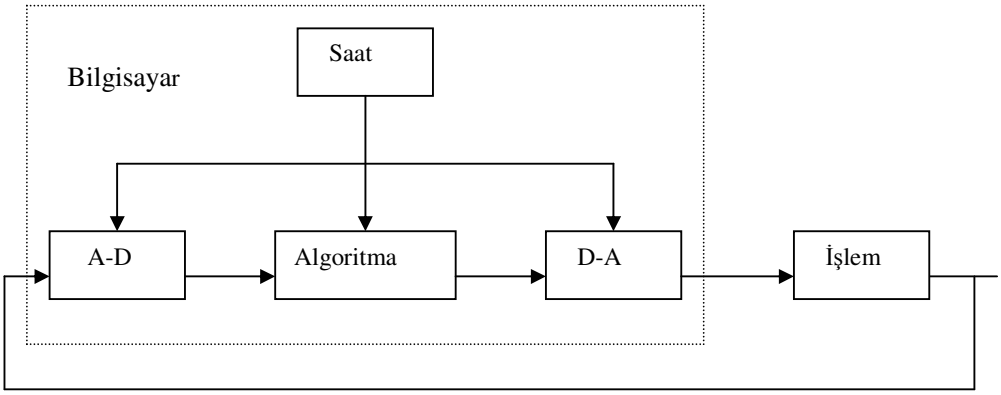
Wong vd. [23] 21. yüzyılda bina otomasyonu ile ilgili olarak yapmış oldukları çalışmada, bina otomasyonunun geçmişi, bugünü ve geleceği ile ilgili bir giriş yaparak, günümüzde yapılan bina otomasyon sistemlerinde mini bilgisayarların, merkezi işlem ünitelerinin (CPU) ve programlanabilir mantık denetleyicilerinin (PLC) artan bir şekilde kullanımından söz ederek, PC tabanlı ev kontrolünün günümüzde uygulamalarından özellikle güvenlikle ilgili sistemlerde olduğunu belirtmektedir. Bu tür bilgisayarlı kontrol uzaktan kontrol sistemleri ve akıllı kontrol sistemlerinin geliştirilmesine de yardımcı olmaktadır.

5.1 Bilgisayarlı Kontrol

Bugün yapılan pratik olarak bütün kontrol sistemleri, bilgisayarlı kontrole dayanır. Bu nedenle bilgisayar kontrollü sistemleri iyi anlamak önemlidir.

Bilgisayar kontrollü bir sistem şematik olarak şekil 5.1' deki gibi tanımlanabilir. İşlemin çıkışı olan y(t) bir sürekli zaman sinyalidir. Çıkış analogdan dijitale dönüştürücü (A-D) ile dijitale dönüştürülür. A-D dönüştürücü seçime bağlı olarak bilgisayar içerisinde veya ayrı bir birim olarak düşünülebilir. Çevirme işlemi örneklem zamanlarında, $t_{k,}$ yapılır. Bilgisayar dönüştürülen sinyali, $\{y(t_k)\}$, bir sayı dizisi olarak ifade eder, ölçümleri bir algoritmaya bağlı olarak yapar ve $\{u(t_k)\}$olarak yeni bir sayı dizisi olarak verir. Bu dizi bir dijitalden analoga (D-A) dönüştürücü ile analog sinyale dönüştürülür. Olaylar bilgisayar içerisindeki gerçek zaman saati ile senkronize edilir.

Dijital bilgisayar zaman içerisinde ardışık olarak çalışırken, her bir işlemde biraz zaman alır.



Şekil 5.1 Bilgisayar kontrollü sistemin şematik diyagramı

Bilgisayarla işlem kontrolünü geliştirmek için dört önemli alan vardır.

i) İşlem bilgisi
ii) Ölçme teknolojisi
iii) Bilgisayar teknolojisi
iv) Kontrol teorisi

5.2 Bir kontrol sisteminde değişken isimleri

Kontrol edilen değişken, (C), kontrol edilmesi gereken işlemin çıkış değişkenidir. Bir süreç kontrol sisteminde, kontrol edilen değişken üretimin kalitesini belirler, yaygın olarak kullanılan işlem değişkenleri, pozisyon, hız, sıcaklık, basınç, seviye ve akış miktarıdır.

Ayar noktası, (SP), kontrol edilen değişkenin istenilen değeridir.

Ölçülen değişken, (C_m), kontrol edilen değişkenin ölçülen değeridir. Ölçme araçlarının çıktısıdır ve genellikle kontrol edilen değişkenin gerçek değerinden küçük miktarlarda farklıdır.

Hata sinyali, (E), ayarlama noktasıyla kontrol değişkeninin ölçülen değeri arasındaki farktır. Şu formülle hesaplanır: $E = SP - C_m$.

Bozucu etki değişkenleri, (D), kontrol sistemi tarafından kontrol edilmeyen ve kontrol edilen değişkeni etkileyen süreç giriş değişkenleridir. Bozucu etki değişkenleri süreç üzerindeki işlem değişkenini değiştirebilirler bu yüzden kapalı döngü kontrol sisteminin kullanılma nedenidirler.

5.3 Kontrol Sistemlerinin Sınıflandırılması

1. Geri beslemeye bağlı olarak,
    a) Açık döngü kontrol sistemleri
    b) Kapalı döngü kontrol sistemleri

Bir açık döngü kontrol sisteminde kontrolü gerçekleştirmek için işlemin sonucu istenilen değerle karşılaştırılmaz. Bunun yerine, istenilen değeri elde etmek için daha önceden bir çeşit ayarlama işlemine veya hesaplamasına göre belirlenmiş değerler kullanılır.

Kapalı döngü kontrol sistemlerinde gerçek sonuç geri besleme ile istenilen değerle karşılaştırılır. Karşılaştırma sonucunda elde edilen hata sinyali kontrol ediciye giriş sinyali olarak verilir. Kontrol edicinin çıkışı kontrol edilen sistemin girişidir, bu nedenle ölçülen sinyal değeri kontrol edilen sistemin çıkışından girişine beslenir

Açık döngü kontrol sistemleri fazla kullanılmaz, fakat kapalı döngü kontrol sistemleri yaygın olarak kullanılır. Kapalı döngü kontrolün temel avantajı sürecin daha doğru kontrolünü sağlamasıdır. Dezavantajları ise, kapalı döngü kontrol, açık döngü kontrolden daha pahalıdır ve kapalı döngü kontrolün geri besleme özelliği sistemin kararsız olmasına yol açabilir.

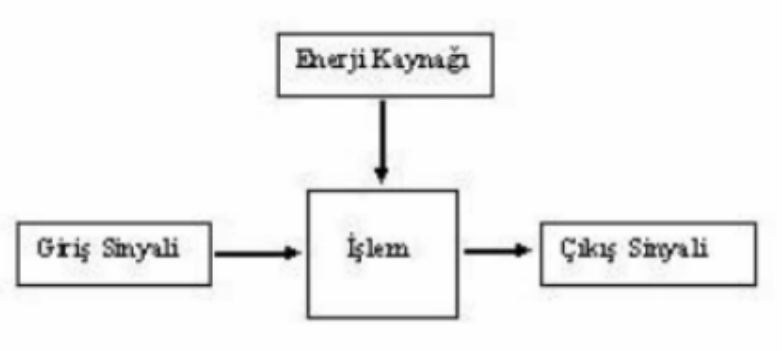
2. Sinyal tiplerine bağlı olarak,
    a) Analog kontrol sistemleri
    b) Dijital (Ayrık) kontrol sistemleri

Analog sinyal sürekli bir sinyal olup, zaman içerisinde sürekli bir değer alır. Dijital sinyal ise ayrık biçimde değişir ve sınırları içinde sadece belirli ayrık değerler alabilir.

5.4 Blok Diyagramlar ve Transfer Fonksiyonları

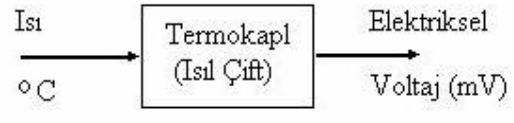
5.4.1 Açık döngü kontrol

Şekil 5.2' de bir açık döngü kontrol sisteminin blok şeması verilmiştir.



Şekil 5.2 Açık döngü bir kontrol sisteminin blok şeması

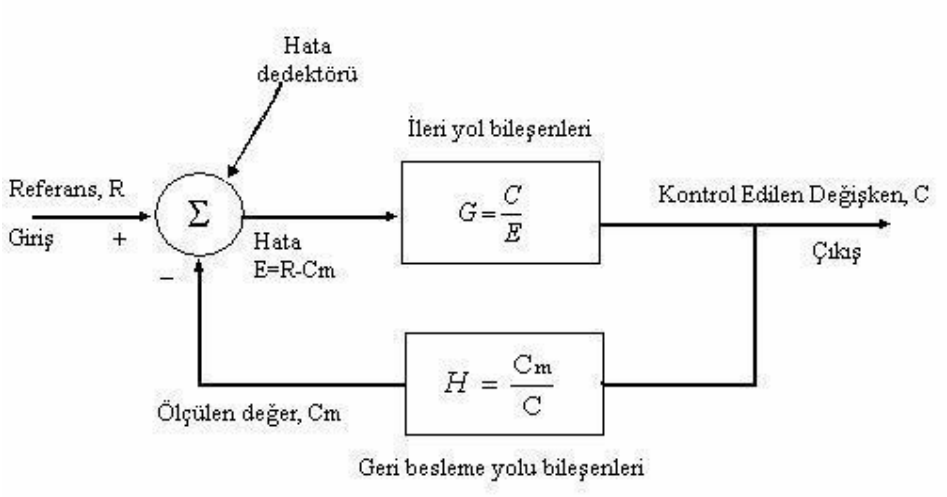
Eğer giriş sinyalini ve transfer fonksiyonunu biliyorsak, çıkış sinyalini elde etmek için giriş sinyalini transfer fonksiyonu ile çarparak hesaplayabiliriz. Bir kontrol sisteminin blok şeması çeşitli bloklardan oluşur, şekil 5.3' de ısı sensörü olarak kullanılan termokaplın (ısıl çift) blok gösterimi verilmiştir.



Şekil 5.3 Kontrol sistemi bileşenlerinin blok gösterimi

5.4.2 Kapalı döngü kontrol

Şekil 5.4' de negatif geri beslemenin yapıldığı kapalı döngü kontrol sisteminin blok diyagramı görülmektedir. Kapalı döngü terimi geri besleme yolu tarafından oluşturulan bu döngüyü ifade etmektedir.



Şekil 5.4 Negatif geri beslemeli kapalı döngü kontrol sisteminin blok diyagramı

Negatif geri beslemenin yapıldığı Şekil 5.4' deki kapalı döngü kontrol sisteminin transfer fonksiyonunu hesaplamak istersek,

Hata Sinyali = Referans değeri – Kontrol edilen değişkenin ölçülen değeri

$$E(t) = R(t) - C_m(t)$$

Kontrol edilen değişkenin değeri = Hata sinyali x Sistemin transfer fonksiyonu

$$C(t) = E(t) \times G$$

Kontrol edilen değişkenin ölçülen değeri	=	Kontrol edilen değişken	x	ölçüm cihazının fonksiyonu

$$C_m = CH$$

$$C = (R - C_m)G$$

$$C = (R - CH)G$$

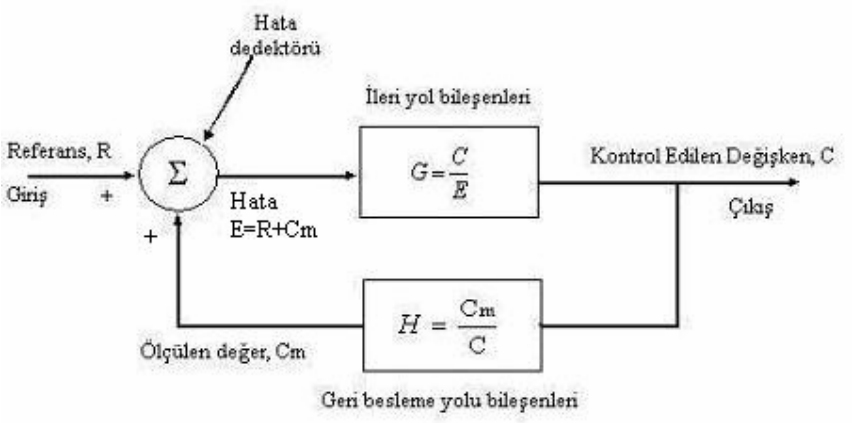
$$C + CGH = RG$$

$$C(1+GH) = RG$$

$$\frac{C}{R} = \frac{G}{1+GH}$$

Yukarıdaki eşitlik kapalı döngü kontrol sisteminin transfer fonksiyonudur. İleri yol transfer fonksiyonu, (G), yükselticiler, jeneratörler, motorlar gibi sistem bileşenlerini içerir. Geri besleme yolu transfer fonksiyonu, (H), ise genellikle kontrol edilen değişkeni hata dedektörüne giriş için uygun sinyale çeviren pasif bir cihazdır.

Pozitif geri beslemeli kontrol sistemlerinde işlemin ölçülen değeri hata sinyalini oluşturmak için referans sinyaline eklenir. Şekil 5.5 pozitif geri beslemeli kapalı döngü kontrol sisteminin blok şemasını göstermektedir. Pozitif geri besleme için transfer fonksiyonunu çıkarmak istersek,



Şekil 5.5 Pozitif geri beslemeli kapalı döngü kontrol sistemi

$$E = R + C_m$$

$$C = EG$$

$$C_m = CH$$

$$C = (R + C_m)G$$

$$C = (R + CH)G$$

$$C - CGH = RG$$

$$C(1-GH) = RG$$

$$\frac{C}{R} = \frac{G}{1-GH}$$

Şekil 5.5' deki ayar noktası (set point, SP) süreç kontrol sistemine giriş olup, kontrol edilen değişken, (C), ise çıkıştır. Geri besleme yolu bir bileşenden oluşur, transfer fonksiyonu H olan ölçüm cihazı. İleri yolun transfer fonksiyonu ise G olarak gösterilmiştir.

5.5 Otomatik Kontrolün Faydaları

Kontrol sistemleri günümüzde gittikçe daha da önem kazanmaktadır. Biz onları hayatın onlarsız hayal edilemez uzantıları olarak düşünürüz. Otomatik kontrol yetenekli işçileri rutin işlerden kurtararak ve her işçinin yaptığı iş miktarını artırarak, iş verimliliğini artırmıştır. Kontrol sistemleri üretilenlerin kalitesini geliştirir ve ürünlerin birbirinin aynı olmasını sağlar. Beğendiğimiz bir çok ürünün otomatik kontrolsüz nerdeyse üretilmesi olanaksızdır. Özetle otomatik kontrolün faydaları aşağıdaki şekilde sıralayabiliriz.

i) Artırılmış verimlilik
ii) Geliştirilmiş kalite
iii) Artırılmış etkinlik
iv) Güç yardımı
v) Güvenlik
vi) Konfor ve hayatı kolaylaştırma

5.6 Kapalı bir Ortamda Isıtma Nemlendirme Kontrolü için Sistemin Dinamik Modeli

5.6.1 Giriş

Kapalı bir ortamda ısıtma, nemlendirme kontrolü için sistemin dinamik modelini; ısıtma, havalandırma ve hava şarlandırma, HVAC, (Heating Ventilation and Air Conditioning) sisteminin dinamik modelini baz alarak modellendirebiliriz.

Bir ısıtma, havalandırma ve hava şartlandırma (HVAC) sisteminin dinamik performansının güç ve enerji tüketimi ve iç kısımlardaki hava kalitesi açısından büyük önem taşıdığı bilinmektedir. Tasarım aşamasında sistem performansını çalışmak için sistem bileşenlerinin yaklaşık matematiksel modelinin elde edilmesi gereklidir. Bunlara ek olarak, binalar için iyileştirilmiş enerji kontrol sistemlerini geliştirmede verimli kontrol stratejileri de önemli rol oynarlar [1].

Bir HVAC alanının tasarımında en önemli kriter enerji verimliliği ve iç kısımlardaki iklim şartlarıdır. Bu iki kriterin elverişli kombinasyonu sahanın istenilen şekilde kontrolünü sağlar.

HVAC sistemleri için başarılı kontrol sistemlerinin tasarımı öncelikle sistemin iyi bir dinamik modelinin ve onun davranışını tanımlayan matematiksel eşitliklerin olabilirliğine bağlıdır. Dağılmış parametreleri ile bir HVAC sistemin kompleksliği, birbirleriyle etkileşimi ve çok değişkenlik onu kontrol kalitesini iyileştirmek için tam bir matematiksel modelini elde etmeyi oldukça zor kılar.

Aşağıdaki çalışmada bir HVAC sistemi kontrol etmek için dinamik modelinin türetilmesi incelenmiştir. HVAC sistem bir oda, nemlendirici, ısıtıcı, soğutucu, fan ve kanallar içermektedir. Bölgenin modellenmesinde soğutma ve ısıtma işlemleri için enerji ve kütle dengesi eşitlikleri uygulanmıştır. Bunlara ilave olarak, matematiksel modeli kabul edilebilir bir komplekslik seviyesinde tutmak ve doğru simülasyon sonuçları elde etmek için ısıtma ve soğutma sistemleri ayrı ayrı analiz edilmiştir. İç kısımlardaki sıcaklık ve nem oranlarını set noktası değerlerinde tutmak için bir hava işleme ünitesince PID kontrol uygulanmıştır. Isıtma sistemi modeli bir hava şartlandırma odası, ısıtma bobini, nemlendirici, fan ve kanallar içermektedir. Soğutma sistemi modeli ise aynı elemanları içermekle beraber ısıtma bobini ve nemlendirici

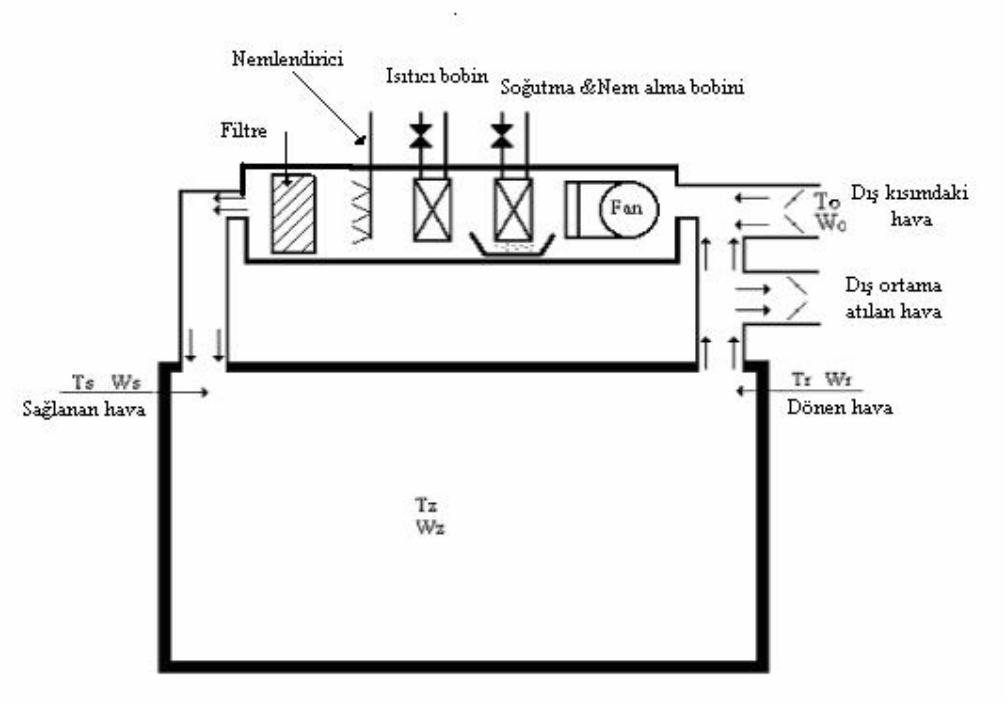
yerine soğutma – nem alma bobini kullanılmıştır. Pratik nedenlerden dolayı ayarlama tekniği olarak Ziegler-Nichols kuralı kullanılmıştır.

5.6.2 Sistem tanımlaması

Sistem modelinde düşünülen başlıca elemanlar, bir hava şartlandırma odası ve hava işleme ünitesidir, AHU (Air Handling Unit). Hava işleme ünitesi Şekil 5.6' da görüldüğü gibi bir fan, soğutma ve nem alma bobini, nemlendirici, filtre ve kanallar içermektedir. İki çalışma mevsimi düşünülmüştür, bunlar yaz ve kış çalışma mevsimleridir.

Yaz çalışma mevsiminde, sıcak ve nemli hava, hava işleme ünitesindeki soğutucu ve nem alıcı bobine doğru gider. Hava bobin içerisinden geçerken sıcaklığı azalır ve nemi su olarak yoğunlaştığından bağıl nem oranı azalır. Bölge içerisinde termostat bölgenin sıcaklığını algılayarak, soğutucu bobin içerisindeki suyun sıcaklığını azaltacak veya artıracak sinyali (şekilde görülmemektedir) kontrol edicinin vermesini sağlar. Nem alma işleminde, bölge içerisindeki nem oranını humidistat algılayarak yüksek bağıl nem değerlerinde, soğutucuya sinyal göndererek suyun sıcaklığını azaltır, böylece sağlanan hava içerisindeki su buharı yoğunlaşarak, bölge içerisindeki nem oranı da azalır.

Kış çalışma mevsiminde, soğuk ve nemli hava, hava işleme ünitesi içerisine girer. Termostat bölge içerisindeki sıcaklığı algılar ve kontrolediciye bir sinyal gönderir, kontrol edici ısıtıcı bobin içerisinden geçen suyun giriş sıcaklığını kontrol etmek için hata sinyalini kullanır. Son olarak hava nemlendirici içerisinden geçip, buhar oluşturarak hava içerisindeki nem oranı kontrol edilir.



Şekil 5.6 HVAC sistemin şematik diyagramı

Sistem içerisindeki hava akış hızını kontrol etmek için sönümlendiriciler ve / veya değişken akış fanları kullanılmıştır. Sistemdeki üç kontrol girişi ; sağlanan havanın sıcaklığı, sağlanan havanın nem oranı ve kütle akış hızı. Bu çalışmada, sağlanan kütle akış hızı sabit ve 0.24 kg/s olarak kabul edilmiştir. Hava sıcaklığı ve nem oranının istenilen değerleri karşılıklı olarak 22 ^{0}C ve 0.008 kg (su)/kg (kuru hava) olarak set edilmiştir. İlgili çıkışlar bölgenin sıcaklığı ve nem oranıdır.

5.6.3 Problemin analitik formülasyonu

5.6.3.1 Bölge modeli

Bölge kompleks bir ısıl sistem olduğundan, modeli tam bir teorik yaklaşımla formüle etmek pratik değildir (imkansızdır). Bölge modeli üç durum değişkeni ile karakterize edilir. Bölge sıcaklığı (Tz), duvarların iç kısımlarının sıcaklığı (Twi), bölgenin nem oranı (Wz). Bölge içerisindeki havanın tamamen karışmış olduğu kabul edilmiştir, bu nedenle bölge içerisinde sıcaklık dağılımı düzgün olduğundan, bölgenin dinamiği kümelenmiş kapasitans modeli ile ifade edilebilir. Bunlara ilave olarak, kuzey

duvarının bölge sıcaklığına etkisi güney duvarının etkisi ile aynı olduğu kabul edilmiştir, yine doğu duvarının etkisi ile batı duvarının etkisinin aynı olduğu kabul ediliştir. Zeminin bölge sıcaklığı üzerine etkisi yoktur. Havanın yoğunluğu sabit ve bölgenin sıcaklık ve nem oranı değişimlerinden etkilenmediği kabul edilmiştir. Bölge boyunca ve karışım kısmındaki basınç kayıpları ihmal edilmiştir. İnsanlar, ışık ve beklenmeyen hava şartları kontrol edilmeyen girişlerdir.

Yukarıdaki kabuller altında, bölgenin enerji ve kütle dengesini kapsayan eşitlikler aşağıdadır,

$$C_z \frac{dT_z}{dt} = f_{sa}\rho_a C_{pa}(T_{sa} - T_z) + 2U_{w1}A_{w1}(T_w - T_z) + U_R A_R (T_R - T_z) + 2U_{w2}A_{w2}(T_{w2} - T_z) + q(t) \tag{5.1}$$

$$C_{w1} \frac{dT_{w1}}{dt} = U_{w1}A_{w1}(T_z - T_{w1}) + U_{w1}A_{w1}(T_o - T_{w1}) \tag{5.2}$$

$$C_{w2} \frac{dT_{w2}}{dt} = U_{w2}A_{w2}(T_z - T_{w2}) + U_{w2}A_{w2}(T_0 - T_{w2}) \tag{5.3}$$

$$C_R \frac{dT_r}{dt} = U_R A_R (T_z - T_R) + U_R A_R (T_O - T_R) \tag{5.4}$$

$$V_z \frac{dW_z}{dt} = f_s (W_s - W_z) + \frac{P(t)}{\rho_a} \tag{5.5}$$

(5.1) nolu eşitlik bölge içerisindeki enerjinin değişim hızının, bölgeye iletim ve konveksiyon yolla transfer edilen enerji ile bölgeden çıkarılan enerjinin farkına eşit olduğunu ifade etmektedir. (5.2) – (5.4) eşitlikleri duvarlar içerisindeki enerji değişim hızının iç ve dış kısımdaki havanın sıcaklık farkından dolayı duvarlar içerisinden transfer edilen enerjiye eşit olduğunu. Benzer şekilde, (5.5) nolu eşitlik bölge içerisindeki nemin değişim hızının, bölge içerisine eklenen ve çıkarılan nem arasındaki farka eşit olduğunu ifade etmektedir.

5.6.3.2 Isıtma bobini modeli

Isıtma bobini sudan havaya ısı dönüştürücüdür, bu dönüştürücü bina içerisine havalandırma ile şartlandırılmış hava sağlar. Bobin içerisine sağlanan suyun kütle akış

hızı sabit ve 0.084 kg/s olarak kabul edilmiştir. Buna ilave olarak, bobin materyalinin son derece yüksek iletkenlikte olduğu böylece ısıl direncinin ihmal edilebilecek değerde olduğu düşünülmüştür. Isıtma bobinine sağlanan ısıtıcı su T_{wi} sıcaklığındadır. Isıtma bobininden ayrılan suyun sıcaklığı, T_{wo} sabit ve 10 ^{o}C' a eşit olduğu düşünülmüştür. Sıcak su ve soğuk hava arasındaki enerji dengesi aşağıdaki gibi ifade edilebilir.

$$C_{ah}\frac{dT_{co}}{dt} = f_{sw}\rho_w C_{pw}(T_{wi} - T_{wo}) + (UA)_a(T_o - T_{co}) + f_{sa}\rho_a C_{pa}(T_m - T_{co}) \tag{5.6}$$

Kütle dengesi ise,

$$V_{ah}\frac{\partial W_{co}}{dt} = f_{sa}(W_m - W_\infty) \tag{5.7}$$

(5.6) nolu eşitlik bobin içerisinden geçen havanın enerji değişim hızının ısıtıcı bobin içerisindeki suyun akış hızının eklediği enerji ve dönüş havası tarafından ortama transfer edilen enerjiye eşit olduğunu göstermektedir.

5.6.3.3 Nemlendirici modeli

Nemlendirme atmosferik havaya su buharı kütlesi transfer işlemidir, bu olayın sonucunda karışımda su buharında bir artış olur. Nemin son derece alçak değerde olması insan vücudu için istenilmeyen etkilere sahiptir. Hava şarlandırmada, havadaki nemin ölçümü ve kontrolü önemli bir safadır. Nemlendirici modeli için Kasahara [5] tarafından geliştirilen enerji ve kütle dengesi eşitliği aşağıdaki gibi ifade edilebilir.

$$C_h\frac{dT_h}{dt} = f_{sa}C_{pa}(T_{si} - T_h) + \alpha_h(T_o - T_h) \tag{5.8}$$

$$V_h\frac{dW_h}{dt} = f_{sa}(W_{si} - W_h) + \frac{h(t)}{\rho_a} \tag{5.9}$$

(5.9) nolu eşitlikte h(t) nemlendiricinin ürettiği nemli hava oranı hızı olup nem oranının bir fonksiyonudur.

5.6.3.4 Sensor Modeli

Sensörlerin fonksiyonu bölgedeki sıcaklık ve bağıl nemi ölçmek ve sistemin performansını artırmak için kontrol sistemine geri besleme sinyali vermektir. Basit olması için bu çalışmada sensörün birinci dereceden bir sistem ve zaman sabiti t' si ise 1 saniye olarak kabul edilmiştir. Bu nedenle eşitlik doğrudan aşağıdaki gibi yazılabilir.

$$T_{se}(s) = \frac{\tau_{se}}{\tau s_{es} + 1} T_{me}(s) \tag{5.10}$$

5.6.3.5 Fan Modeli

Birinci dereceden fan modeli seçildi ve hava sıcaklığındaki değişikliklerin havanın fiziksel özellikleri üzerine etkisinin ihmal edilebilir düzeyde olduğu kabul edildi. Bu nedenle fan modeli lineer olup, birinci dereceden transfer fonksiyonu ile ifade edilebilir. Motor akış içerisine yerleştirilirse, fan motorundan havaya ısı transfer edilir. Normal olarak bu ısı transferi hava sıcaklığının yaklaşık 1- 2 ^{o}C artmasına sebep olur. Fanın nem oranına etkisi olmadığına dikkat ediniz, çünkü havanın fan içerisinden geçerken bir kütle transferi olmamaktadır.

5.6.3.6 Karıştırma kutusu

Hava şartlandırma sistemlerinde hava akımlarının karıştırılması yaygın bir uygulamadır ve genellikle değişmez ve adiabatic şartlar altında gerçekleşir. Bu çalışmada, dönüştürme kısmında sürtünme kuvvetlerinin olmadığı kabul edilmiştir. Enerji ve kütle dengesi eşitlikleri aşağıdaki gibidir.

$$m_r C_{pa} T_r + m_o C_{pa} T_o = m_m C_{pa} T_m \tag{5.11}$$

$$m_r + m_o = m_m \tag{5.12}$$

burada

$$T_m = \frac{m_r T_r + m_o T_o}{m_r + m_o} \tag{5.13}$$

$$W_m = \frac{m_r W_r + m_o W_o}{m_r + m_o} \tag{5.14}$$

5.6.3.7 Kanal modeli

Kanal ünitesi için Clark vd. [2] tarafından geliştirilen geçiş modeli kullanılmıştır. İçeri doğru giden havanın sıcaklığı T_i ve dışarı doğru çıkan havanın sıcaklığı ise T_{out}' dur. Hava sıcaklığının değişim hızı ise,

$$\frac{dT_{out}}{dt} = \frac{(h_i + h_o) m_a C_p}{h_i M_c C_c} (T_{in} - T_{out}) \tag{5.15}$$

5.6.3.8 Soğutma ve nem alma bobini modeli

Soğutma bobini hava şartlandırma ünitesinin önemli bir parçasıdır ve birincil saha ile ikincil hava dağıtım sistemi arasında en önemli ara birimdir. Hava bobin içerisinden geçerken, soğuk dikey yüzeylerle temas haline gelerek, havadan tüp içerisinde akan suya doğru ısı transfer olur.

Soğutma bobininin öneminden dolayı, onun geçiş davranışını ve cevap karakteristiklerini çalışmak için birkaç model geliştirildi. Bu çalışmada, Elmahdy [9] tarafından geliştirilen soğutucu su bobininin geçiş modeli uygulandı. Hava sıcaklığı Ta ve nem oranı Wa, suyun sıcaklığının fonksiyonu olarak aşağıdaki eşitliklerle yaklaşım yapıldı,

$$T_a(T_w) = -0.0587(T_w)^2 + 1.773T_w + 1.1816 \tag{5.16}$$

$$W_a(T_w) = 3.2434x10^{-5} {T_w}^2 - 1.7972x10^{-5} T_w + 6.223x10^{-3} \tag{5.17}$$

Yukarıdaki eşitliklerin geçerlilik genliği Tw = 5 – 50 ^oC' dır. Bölge modeli için (5.1) – (5.5) eşitliklerinin Laplace dönüşümünü aldığımızda, aşağıdaki cebirsel eşitlikleri verir.

$$(C_{zs} + f_s\rho_a C_{pa} + 2U_{w1}A_{w1} + U_R A_R + 2U_w A_{w2})T_z(s)$$
$$= f_s\rho_a C_{pa}T_s(s) + 2U_{w1}A_{w1}T_{w1}(s) + U_R A_R T_R(s) + 2U_{w2}A_{w2}T_{w2}(s) + q(s) \quad (5.1a)$$

$$(C_{w1}s + 2U_{w1}A_{w1})T_{w1}(s) = U_{w1}A_{w1}T_z(s) + U_{w1}A_{w1}T_o(s) \quad (5.2a)$$

$$C_{w2}s + 2U_{w2}A_{w2})T_{w2}(s) = U_{w2}A_{w2}T_z(s) + U_{w2}A_{w2}T_o(s) \quad (5.3a)$$

$$C_R s + 2U_R A_R)T_R(s) = U_R A_R T_z(s) + U_R A_R T_o(s) \quad (5.4a)$$

$$(V_z s + f_{sa})W_z(s) = f_{sa}W_s(s) \quad (5.5a)$$

Bölge modelinin transfer fonksiyonu aşağıdaki gibi gösterilebilir,

$$T_z(s) = G_z(s)\beta T_s(s) + \gamma G_{w1}(s)[T_z(s) + T_o(s)] + \delta G_{w1}(s)[T_z(s) + T_o(s)] + \lambda G_R(s)[T_z(s) + T_o(s)] \quad (5.18)$$

$$W_z(s) = G_{wz}(s)W_s(s) \quad (5.19)$$

burada

$$\alpha = f_{sa}\rho_a C_{pa} + 2U_{w1}A_{w1} + U_R A_R + 2U_w A_{w2}, \qquad \beta = f_{sa}\rho_a C_{pa}$$

$$\gamma = 2U_{w1}A_{w1}, \qquad \delta = 2U_{w2}A_{w2}$$

$$\lambda = U_R A_R \qquad G_z(s) = \frac{1}{C_z s + \alpha}$$

$$G_{w1}(s) = \frac{1}{C_{w1}s + \gamma} \qquad G_{w2}(s) = \frac{1}{C_{w2}s + \delta}$$

$$G_R(s) = \frac{1}{C_R s + \lambda} \qquad G_{wz}(s) = \frac{f_{sa}}{V_z s + f_{sa}}$$

Bölgenin blok diyagramı Şekil 5.7' de görülmektedir.

5.6.3.9 Isıtıcı bobin modeli

Isıtıcı bobin modeli için (5.6) ve (5.7) eşitliklerinin Laplace dönüşümü,

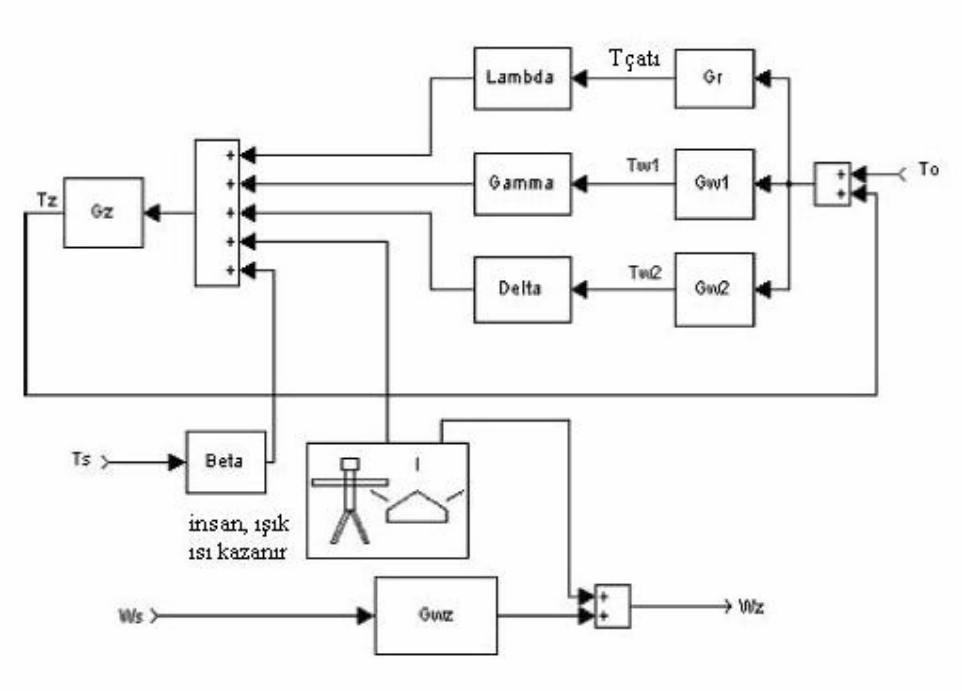
$$[C_{ah} s + f_{sa} \rho_a C_{pa} + (UA)_a T_\infty(s) = f_{sw} \rho_w C_{pw} [T_{wi}(s) - T_{wo}(s)] + (UA)_a T_o(s) + f_{sa} \rho_a C_{pa} T_m(s) \quad (5.6a)$$

$$(V_{ah} s + f_{sa}) W_\infty(s) = f_{sa} W_m(s) \quad (5.7a)$$

Bu nedenle

$$T_s(s) = G_s(s)\{ v[T_{wi}(s) - T_{wo}(s)] + \chi T_o(s) + \beta T_{si}(s)\} \quad (5.20)$$

$$W_s(s) = G_{ws}(s) W_{si}(s) \quad (5.21)$$



Şekil 5.7 Bölge modelinin blok diyagramı

burada

$$\xi = f_{sa}\rho_a C_{pa} + (UA)_a, \qquad v = f_{sw}\rho_w C_{pw}, \qquad \chi = (UA)_a$$

$$G_s(s) = \frac{1}{C_{ah}s + \xi}, \qquad G_{ws}(s) = \frac{f_{sa}}{V_{ah}s + f_{sa}}$$

(5.8) ve (5.9) nolu eşitliklerin Laplace dönüşümü

$$(C_h s + f_{sa}C_{pa} + \alpha)T_h(s) = f_{sa}C_{pa}T_{si}(s) + \alpha_h T_o(s) \qquad (5.8a)$$

$$(V_h s + f_{sa})W_h(s) = f_{sa}W_s(s) + \frac{h(s)}{\rho_a} \qquad (5.9a)$$

Bu nedenle

$$T_h(s) = G_h(s)[f_{sa}C_{pa}T_{si}(s) + \alpha_h T_o(s)] \tag{5.22}$$

$$W_h(s) = G_{wh}(s)\left[f_{sa}W_{si}(s) + \frac{h(s)}{\rho_a}\right] \tag{5.23}$$

burada

$$\psi = f_{sa}C_{pa} + \alpha_h, \qquad G_h(s) = \frac{1}{C_{hs} + \psi}, \qquad G_{wh}(s) = \frac{1}{V_{hs} + f_{sa}}$$

Kanalın enerji dengesi eşitliğinin (5.15) Laplace dönüşümü,

$$T_O(s) = G_{kanal}(s)T_{in}(s) \tag{5.24}$$

burada

$$G_{kanal}(s) = \frac{\tau}{s + \tau}, \qquad \tau = \frac{(h_i + h_o)m_a C_p}{h_i M_c C_c}$$

5.6.4 Kontrol edilmeyen sistemin geçiş cevabı

Sistemin geçerliliği ve onun bileşenlerinin zaman cevabını test etmek için bir simülasyon yapıldı. Açık döngü sistemin (kontrol edilmeyen) geçiş cevabı çalışıldı. Çevre şartlarının set noktası değerleri aşağıdaki gibidir:

(i) Çevre şartları:

Yaz çalışma mevsimi: To = 32 ^{o}C, Wo = 0.01251 kg/kg (kuru hava), Ts = 13 ^{o}C

Kış çalışma mevsimi: To = 5 ^{o}C, Wo = 0.00377 kg/kg (kuru hava), Ts = 25 ^{o}C

(ii) Hava kaynağının hacimsel akış hızı fs=0.192 m3/s.

(iii) Bölge içerisinde 0.15 kW' lık yük ile insan sayısının 2 ve 0.5 kW' lık yük ile 2 adet lamba olduğu düşünülmüştür.

(iv) t=0 olduğu zamanda başlangıç şartları yaz ve kış çalışma mevsimi için Tz(0)=To and Wz(0)=Wo olarak verilmiştir.

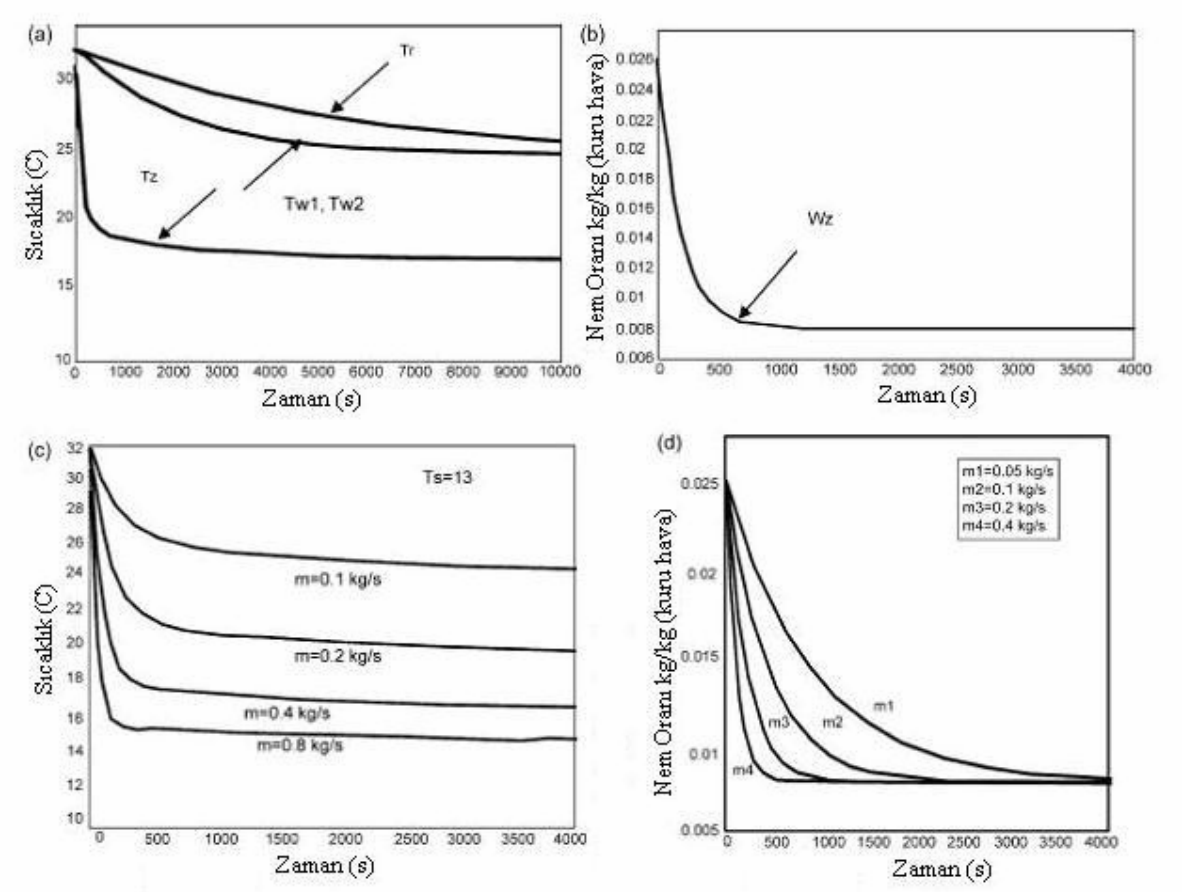
Bölgenin sıcaklığı değişmez durum şartlarına, bölgenin yüksek ısıl kapasitansından dolayı yavaş cevap vermesinden, yaklaşık 8000 s' de ulaşmaktadır. Yüksek ısıl kapasitansdan dolayı duvarların değişmez durum şartlarına ulaşması için daha fazla zamana ihtiyaç duyduğuna dikkat ediniz. Bu Şekil 5.8a' da yaz çalışma mevsimi için açıkça görülmektedir. Şekil 5.8b' de açıkça görülmektedir ki bölgenin nem oranı değişmez durum değerine yaklaşık 3000 s sonra ulaşmaktadır. Sağlanan havanın soğutulmasının hızlı olmasından nem oranı hızlıca azalmaktadır.

Beklenildiği gibi, Şekil 5.8c' de görüldüğü gibi kütle akış hızı arttığında bölgenin hava sıcaklığı azalır. Bunlara ek olarak, nem oranının değişimi kütle akış hızının bir fonksiyonu olarak Şekil 5.8d' de verilmiştir. Şekil 5.9a kış çalışma mevsiminde bölge ve duvarların sıcaklık değişimini göstermektedir. Bölgenin yüksek sıcaklık kapasitansından dolayı yavaş cevap vermesinden, bölge değişmez durum sıcaklığına yaklaşık 8000 s sonra ulaşmaktadır. Bunlara ek olarak, yüksek ısıl kapasitanslarından dolayı duvarların değişmez durum değerlerine ulaşması için daha fazla zamana gereksinimleri vardır. Şekil 5.9b bölgenin nem oranı cevabını göstermektedir, açıkça görülmektedir ki nem değişmez durum değerine yaklaşık 2000 s sonra ulaşmaktadır.

Şekil 5.9c' de görüldüğü gibi kütle akış hızı arttığında bölgenin hava sıcaklığı artmaktadır. Şekil 5.9d kütle akış hızındaki değişime bağlı olarak nem oranındaki değişimi göstermektedir. Açıkça sonuçlar göstermektedir ki kütle akış hızı arttığında nem oranındaki değişimin hızı da artmaktadır. Bu cevap karakteristikleri, geri beslemeli kontrol sistemi uygulandığında artacaktır.

5.6.5 Kontrollü sistemin cevap karakteristiği

Birçok endüstriyel uygulamalarda PID (Proportional Integral Derivative), Oransal Integral Türevsel kontrol ediciler yaygın olarak kullanılır, bu birçok ileri düzey kontrol algoritmalarının da temelidir. PID kontrol edicilerin transfer fonksiyonları üç farklı eleman içerir, ki bunlar aşağıdaki gibi tanımlanabilir,



Şekil 5.8 (a) bölge ve duvarların sıcaklığı (Yaz) (b) Bölgenin nem oranı (Yaz)
(c) kütle akış hızının farklı değerlerinde bölgenin sıcaklığı
(d) kütle akış hızının farklı değerlerinde nem oranı

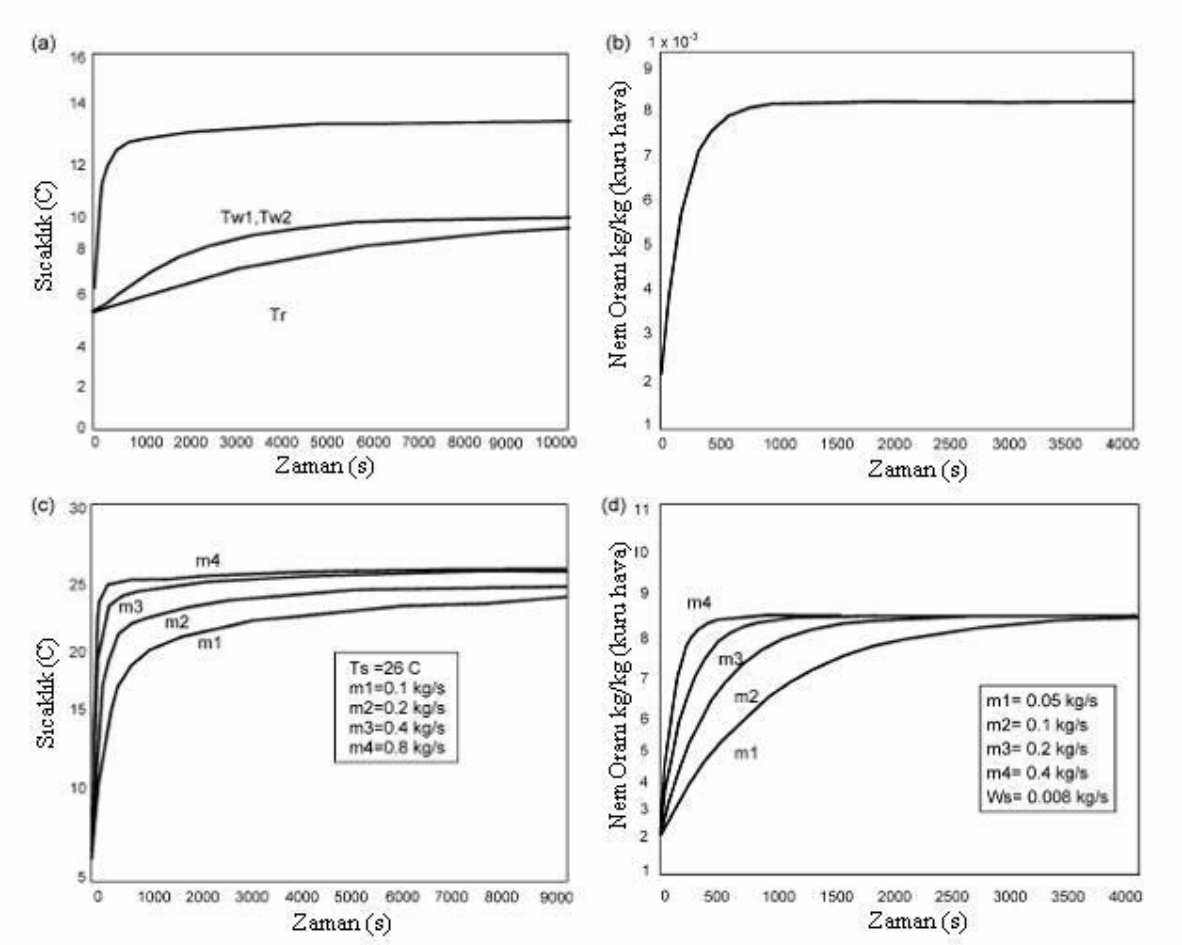
$$G_c(s) = K_P + \frac{K_I}{s} + K_D s \qquad (5.25)$$

burada K_P orantılı kazanç, K_I integral kazanç ve K_D ise türevsel kazançtır.

Böyle bir kontroledici birbirleriyle etkileşimde olan K_P, K_I ve K_D gibi üç farklı ayara sahiptir. Bu nedenle, sistemin tasarım özelliklerine göre en iyi performansı elde edecek şekilde bu üç değeri ayarlamak çok zor ve zaman alıcı olacaktır.

Bu çalışmada, görev HVAC modeli için bölgenin sıcaklık ve nem oranını bir set değerinde tutacak geri beslemeli PID kontrol edicileri geliştirmektir. PID kontrol edicinin parametrelerini bulmak için Ziegler-Nichols metodu kullanıldı. The Ziegler-

Nichols kapalı döngü metodu kontrol döngülerini ayarlamakta yaygın olarak kullanılan birçok metottan biridir. Bu metotla yalnızca oransal kontrol olduğunda, ki osilasyon olacaktır, döngünün kazancı belirlenir, osilasyonların devam ettiği kazançtan ve bu kazançta osilasyonların periyodundan, kontrol edicinin kazancını, integral ve türevsel değerlerini türetir. Bu metot ilk yaklaşım olarak kullanıldı, çünkü iyi bir ayarlama olarak düşünüldü fakat en iyi birisi olarak gerekli değil.

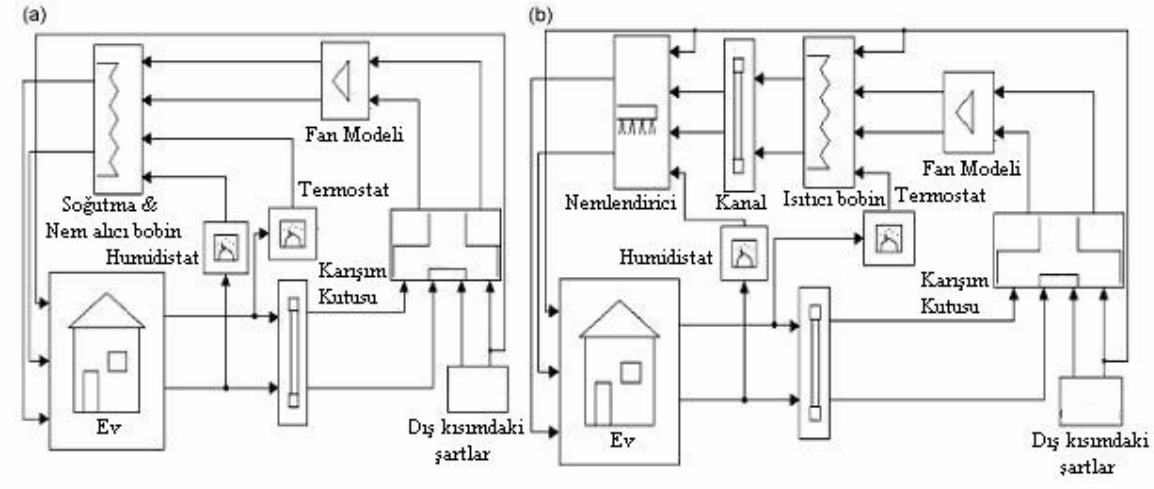


Şekil 5.9 (a) bölge ve duvarların sıcaklığı (Kış) (b) Bölgenin nem oranı (Kış)
(c) kütle akış hızının farklı değerlerinde bölgenin sıcaklığı
(d) kütle akış hızının farklı değerlerinde nem oranı

Şekil 5.10a ve b karşılıklı olarak ısıtma ve soğutma sistemlerinin tamamının şematik diyagramını göstermektedir. Bu sistemler bir bölge, kanal, karıştırma kutusu, fan, termostat, nem ölçücü, ısıtıcı bobin, nemlendirici ve soğutma bobini içermektedir.

Bu diyagramlarda, ısıtma modunda bir ısıtıcı bobin sıcaklığı artırmak için ve nemlendiricide nem oranını artırmak için kullanıldı. Soğutma modunda sıcaklığı ve nem oranını azaltmak için soğutucu – nem alıcı bobin kullanılmıştır. Sistemde kontrol ediciler bu elemanları kontrol etmek ve sıcaklık ve nem oranının set değerlerini izlemek için kullanılmıştır.

Bu analizde, bölgenin sıcaklık ve nem oranı sağlanan havanın sıcaklık ve nem oranının değiştirilmesi ile kontrol edilmiştir. Belirlenen PID katsayıları birinci yaklaşım olarak Ziegler-Nichols metodu ile belirlenmiştir. Daha sonra, deneme yanılma metodu sistemin tasarım özelliklerini karşılayacak parametreleri bulmak için kullanılmıştır. Kontrol ediciler için bu parametre değerleri elde edilmiş ve Çizelge 5.1' de listelenmiştir. Bu tablodan, kazanç değerlerinin küçük olduğu açıktır, diğer bir deyişle yüksek kazanç sistemin cevabını hızlı yapsa bile sistem kararsız olacaktır. Türevsel işlemin değerleri, HVAC alanının bu değerlerden etkilenmediğinden Ziegler-Nichols Metodu ile elde edilen değerlerle aynı tutulmuştur.



Şekil 5.10 (a) Isıtma sisteminin tamamının şematik diyagramı
(b) soğutma sisteminin tamamının şematik diyagramı

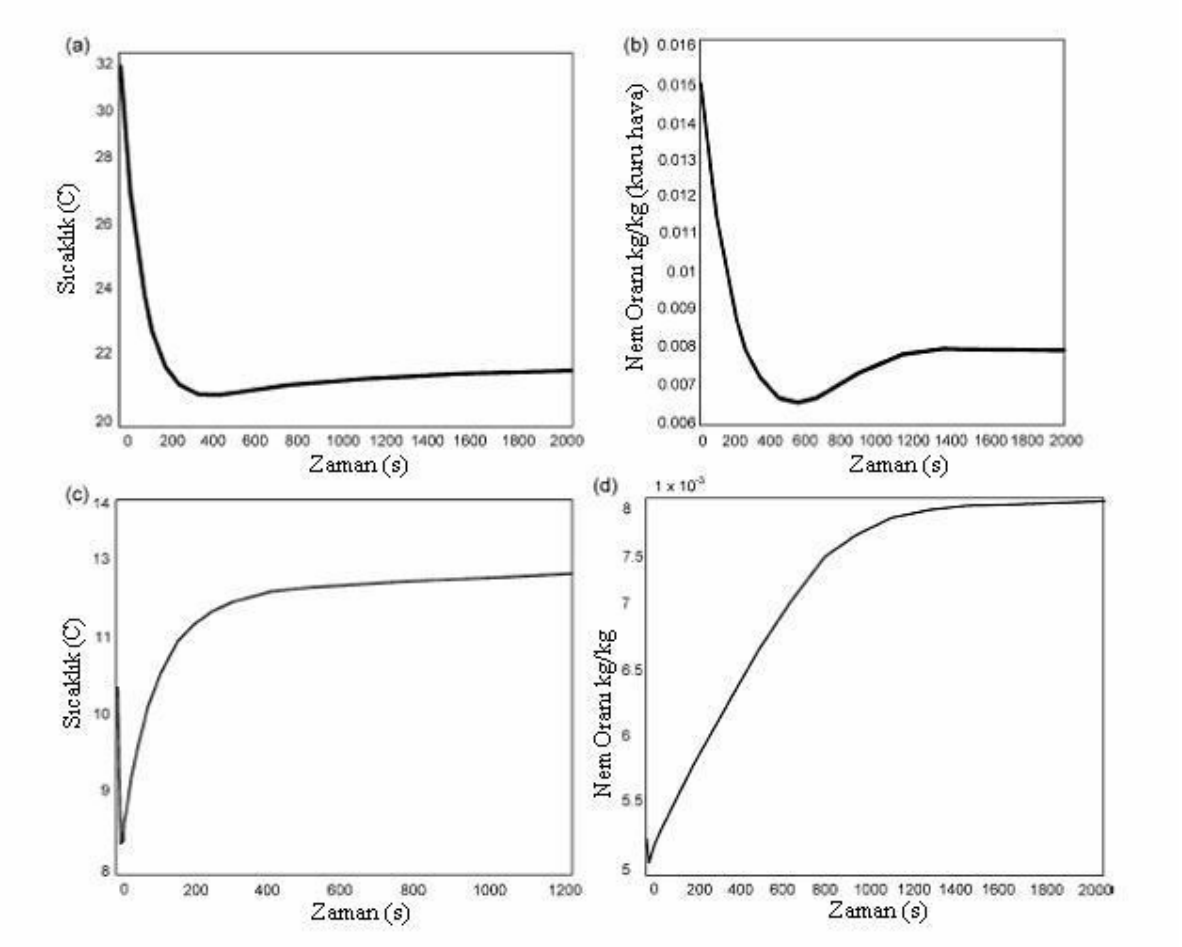
PID kontroledicinin Ziegler-Nichols Metodu ile elde edilen parametre değerleri, çıkışta çok iyi sonuç vermektedir. Örneğin, kış çalışma mevsiminde, sıcaklık yaklaşık 400 s içerisinde 22 ^{o}C' a ulaşmaktadır, fakat bu parametre değeri ısıtıcı bobine 150 ^{o}C' lık bir aşırı yükselmeye sebep olacak çok yüksek bir yük değeri verecektir, bu sonuç onu kabul edilemez yapmaktadır. Bunlara ek olarak, nem oranı için olan 0.008 kg (su)/kg (kuru hava) set değerini karşılamak için, sağlanan hava çok yüksek nem oranına ulaşmalıdır. Bu ise nemlendiricide istenmeyen durum olan yoğuşmaya sebep olacaktır. Bu nedenle, set noktası değerlerine ve ısıtıcı bobin ve nemlendiricide kabul edilebilir yüklere ulaşmak için deneme yanılma ile ince ayar yapılmıştır.

Şekil 5.11a yaz çalışma mevsiminde kontrollü sistem için bölgenin hava sıcaklığını göstermektedir. Sıcaklık set noktası değerine % 2.3' lük bir hata ile ulaşmak için yaklaşık 1200 s yavaşça ve üstel bir şekilde azalmaktadır. Şekil 5.11b yaz çalışma mevsimi için kontrol edilen bölgenin nem oranını göstermektedir. Bu şekilden açıkça görülmektedir ki nem oranı hatasız olarak set noktası değerine 1200 s' de ulaşmak için düzgün bir şekilde azalmaktadır. Şekil 5.11c sağlanan hava sıcaklığını zamanın fonksiyonu olarak göstermektedir.

Çizelge 5.1

PID kontrol edicilerin parametreleri

	Isıtma Modu		Soğutma Modu	
	Nemlendirici	Isıtıcı bobin	Soğutucu bobin	Nem alıcı
Kazanç	0.65	3	2	5
İntegral alıcı	0.008	0.011	0.0015	0.0035
Türev alıcı	10	25	0.1	0.1

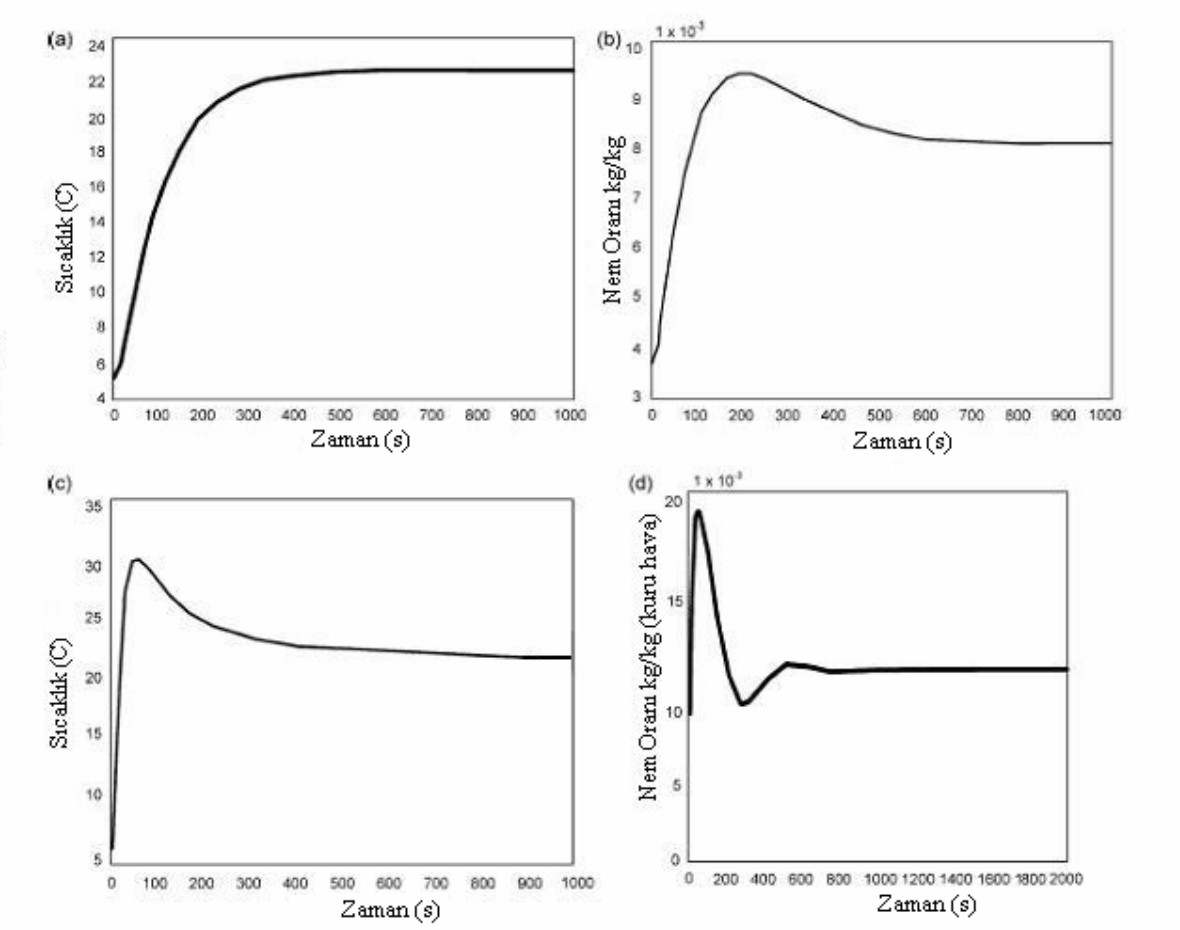


Şekil 5.11 (a) Kontrol edilen bölgenin sıcaklığı (yaz)
(b) Kontrol edilen bölgenin nem oranı(yaz)
(c) Kontrol edilen bölgeye sağlanan havanın sıcaklığı (yaz)
(d) Kontrol edilen bölgeye sağlanan havanın nem oranı (yaz)

Dikkat edilirse sıcaklık başlangıçta 8.5 ^{o}C' a kadar azalmakta ve daha sonra artmakta ve 13 ^{o}C' da kararlı hale gelmektedir. Son olarak Şekil 5.11d düzgün bir şekilde artan ve yaklaşık 1600 s içerisinde kararlı hale gelen sağlanan havanın nem oranını göstermektedir.

Şekil 5.12a kış çalışma mevsiminde kontrollü sistemin bölge hava sıcaklığını göstermektedir. Sıcaklık küçük bir hata ile set noktası değerine ulaşmak için yaklaşık 800 s süre yavaşça ve üstel olarak artmaktadır. Şekil 5.12b kış çalışma mevsimi için kontrollü bölgenin nem oranını göstermektedir. Açıktır ki nem oranı hatasız olarak yaklaşık 800 s sürede set noktası değerine doğru düzgün bir şekilde artmaktadır. Şekil

5.12c kontrollü bölgeye sağlanan havanın sıcaklığını göstermektedir, burada başlangıçta sıcaklık değeri 30 ^{o}C' a kadar artmakta ve daha sonra azalarak yaklaşık 24 ^{o}C' da kararlı olmaktadır. Son olarak Şekil 5.12d kış çalışma mevsimi için kontrollü bölgeye sağlanan havanın nem oranını göstermektedir, ki burada nem oranı hızlı bir şekilde 0.0186 kg (su)/kg (kuru hava) değerine yükselmekte ve daha sonra 0.009 kg/kg (kuru hava) değişmez durum değerine ulaşmaktadır.

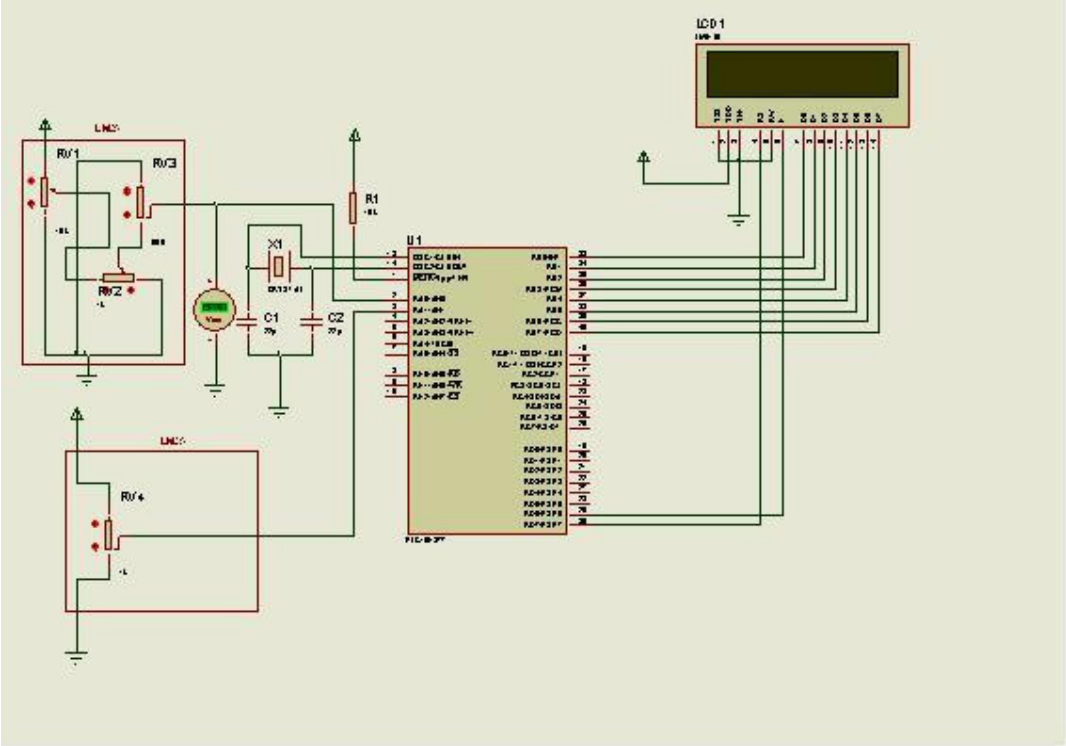


Şekil 5.12 (a) Kontrol edilen bölgenin sıcaklığı (kış)
(b) Kontrol edilen bölgenin nem oranı (kış)
(c) Kontrol edilen bölgeye sağlanan havanın sıcaklığı (kış)
(d) Kontrol edilen bölgeye sağlanan havanın nem oranı (kış)

6. DENEYSEL ÇALIŞMA

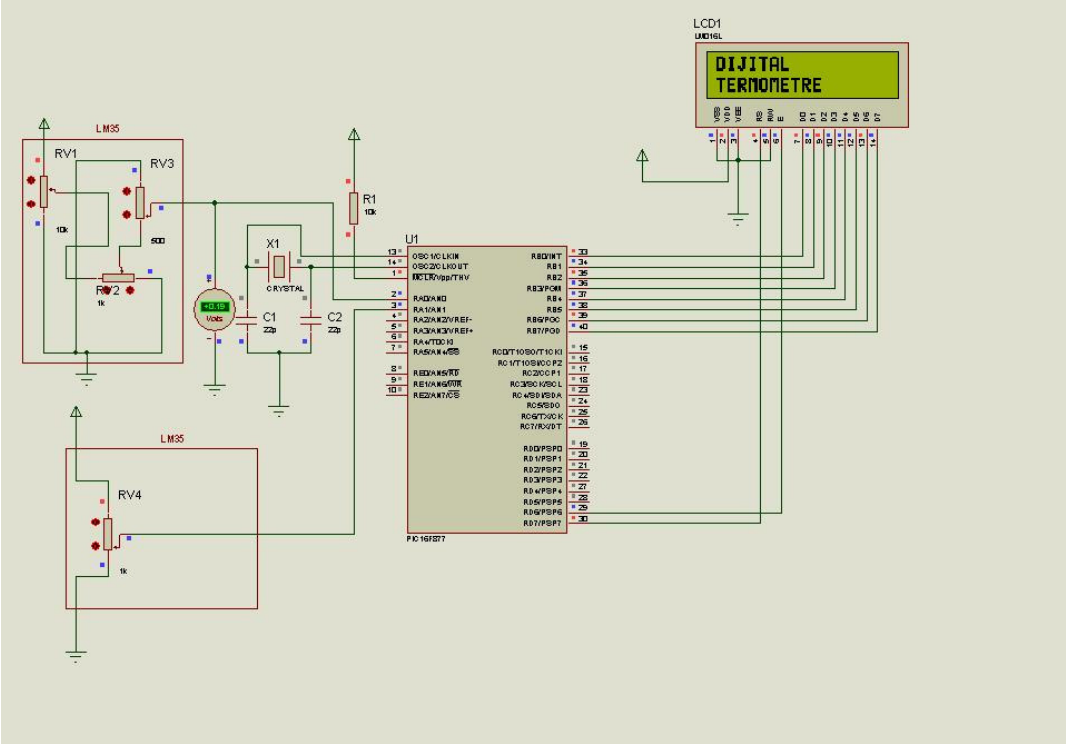
6.1 Dijital Termometre

Tezle ilgili olarak yapılan deneysel çalışmada ilk olarak ortam sıcaklığını ölçen dijital bir termometre yapıldı. Dijital termometre PIC 16F877 mikrodenetleyicisi ve sıcaklık sensörü olarak da LM35 entegresi kullanılarak yapıldı. Bu devrenin ISIS' de çizilmiş devre şeması Şekil 6.1' de görülmektedir.

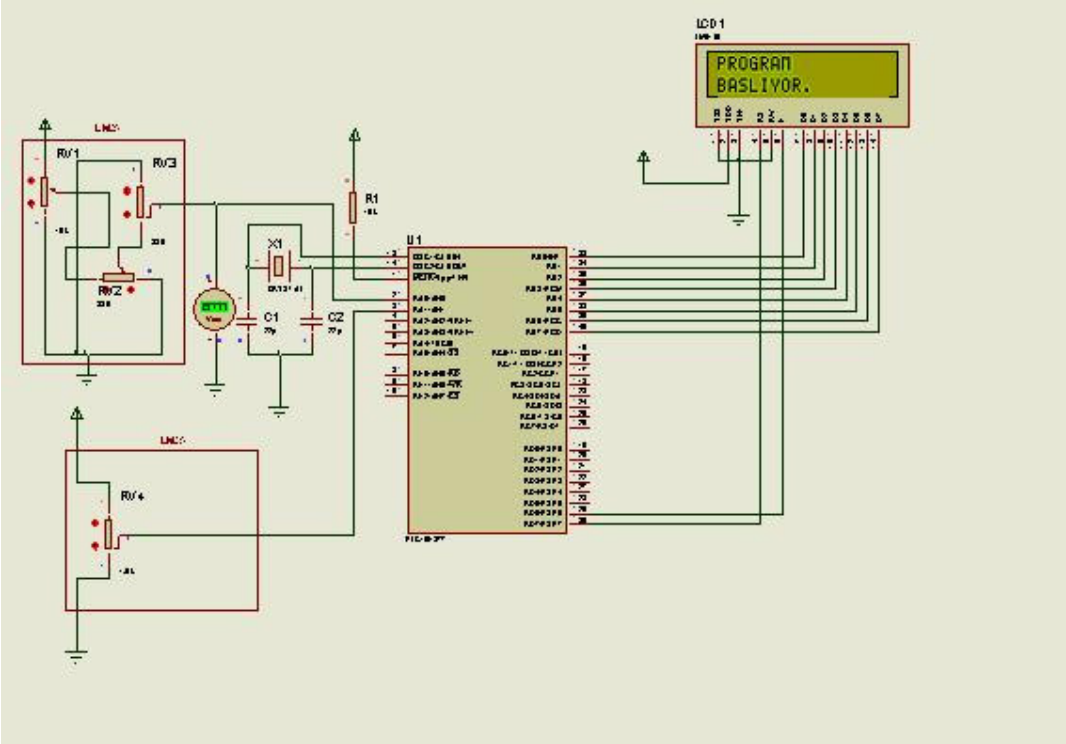


Şekil 6.1 PIC 16F877 ve LM35 sıcaklık sensörü ile oluşturulmuş dijital termometre

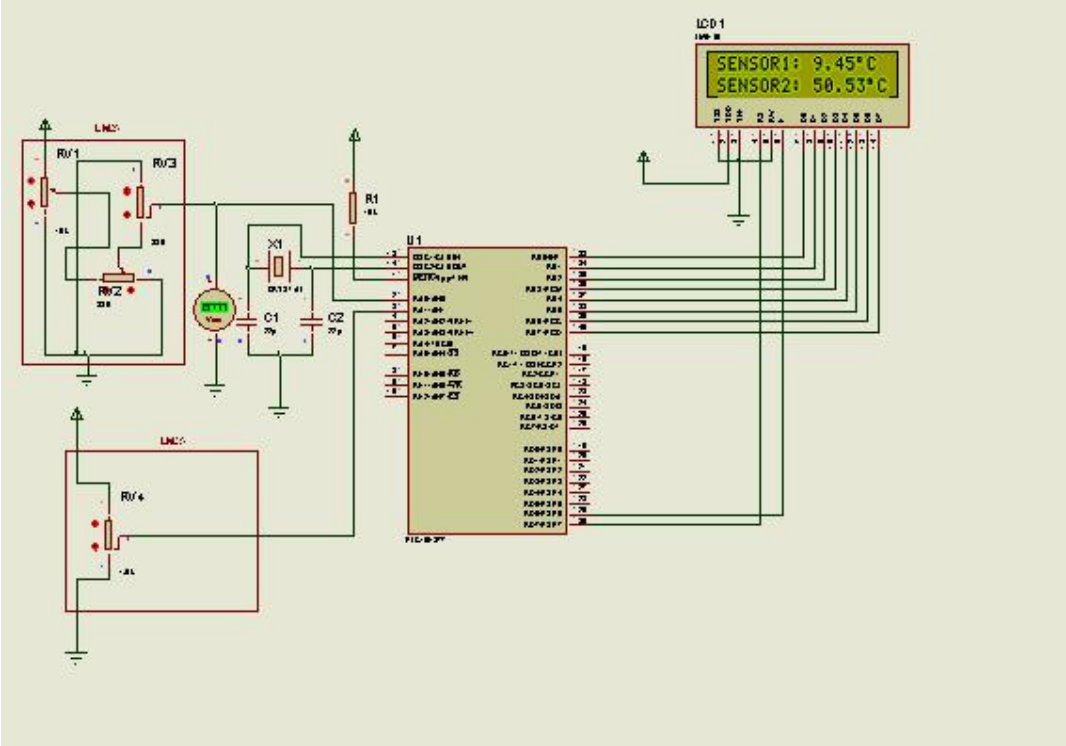
Devrenin ISIS programında simülasyon çalışması yapıldı. Devrenin simülasyonundan görüntüler (Şekil 6.2a, b, c) aşağıdadır.



Şekil 6.2a Dijital termometrenin simülasyon görüntüsü



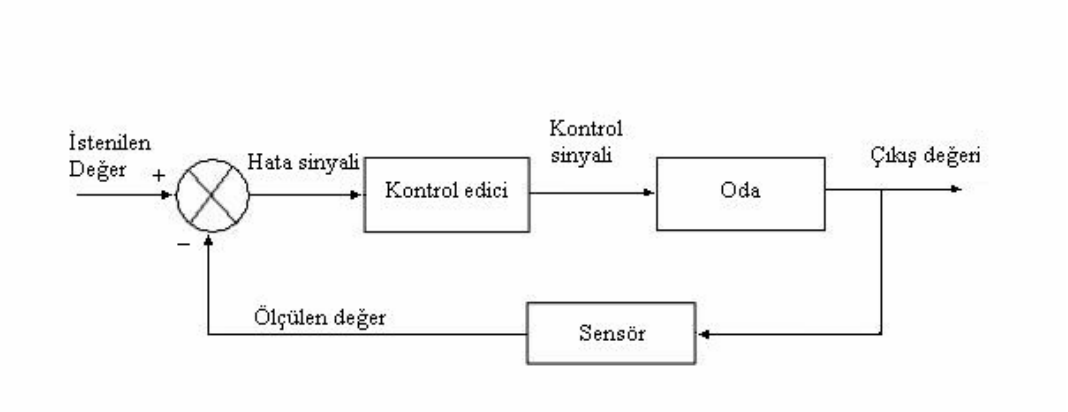
Şekil 6.2b Dijital termometrenin simülasyon görüntüsü



Şekil 6.2c Dijital termometrenin simülasyon görüntüsü

6.2 PIC Mikrodenetleyicisi ile Sıcaklık ve Nem Kontrolü

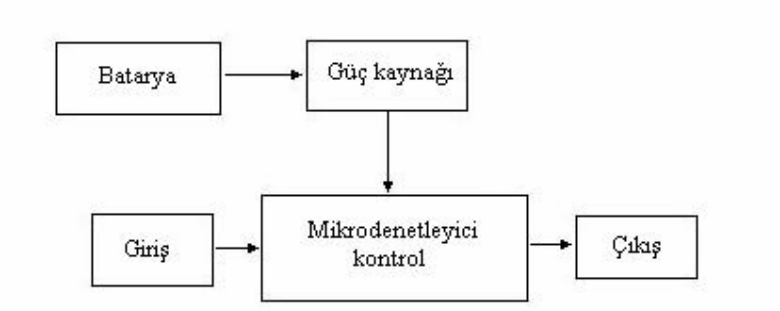
PIC Mikrodenetleyicisi ile kapalı bir odadaki sıcaklık ve nemin kontrol sisteminde, ilk gerekli olan odadaki sıcaklık ve nemdeki değişikleri algılamak veya belirlemektir. İkinci gerekli olan ise, eğer algılanan sıcaklık ve nem istenilen değerlerden farklı ise ısıtıcı ve nemlendiriciyi kontrol etmektir. Bu nedenle odanın çıkış değişkenlerini kontrol etmek için Şekil 6.3'de görüldüğü gibi tasarlanan sistemde sensör ve kontrol edici olmalıdır.



Şekil 6.3 Deneysel devre için kontrol sistemi

Kontrol edici ve oda ileri yöndeki yol üzerinde bulunmakta, sensör ise geri besleme yolu üzerindedir. Odadaki ölçülen sıcaklık ve nem değerleri, toplama noktasında istenilen değerlerle karşılaştırılmaktadır. Fark, yani hata sinyali ise kontrol ediciyi besleyerek, kontrol edicinin odanın çıkış değerlerinin istenilen değere eşit oluncaya kadar, odadaki ısıtıcı ve nemlendirici ünitelerini sürecek kontrol sinyallerini üretmesini sağlar. Böyle bir sistem bazen hata ile harekete geçen sistem olarak adlandırılır.

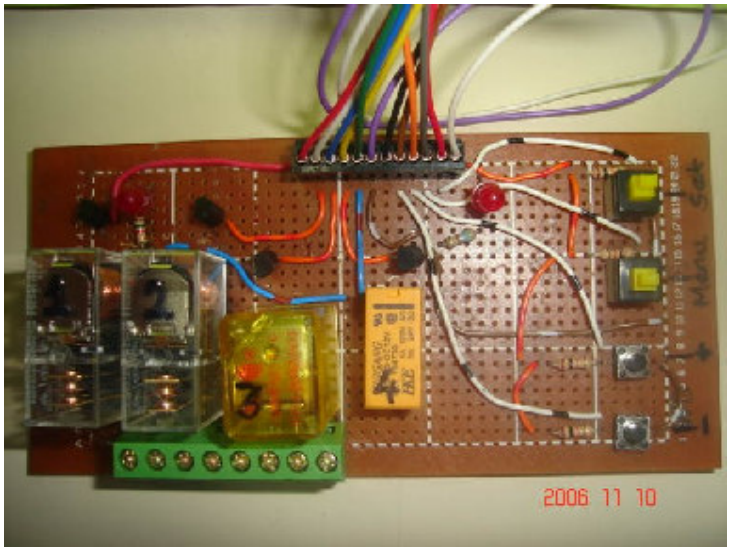
Deneysel devrenin fonksiyonel blok şeması Şekil 6.4' de görülmektedir. Deneysel devrede sistemin elektrik kesilmelerinden etkilenmemesi ve çalışmasına devam edebilmesi için bir batarya gurubu kullanılmıştır. Sistem normal çalışma durumunda şehir şebekesinden beslenmekte ve batarya gurubu şarj olmaktadır, elektrik kesilmesinde ise batarya devreye girerek, sistemin çalışmasına devam etmesini sağlamaktadır.



Şekil 6.4 Deneysel devrenin fonksiyonel blok şeması

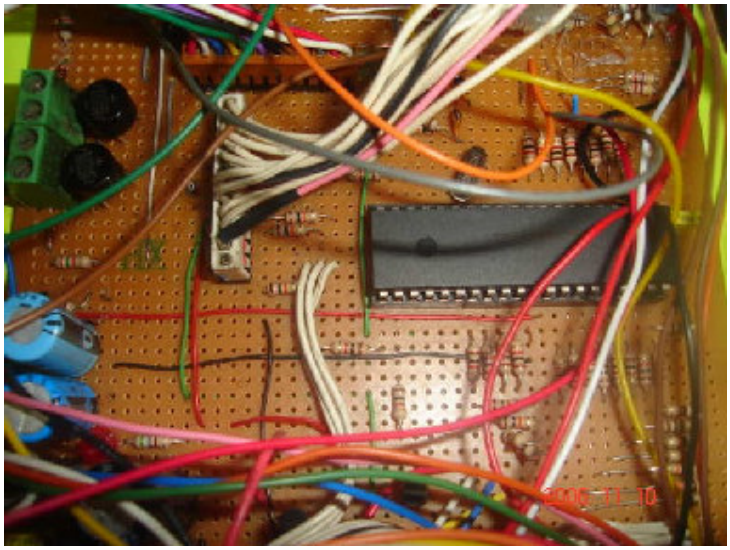
Güç kaynağı devresi, ünitenin çalışması için değişik voltaj değerlerinin elde edilmesini sağlayarak, mikrodenetleyici devresi ve giriş, çıkış devreleri için gerekli olan +5 Volt, +15 Volt gibi voltaj değerlerini verir.

Giriş ve çıkış devresi aynı kart üzerine monte edilerek, giriş devresinde 4 adet buton kullanılmıştır. Bu butonlardan bir tanesi menüyü seçmek için kullanılırken, diğer bir tanesi set butonu olarak kullanıldı, diğer iki butondan bir tanesi set değerlerini ayarlarken artırma butonu, diğeri de azaltma butonu olarak kullanıldı. Çıkış devresinde 4 adet röle kullanılarak; ısıtıcı, nemlendirici, fan ve klimanın kontrolü sağlanmıştır. Röle bobinlerinin enerjilenmesi mikrodenetleyici kontrol devresi tarafından kontrol edilmektedir. Çıkış elemanlarının sürülmesi yolunda röle kontakları seri bağlı olarak bulunduğundan, kontrol devresi ile çıkış devresi arasında elektriksel olarak izolasyon sağlanmış durumdadır. Kontrol devresi ile çıkış devresi arasında elektriksel izolasyonun olması, çıkış devresindeki arızalardan ve büyük değerdeki yükleri sürmek için gerekli olan voltaj değerlerinden kontrol devresinin etkilenmemesini sağlamaktadır. Giriş ve çıkış devresinin fotoğrafı Şekil 6.5' de görülmektedir.



Şekil 6.5 Giriş ve çıkış devresi

PIC 16F877 entegresinin kullanıldığı mikrodenetleyici kontrol devresi sistemin kontrolünü sağlayan devredir. Devrenin baskı devre üzerideki fotoğrafı Şekil 6.6' da görülmektedir. Devre sensörlerden gelen sıcaklık ve nem değeri bilgilerini mikrodenetleyiciye göndererek belirlenen algoritmaya göre kontrol işlemini gerçekleştirmektedir.

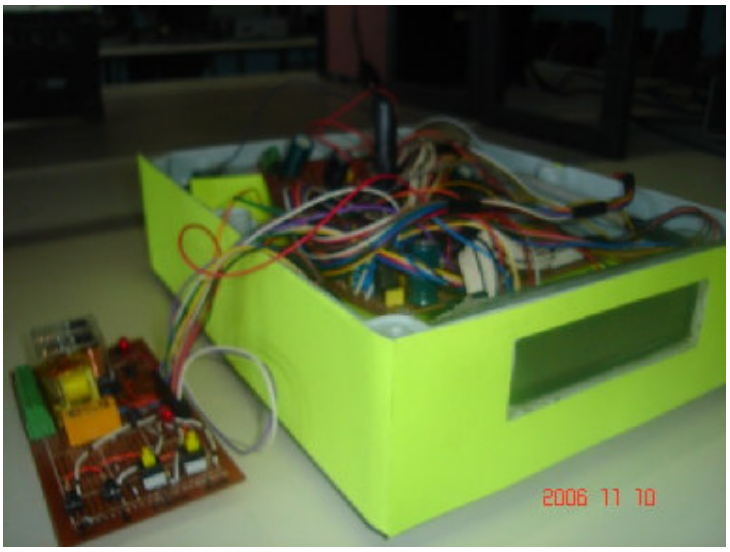


Şekil 6.6 Mikrodenetleyici kontrol devresinin fotoğrafı

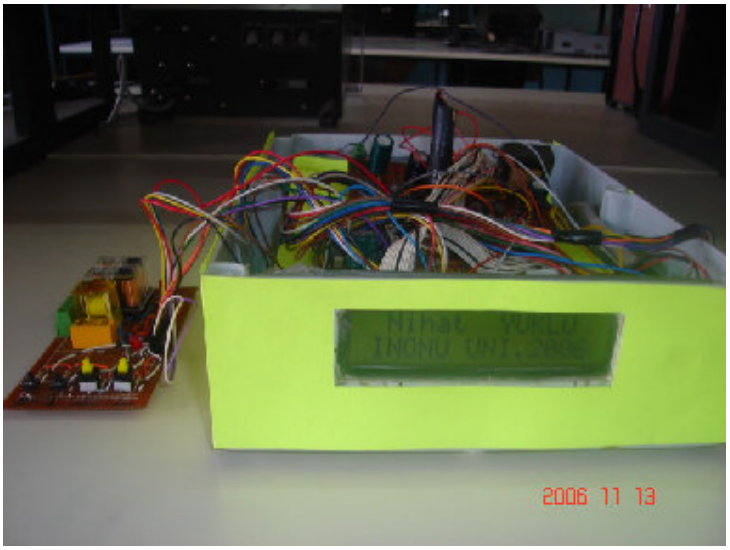
Sensörlerden okunan sıcaklık ve nem değerleri LCD (Liquid Crystal Display) gösterge üzerinde görülebilmektedir. Şekil 6.7' de sistemin çalışır durumda iken elektronik laboratuarının sıcaklık ve nem değerlerini gösterdiği fotoğraf Şekil 6.7' de verilmiştir. Sistemin genel görünüşü ile ilgili görüntüler Şekil 6.8a ve b' de görülmektedir. Sistemin yapıldığı üniversitenin adı ve yıla ait bilgiler Şekil 6.9a ve b' de fotoğraflanmıştır.

Şekil 6.7 Devrede elektronik laboratuarının sıcaklık ve nem değerlerinin okunması

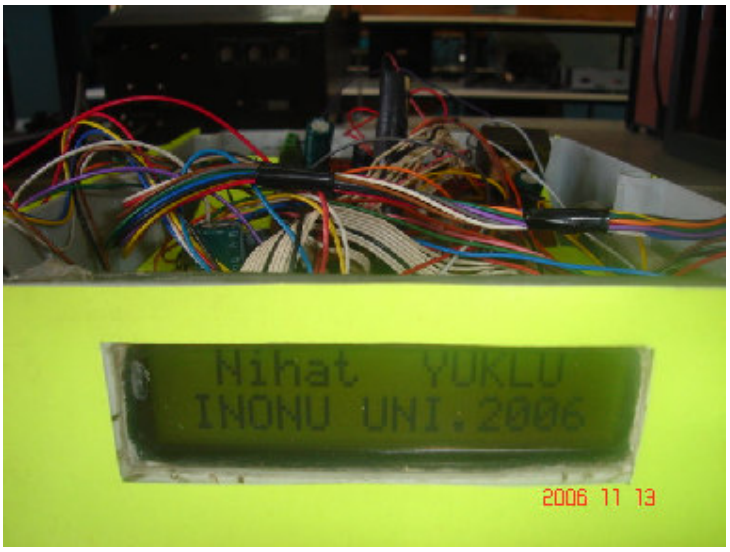
Şekil 6.8a Sistemin genel görünüşü



Şekil 6.8b Sistemin genel görünüşü

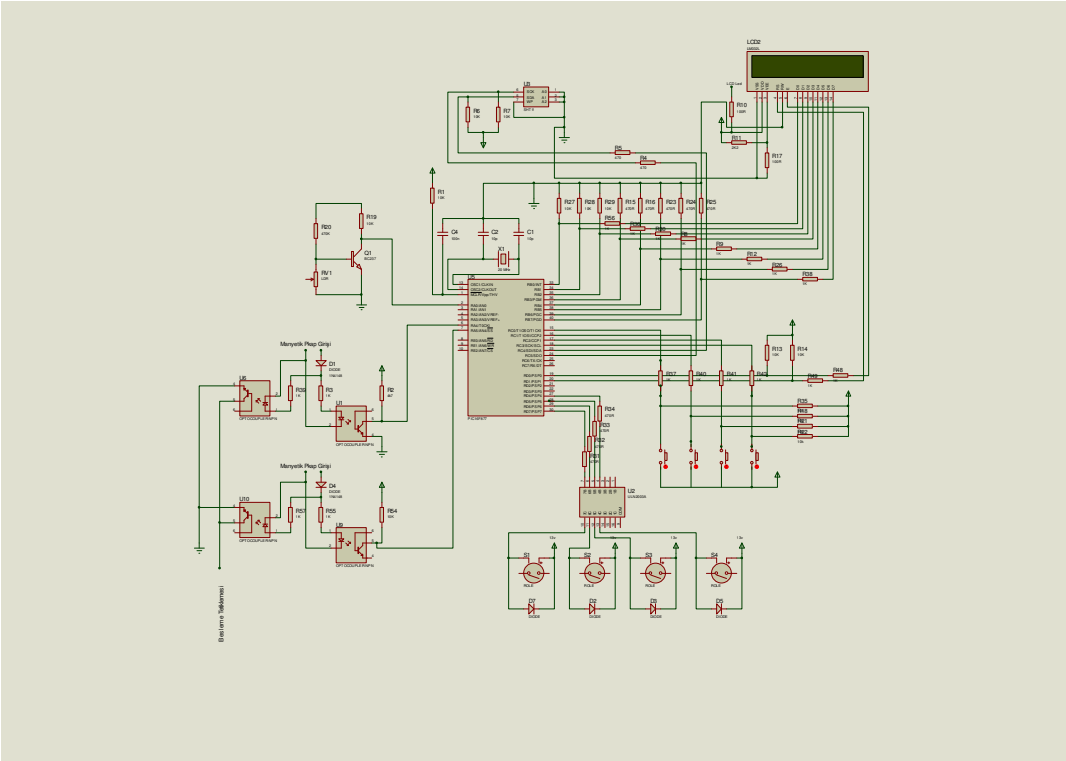


Şekil 6.9a Sistemin yapıldığı üniversitenin adı ve yıla ait bilgiler



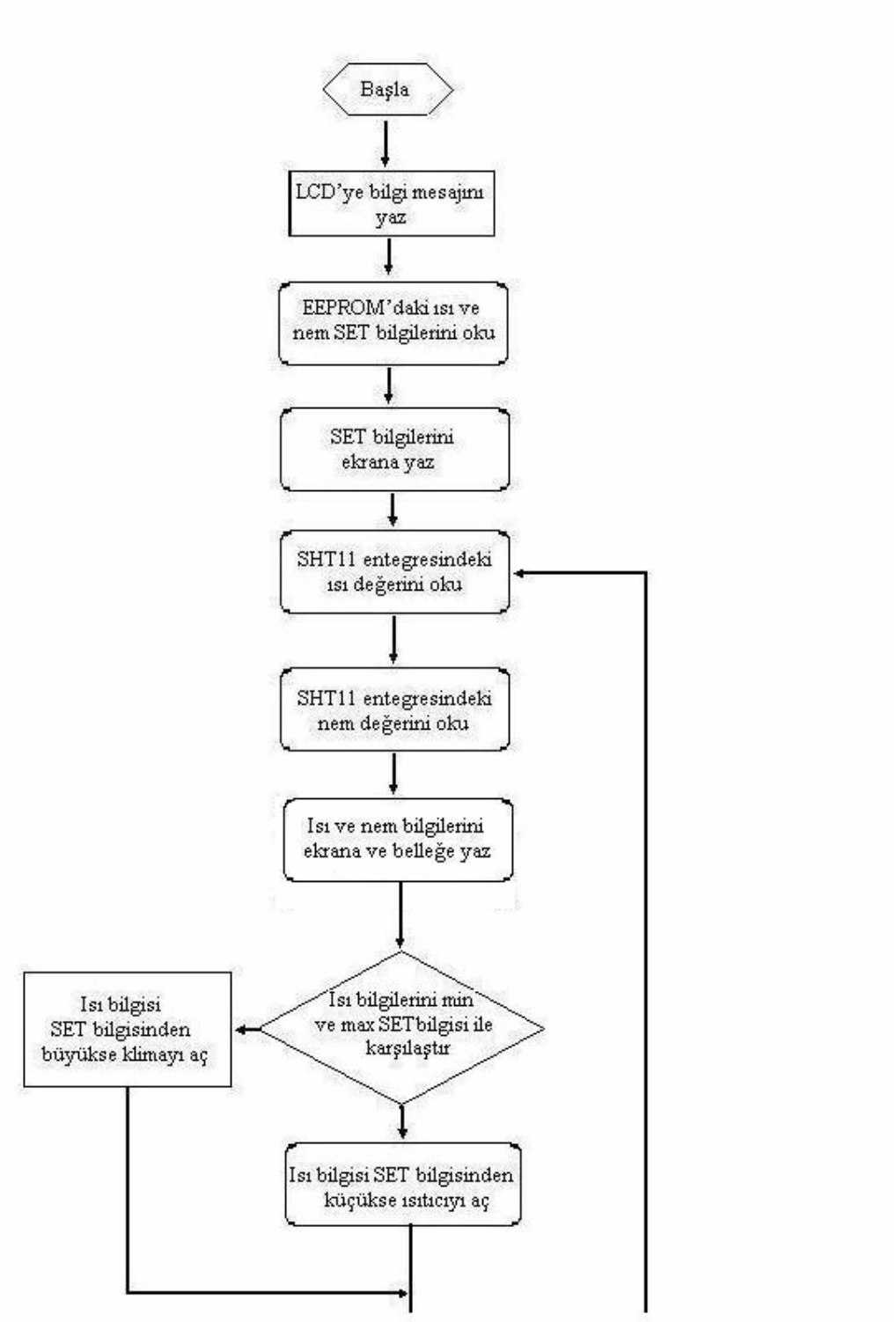
Şekil 6.9b Sistemin yapıldığı üniversitenin adı ve yıla ait bilgiler

PIC Mikrodenetleyicisi ile yapılan sıcaklık ve nem kontrolü ile ilgili devrenin şeması Şekil 6.10' da verilmiştir. Devre ISIS elektronik devre çizim programında çizilmiştir.

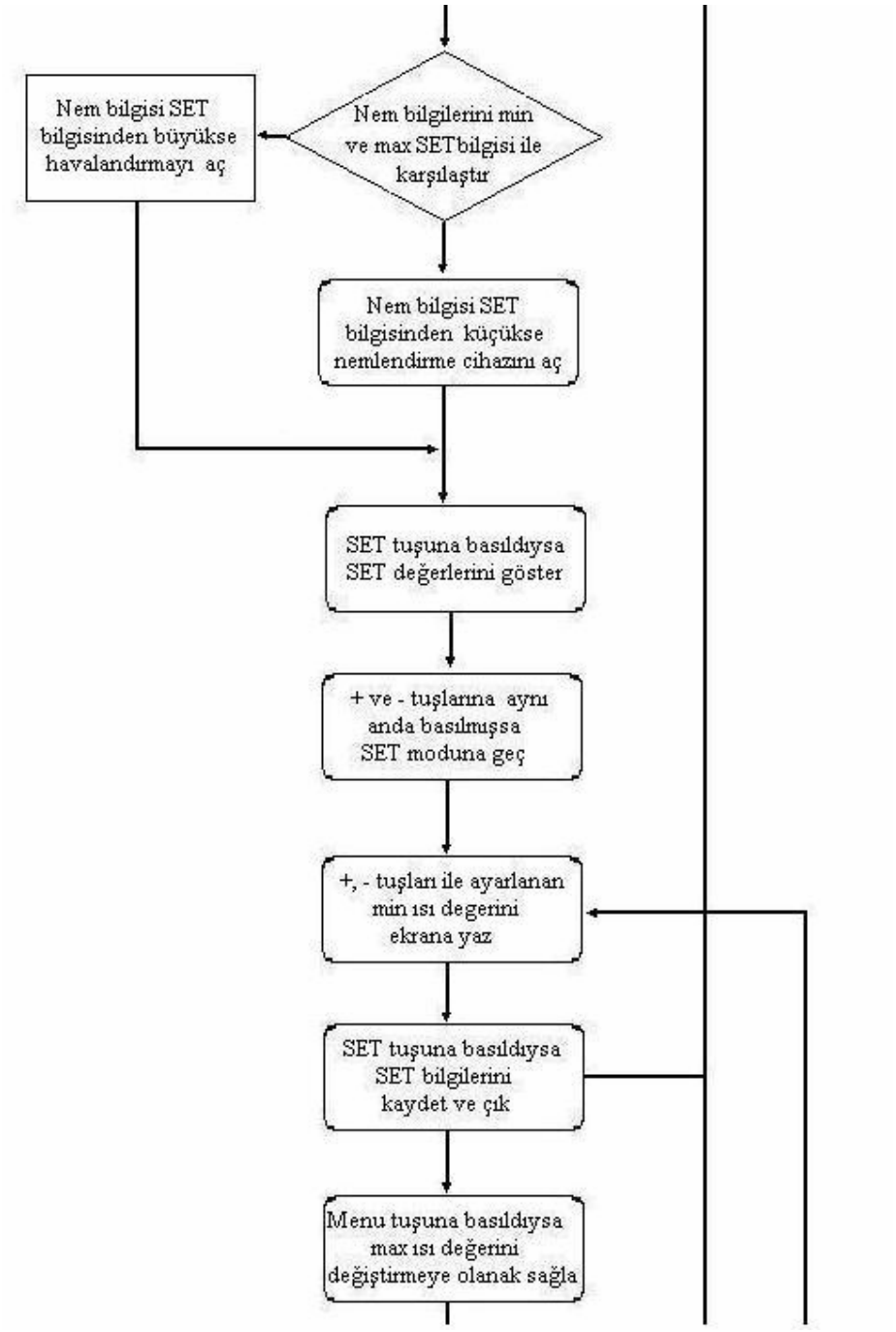


Şekil 6.10 PIC Mikrodenetleyicisi ile yapılan sıcaklık ve nem kontrolü devresi

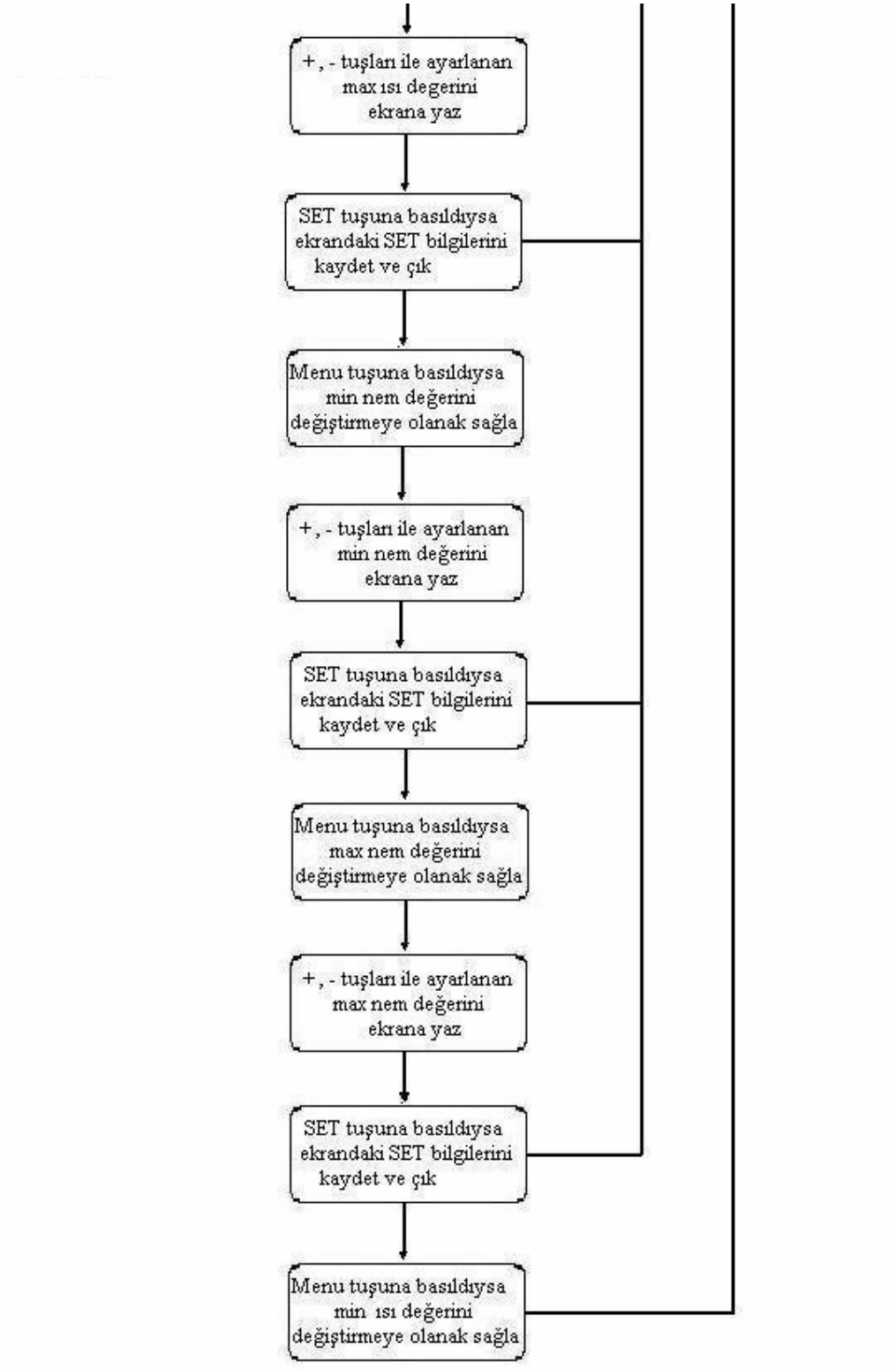
Deneysel çalışmada kontrol işlemi PIC mikrodenetleyicisi içerisine yazılan programla yapıldığından belirlenen algoritmaya bağlı olarak çizilen programın akış diyagramı ise Şekil 6. 11' de verilmiştir. PIC 16F877 entegresine yüklenen program Ek' de verilmiştir. Sistem yazılan programa bağlı olarak deney boardunda butonlarla ayarlanarak LCD göstergede okunan maximum ve minimum sıcaklık değerleri ile maksimum ve minimum nem değerleri arasında kalacak şekilde kapalı ortamın sıcaklık ve nem değerlerini kontrol etmektedir.



Şekil 6.11 Deneysel kontrol işlemi için yazılan programın akış diyagramı



Şekil 6.11 (devamı)



Şekil 6.11 (devamı)

7. SONUÇ

Bu çalışmada kapalı bir ortamda ısıtıcı bobin, soğutucu, nem alma bobini, nemlendirici, kanal ve fan içeren bir HVAC sistemi kontrol etmek için dinamik modelin türetilmesi işlemi incelenerek bileşenlerin matematiksel modeli türetilmiştir. Matematiksel model üzerinde PID kontrol ediciler kullanılmıştır. Sistemin oransal türevsel ve integral kazanç parametreleri Nichols-Ziegler metodu kullanılarak belirlenmiştir. Sonraki aşamada set noktası değerlerini karşılamak için deneme yanılma metodu kullanılmıştır.

Mevcut çalışmalarda analog sinyal olarak ölçülen sıcaklık algılayıcı eleman olan termostat ve nem algılayıcısı humidistattan farklı olarak yapılan tez çalışmasında sıcaklık ve nem değerlerinin ölçümü katı hal yarı iletken teknolojisindeki gelişmeler dikkate alınarak SHT11 entegresi kullanılarak tek bir yongayla ölçülmüştür. Bu yonganın çıkışı dijital olduğu için PIC mikrodenetleyiciye gelen işlem değişkeni bilgileri bir A/D dönüştürücüye gereksinim olmadan doğrudan mikrodenetleyicinin işleyebileceği binary bilgi olarak ulaşmaktadır. Sıcaklık ve nem algılayıcısı eleman olarak katı hal yarı iletken teknolojisi ile üretilen elemanların kullanılması sensörlerin güvenilirliklerini artırdığı gibi uzun süreli çalışma dikkate alındığında sensörlerdeki kararlılığın da optimum düzeyde olmasını sağlamış olacaktır.

SHT11 sensörünün kullanımı sıcaklık ve nem değerlerini elektronik bir sinyal olarak sağlar ki, bu çıkış bize sensör bilgileri üzerinde tez çalışmasında olduğu gibi mantıksal kararlar verebilme yeteneği sağladığı gibi, bu bilgileri dijital ortamda saklayabilme, bilgiler üzerinde matematiksel ve fonksiyonel işlemler yapabilmeyi olanaklı kılarak çok daha karmaşık kontrol sistemlerinin gerçekleştirilmesini sağlar.

Kontrol işleminin PIC mikrodenetleyici kullanılarak gerçekleştirmesi, kontrol işleminde yapılabilecek değişikliği yazılımla gerçekleştirebilme olanağı sunduğu için, bu bize sistem kontrolünde yapılabilecek değişikliğin donanımla gerçekleştirildiği sistemlere göre ekonomik avantaj sağladığı gibi kontrol elastikiyeti ve sistemlerin bakım, onarım ve işletim giderleri açısından önemli faydalar sağlayarak, kontrol edilen sistemin ticari olarak da rekabet edebilir olmasını olanaklı kılar.

8. KAYNAKLAR

1. B. Tashtoush , M. Molhim, M. Al-Rousan, *Dynamic model of an HVAC system for control analysis*. Energy V.30, 2005, pp. 1729–1745

2. D.R. Clark, C.W. Hurley, C.R. Hill, *Dynamic models for HVAC system components*. ASHRAE Trans. 1985, V.91(1), pp.737-751

3. P. Riederera, D. Marchiob, J.C. Visiera, A. Husaunndeea, R. Lahrecha, *Room thermal modelling adapted to the test of HVAC control systems.* Building and Environment V.37, 2002, pp. 777 – 790

4. X. Peng, A. Passen, *State space model for predicting and controlling the temperature response of indoor air zones*. Energy Building Trans. 1998, V.28, pp. 197-203

5. Kasahara M, Kuzu E, Matsuba T, Hashimoto Y, Kamimura K, Kurosu S. *Physical model of air conditioned space for control analysis*, ASHRAE Trans, 2000, V.106 (2), pp. 307-317.

6. M. A. Morteza, Orteza, F.S. Theodore, *Evaluation of HVAC system operational strategies for commercial buildings.* Energy Convers. V.38, N.3, 1997, pp. 225-236.

7. Lu Lu, Wenjian Cai, Yeng Soh Chai, Lihua Xie, *Global optimization for overall HVAC systems—Part I problem formulation and analysis*, Energy Conversion and Management, V.46, 2005, pp. 999–1014.

8. Lu Lu, Wenjian Cai∗, Lihua Xie, Shujiang Li, Yeng Chai Soh, *HVAC system optimization—in-building section,* Energy and Buildings V.37, 2005, pp. 11- 22.

9. K.G. Arvanitis, P.N. Paraskevopoulos, A.A. Vernardos, *Multirate adaptive temperature control of greenhouses,* Computers and Electronics in Agriculture, V.26, 2000 pp. 303–320.

10. Tim Salsbury, Rick Diamond, *Performance validation and energy analysis of HVAC systems using simulation*, Energy and Buildings, V.32, 2000, pp.5–17.

11. Sigurd Skogestad, *Simple analytic rules for model reduction and PID controller tuning*, Journal of Process Control, V.13, 2003, pp. 291–309.

12. P. Ole Fanger, *Indoor Air Quality in the 21st Century: Search for Excellence*, Indoor Air,
V.10, 2000, pp. 68–73.

13. Orhan Altınbaşak, *Mikrodenetleyiciler ve PIC Programlama,* Altaş Yayıncılık ve Elektronik Tic. Ltd. Şti, İstanbul, 2004, p. 12-34.

14. Anonymous, *PIC16F84 Datasheet,* www.microchip.com, Microchip Technology Incorporated, U.S.A, 2002, p. 1- 54

15. Doğan İbrahim, *PIC Mikrokontrolör Öğreniyorum,* Bileşim Yayınları, İstanbul, 2005, p.18 – 32.

16. John Iovine, *PIC Microcontroller Project Book*, McGraw – Hill, U.S.A, 2000, p. 1 – 64.

17. Nebosja Matic, *The PIC Microcontroller*, http://www.mikroelektronika.co.yu, 2003

18. Jon S. Wilson, *Sensor Technology Handbook*, Elsevier Inc, UK, 2005, p. 531-561

19. Matthias Nau, *Electrical Temperature Measurement with thermocouples and Resistance Termometers*, M. K. Juchheim, Fulda, 2002, p. 7 – 62.

20. Anonymous, *LM35 Precision Centigrade Temperature Sensors Datasheet*, www.national.com, National Semiconductor Corporation, 2000.

21. Anonymous, SHT11 Humidity and Temperature Sensor Datasheet, www.sensirion.com, Sensirion The Sensor Company, Switzerland, 2005.

22. Karl J. Aström, Björn Wittenmark, *Computer – Controlled Systems Theory and Design,* Tsinghua University Pres, Prentice Hall, 1997, p. 1- 12

23. A.C.W.Wong, A.T.P.So, *Building Automation In The 2lst Century*, Proceedings of the 4th International Conference on Advances in Power System Control, Operation and Management, APSCOM-97, Hong Kong, November 1997

EK - PIC ENTEGRESİNE YÜKLENEN PROGRAM

Deneysel çalışmada kontrol işlemi PIC mikrodenetleyicisi içerisine yazılan programla gerçekleştirildiğinden, PIC 16F877 entegresine yüklenen program aşağıda verilmiştir.

```
..... ////////////////////////////////////////////////////////////////////////
..... //Bu programdaki amac SHT11 ile ısı ve nem olcumu yapmak içindir devrede  //
..... //PIC16f877 ye program yazılmıştır.Bilgiler LCD üzerinden gösterilir.     //
..... //
..... ////////////////////////////////////////////////////////////////////////
.....
..... #include <16F877.h>
.....  //////// Standard Header file for the PIC16F877 device ////////////////
..... #device PIC16F877
..... #list
.....
..... #fuses HS,NOWDT,NOPROTECT,NOLVP
..... #use delay(clock=20000000)
*
0011:  MOVLW  44
0012:  MOVWF  04
0013:  MOVF   00,W
0014:  BTFSC  03.2
0015:  GOTO   023
0016:  MOVLW  06
0017:  MOVWF  78
0018:  CLRF   77
0019:  DECFSZ 77,F
001A:  GOTO   019
001B:  DECFSZ 78,F
001C:  GOTO   018
001D:  MOVLW  7B
001E:  MOVWF  77
001F:  DECFSZ 77,F
0020:  GOTO   01F
0021:  DECFSZ 00,F
0022:  GOTO   016
0023:  RETLW  00
..... #byte port_a =5                 // portlar tanıtılır
..... #byte port_b =6
..... #byte port_c =7
..... #byte port_d =8
..... #byte port_e =9
..... #define all_out 0
..... #define all_in 0xff
.....
..... #include <2432.c>                // eeprom entegresinin programı
.....  ////////////////////////////////////////////////////////////////////////
..... ////   Library for a MicroChip 24C32                                ////
..... ////                                                               ////
..... ////   init_ext_eeprom();    Call before the other functions are used  ////
..... ////                                                               ////
..... ////   write_ext_eeprom(a, d);  Write the byte d to the address a      ////
..... ////                                                               ////
..... ////   d = read_ext_eeprom(a);  Read the byte d from the address a     ////
..... ////                                                               ////
..... ////   b = ext_eeprom_ready();  Returns TRUE if the eeprom is ready    ////
..... ////                           to receive opcodes                    ////
..... ////                                                               ////
..... ////   The main program may define EEPROM_SDA                       ////
..... ////   and EEPROM_SCL to override the defaults below.               ////
..... ////                                                               ////
..... ////                           Pin Layout                          ////
..... ////   ------------------------------------------------------------    ////
..... ////   |                                                      |    ////
```

```
..... ////   | 1: NC   Not Connected | 8: VCC   +5V                  |    ////
..... ////   |                  |                          |    ////
..... ////   | 2: NC   Not Connected | 7: WP    GND                  |    ////
..... ////   |                  |                          |    ////
..... ////   | 3: NC   Not Connected | 6: SCL   EEPROM_SCL and Pull-Up |    ////
..... ////   |                  |                          |    ////
..... ////   | 4: VSS  GND          | 5: SDA   EEPROM_SDA and Pull-Up |    ////
..... ////   ------------------------------------------------------------    ////
..... ////                                                                   ////
..... ////////////////////////////////////////////////////////////////////////////
..... ////        (C) Copyright 1996, 2003 Custom Computer Services          ////
..... //// This source code may only be used by licensed users of the CCS C  ////
..... //// compiler.  This source code may only be distributed to other      ////
..... //// licensed users of the CCS C compiler.  No other use, reproduction ////
..... //// or distribution is permitted without written permission.          ////
..... //// Derivative programs created using this software in object code    ////
..... //// form are not restricted in any way.                              ////
..... ////////////////////////////////////////////////////////////////////////////
..... #ifndef EEPROM_SDA
.....
..... #define EEPROM_SDA  PIN_C4
..... #define EEPROM_SCL  PIN_C3
.....
..... #endif
.....
.....
..... #define hi(x)  (*(&x+1))
.....
..... #use i2c(master, sda=EEPROM_SDA, scl=EEPROM_SCL)
*
00C1:  MOVLW  08
00C2:  MOVWF  78
00C3:  NOP
00C4:  MOVLW  02
00C5:  MOVWF  77
00C6:  DECFSZ 77,F
00C7:  GOTO   0C6
00C8:  BCF    07.3
00C9:  BCF    20.3
00CA:  MOVF   20,W
00CB:  BSF    03.5
00CC:  MOVWF  07
00CD:  MOVLW  03
00CE:  MOVWF  77
00CF:  DECFSZ 77,F
00D0:  GOTO   0CF
00D1:  BCF    03.5
00D2:  RLF    47,F
00D3:  BCF    07.4
00D4:  BTFSS  03.0
00D5:  GOTO   0DC
00D6:  BSF    20.4
00D7:  MOVF   20,W
00D8:  BSF    03.5
00D9:  MOVWF  07
00DA:  GOTO   0E0
00DB:  BCF    03.5
00DC:  BCF    20.4
00DD:  MOVF   20,W
00DE:  BSF    03.5
00DF:  MOVWF  07
00E0:  NOP
00E1:  BCF    03.5
00E2:  BSF    20.3
00E3:  MOVF   20,W
00E4:  BSF    03.5
00E5:  MOVWF  07
```

```
00E6:  BCF   03.5
00E7:  BTFSC  07.3
00E8:  GOTO  0EB
00E9:  BSF   03.5
00EA:  GOTO  0E6
00EB:  DECFSZ 78,F
00EC:  GOTO  0C3
00ED:  MOVLW  02
00EE:  MOVWF  77
00EF:  DECFSZ 77,F
00F0:  GOTO  0EF
00F1:  BCF   07.3
00F2:  BCF   20.3
00F3:  MOVF  20,W
00F4:  BSF   03.5
00F5:  MOVWF  07
00F6:  NOP
00F7:  BCF   03.5
00F8:  BSF   20.4
00F9:  MOVF  20,W
00FA:  BSF   03.5
00FB:  MOVWF  07
00FC:  MOVLW  03
00FD:  MOVWF  77
00FE:  DECFSZ 77,F
00FF:  GOTO  0FE
0100:  MOVLW  03
0101:  MOVWF  77
0102:  DECFSZ 77,F
0103:  GOTO  102
0104:  BCF   03.5
0105:  BSF   20.3
0106:  MOVF  20,W
0107:  BSF   03.5
0108:  MOVWF  07
0109:  BCF   03.5
010A:  BTFSC  07.3
010B:  GOTO  10E
010C:  BSF   03.5
010D:  GOTO  109
010E:  CLRF  78
010F:  MOVLW  03
0110:  MOVWF  77
0111:  DECFSZ 77,F
0112:  GOTO  111
0113:  BTFSC  07.4
0114:  BSF   78.0
0115:  BCF   07.3
0116:  BCF   20.3
0117:  MOVF  20,W
0118:  BSF   03.5
0119:  MOVWF  07
011A:  BCF   03.5
011B:  BCF   07.4
011C:  BCF   20.4
011D:  MOVF  20,W
011E:  BSF   03.5
011F:  MOVWF  07
0120:  BCF   03.5
0121:  RETLW  00
0122:  MOVLW  08
0123:  MOVWF  47
0124:  MOVF  77,W
0125:  MOVWF  48
0126:  BSF   20.4
0127:  MOVF  20,W
0128:  BSF   03.5
```

```
0129:  MOVWF  07
012A:  MOVLW  03
012B:  MOVWF  77
012C:  DECFSZ 77,F
012D:  GOTO   12C
012E:  BCF    03.5
012F:  BSF    20.3
0130:  MOVF   20,W
0131:  BSF    03.5
0132:  MOVWF  07
0133:  BCF    03.5
0134:  BTFSC  07.3
0135:  GOTO   138
0136:  BSF    03.5
0137:  GOTO   133
0138:  BTFSC  07.4
0139:  BSF    03.0
013A:  BTFSS  07.4
013B:  BCF    03.0
013C:  RLF    78,F
013D:  MOVLW  02
013E:  MOVWF  77
013F:  DECFSZ 77,F
0140:  GOTO   13F
0141:  BCF    20.3
0142:  MOVF   20,W
0143:  BSF    03.5
0144:  MOVWF  07
0145:  BCF    03.5
0146:  BCF    07.3
0147:  DECFSZ 47,F
0148:  GOTO   126
0149:  BSF    20.4
014A:  MOVF   20,W
014B:  BSF    03.5
014C:  MOVWF  07
014D:  MOVLW  03
014E:  MOVWF  77
014F:  DECFSZ 77,F
0150:  GOTO   14F
0151:  BCF    03.5
0152:  BCF    07.4
0153:  MOVF   48,W
0154:  BTFSC  03.2
0155:  GOTO   15B
0156:  BCF    20.4
0157:  MOVF   20,W
0158:  BSF    03.5
0159:  MOVWF  07
015A:  BCF    03.5
015B:  NOP
015C:  BSF    20.3
015D:  MOVF   20,W
015E:  BSF    03.5
015F:  MOVWF  07
0160:  BCF    03.5
0161:  BTFSC  07.3
0162:  GOTO   165
0163:  BSF    03.5
0164:  GOTO   160
0165:  MOVLW  02
0166:  MOVWF  77
0167:  DECFSZ 77,F
0168:  GOTO   167
0169:  BCF    07.3
016A:  BCF    20.3
016B:  MOVF   20,W
```

```
016C:  BSF    03.5
016D:  MOVWF  07
016E:  MOVLW  03
016F:  MOVWF  77
0170:  DECFSZ 77,F
0171:  GOTO   170
0172:  BCF    03.5
0173:  BCF    07.4
0174:  BCF    20.4
0175:  MOVF   20,W
0176:  BSF    03.5
0177:  MOVWF  07
0178:  BCF    03.5
0179:  BCF    0A.3
017A:  BCF    0A.4
017B:  GOTO   226 (RETURN)
*
029C:  MOVLW  FF
029D:  BCF    03.5
029E:  MOVWF  20
.....................
..................... #define EEPROM_ADDRESS long int
..................... #define EEPROM_SIZE    4096
.....................
..................... void init_ext_eeprom() {
.....................    output_float(EEPROM_SCL);
.....................    output_float(EEPROM_SDA);
..................... }
.....................
..................... BOOLEAN ext_eeprom_ready() {
.....................    int1 ack;
.....................    i2c_start();            // If the write command is acknowledged,
*
017C:  BSF    20.4
017D:  MOVF   20,W
017E:  BSF    03.5
017F:  MOVWF  07
0180:  MOVLW  02
0181:  MOVWF  77
0182:  DECFSZ 77,F
0183:  GOTO   182
0184:  BCF    03.5
0185:  BSF    20.3
0186:  MOVF   20,W
0187:  BSF    03.5
0188:  MOVWF  07
0189:  MOVLW  03
018A:  MOVWF  77
018B:  DECFSZ 77,F
018C:  GOTO   18B
018D:  BCF    03.5
018E:  BCF    07.4
018F:  BCF    20.4
0190:  MOVF   20,W
0191:  BSF    03.5
0192:  MOVWF  07
0193:  MOVLW  02
0194:  MOVWF  77
0195:  DECFSZ 77,F
0196:  GOTO   195
0197:  BCF    03.5
0198:  BCF    07.3
0199:  BCF    20.3
019A:  MOVF   20,W
019B:  BSF    03.5
019C:  MOVWF  07
.....................    ack = i2c_write(0xa0);  // then the device is ready.
```

```
019D:  MOVLW  A0
019E:  BCF    03.5
019F:  MOVWF  47
01A0:  CALL   0C1
01A1:  MOVF   78,W
01A2:  BTFSC  78.0
01A3:  GOTO   1A6
01A4:  BCF    46.0
01A5:  GOTO   1A7
01A6:  BSF    46.0
.....    i2c_stop();
01A7:  BCF    20.4
01A8:  MOVF   20,W
01A9:  BSF    03.5
01AA:  MOVWF  07
01AB:  NOP
01AC:  BCF    03.5
01AD:  BSF    20.3
01AE:  MOVF   20,W
01AF:  BSF    03.5
01B0:  MOVWF  07
01B1:  BCF    03.5
01B2:  BTFSC  07.3
01B3:  GOTO   1B6
01B4:  BSF    03.5
01B5:  GOTO   1B1
01B6:  MOVLW  02
01B7:  MOVWF  77
01B8:  DECFSZ 77,F
01B9:  GOTO   1B8
01BA:  NOP
01BB:  NOP
01BC:  NOP
01BD:  BSF    20.4
01BE:  MOVF   20,W
01BF:  BSF    03.5
01C0:  MOVWF  07
01C1:  MOVLW  02
01C2:  MOVWF  77
01C3:  DECFSZ 77,F
01C4:  GOTO   1C3
.....    return !ack;
01C5:  MOVLW  00
01C6:  BCF    03.5
01C7:  BTFSS  46.0
01C8:  MOVLW  01
01C9:  MOVWF  78
..... }
.....
..... void write_ext_eeprom(long int address, BYTE data) {
.....    while(!ext_eeprom_ready());
.....    i2c_start();
.....    i2c_write(0xa0);
.....    i2c_write(hi(address));
.....    i2c_write(address);
.....    i2c_write(data);
.....    i2c_stop();
..... }
.....
.....
..... BYTE read_ext_eeprom(long int address) {
.....    BYTE data;
.....
.....    while(!ext_eeprom_ready());
01CA:  MOVF   78,F
01CB:  BTFSC  03.2
01CC:  GOTO   17C
```

```
.....................  i2c_start();
01CD:  BSF    20.4
01CE:  MOVF   20,W
01CF:  BSF    03.5
01D0:  MOVWF  07
01D1:  MOVLW  02
01D2:  MOVWF  77
01D3:  DECFSZ 77,F
01D4:  GOTO   1D3
01D5:  BCF    03.5
01D6:  BSF    20.3
01D7:  MOVF   20,W
01D8:  BSF    03.5
01D9:  MOVWF  07
01DA:  MOVLW  03
01DB:  MOVWF  77
01DC:  DECFSZ 77,F
01DD:  GOTO   1DC
01DE:  BCF    03.5
01DF:  BCF    07.4
01E0:  BCF    20.4
01E1:  MOVF   20,W
01E2:  BSF    03.5
01E3:  MOVWF  07
01E4:  MOVLW  02
01E5:  MOVWF  77
01E6:  DECFSZ 77,F
01E7:  GOTO   1E6
01E8:  BCF    03.5
01E9:  BCF    07.3
01EA:  BCF    20.3
01EB:  MOVF   20,W
01EC:  BSF    03.5
01ED:  MOVWF  07
.....................  i2c_write(0xa0);
01EE:  MOVLW  A0
01EF:  BCF    03.5
01F0:  MOVWF  47
01F1:  CALL   0C1
.....................  i2c_write(hi(address));
01F2:  MOVLW  44
01F3:  MOVWF  04
01F4:  MOVF   00,W
01F5:  MOVWF  46
01F6:  MOVWF  47
01F7:  CALL   0C1
.....................  i2c_write(address);
01F8:  MOVF   43,W
01F9:  MOVWF  47
01FA:  CALL   0C1
.....................  i2c_start();
01FB:  BSF    20.4
01FC:  MOVF   20,W
01FD:  BSF    03.5
01FE:  MOVWF  07
01FF:  MOVLW  02
0200:  MOVWF  77
0201:  DECFSZ 77,F
0202:  GOTO   201
0203:  BCF    03.5
0204:  BSF    20.3
0205:  MOVF   20,W
0206:  BSF    03.5
0207:  MOVWF  07
0208:  MOVLW  03
0209:  MOVWF  77
020A:  DECFSZ 77,F
```

```
020B:  GOTO   20A
020C:  BCF    03.5
020D:  BTFSC  07.3
020E:  GOTO   211
020F:  BSF    03.5
0210:  GOTO   20C
0211:  BCF    07.4
0212:  BCF    20.4
0213:  MOVF   20,W
0214:  BSF    03.5
0215:  MOVWF  07
0216:  MOVLW  02
0217:  MOVWF  77
0218:  DECFSZ 77,F
0219:  GOTO   218
021A:  BCF    03.5
021B:  BCF    07.3
021C:  BCF    20.3
021D:  MOVF   20,W
021E:  BSF    03.5
021F:  MOVWF  07
.....  i2c_write(0xa1);
0220:  MOVLW  A1
0221:  BCF    03.5
0222:  MOVWF  47
0223:  CALL   0C1
.....  data=i2c_read(0);
0224:  CLRF   77
0225:  GOTO   122
0226:  MOVF   78,W
0227:  MOVWF  45
.....  i2c_stop();
0228:  BCF    20.4
0229:  MOVF   20,W
022A:  BSF    03.5
022B:  MOVWF  07
022C:  NOP
022D:  BCF    03.5
022E:  BSF    20.3
022F:  MOVF   20,W
0230:  BSF    03.5
0231:  MOVWF  07
0232:  BCF    03.5
0233:  BTFSC  07.3
0234:  GOTO   237
0235:  BSF    03.5
0236:  GOTO   232
0237:  MOVLW  02
0238:  MOVWF  77
0239:  DECFSZ 77,F
023A:  GOTO   239
023B:  NOP
023C:  NOP
023D:  NOP
023E:  BSF    20.4
023F:  MOVF   20,W
0240:  BSF    03.5
0241:  MOVWF  07
0242:  MOVLW  02
0243:  MOVWF  77
0244:  DECFSZ 77,F
0245:  GOTO   244
.....  return(data);
0246:  BCF    03.5
0247:  MOVF   45,W
0248:  MOVWF  78
..... }
```

```
0249:  RETLW  00
.....
.....
..... #use i2c(MASTER, SDA=PIN_C4, SCL=PIN_C3)
.....
..... setup_adc_ports( ALL_DIGITAL);
.....
.....    int   seg;              // global tanımlı degisken
.....    int   isimin,isimax,nemmin,nemmax,isi,nem;
.....
.....
..... void enb() {            //bu birim sisteme yazı yazabilmek için pals gönderilir.
..... bit_set(port_d,0);
*
0004:  BSF    08.0
..... delay_us(100);
0005:  MOVLW  A6
0006:  MOVWF  77
0007:  DECFSZ 77,F
0008:  GOTO   007
0009:  NOP
..... bit_clear(port_d,0);
000A:  BCF    08.0
..... delay_us(100);          // datanın yazılması için gerekli zaman gecikmesi vardır.
000B:  MOVLW  A6
000C:  MOVWF  77
000D:  DECFSZ 77,F
000E:  GOTO   00D
000F:  NOP
..... }
0010:  RETLW  00
.....
.....
..... void lcd_set() {        // lcd için ön ayarlama yapılır
..... port_b=0x00;
*
0024:  CLRF   06
..... bit_clear(port_d,0);
0025:  BCF    08.0
..... bit_clear(port_d,1);
0026:  BCF    08.1
..... bit_clear(port_d,2);
0027:  BCF    08.2
..... enb();
0028:  CALL   004
.....
..... delay_ms(10);
0029:  MOVLW  0A
002A:  MOVWF  44
002B:  CALL   011
..... port_b=0x30;          // 8 bite ayarlandi
002C:  MOVLW  30
002D:  MOVWF  06
..... enb();
002E:  CALL   004
.....
..... delay_ms(10);
002F:  MOVLW  0A
0030:  MOVWF  44
0031:  CALL   011
..... port_b=0x38;          // 2 satr 5x7 font ayarlandi
0032:  MOVLW  38
0033:  MOVWF  06
..... enb();
0034:  CALL   004
.....
..... port_b=0x0d;          //dis on. kur off. blink on
```

```
0035:  MOVLW  0D
0036:  MOVWF  06
..... enb();
0037:  CALL   004
.....
..... port_b=0x06;          //inc on. shift on.
0038:  MOVLW  06
0039:  MOVWF  06
..... enb();
003A:  CALL   004
..... }
003B:  BCF    0A.3
003C:  BCF    0A.4
003D:  GOTO   2E7 (RETURN)
.....
.....
..... void lcd_sil() {      //ekranı siler
..... port_b=0x01;
003E:  MOVLW  01
003F:  MOVWF  06
..... enb();
0040:  CALL   004
..... delay_ms(10);
0041:  MOVLW  0A
0042:  MOVWF  44
0043:  CALL   011
..... }
0044:  RETLW  00
.....
.....
..... void bosluk() {
..... bit_set(port_d,1);
0045:  BSF    08.1
..... port_b=0x20;          //bosluk
0046:  MOVLW  20
0047:  MOVWF  06
..... enb();
0048:  CALL   004
..... }
0049:  RETLW  00
.....
.....
..... void rek_nihat() {
..... bit_clear(port_d,0);
004A:  BCF    08.0
..... bit_clear(port_d,1);
004B:  BCF    08.1
..... bit_clear(port_d,2);
004C:  BCF    08.2
..... port_b=0x80;          //adr birinci satır için
004D:  MOVLW  80
004E:  MOVWF  06
..... enb();
004F:  CALL   004
.....
..... bosluk();
0050:  CALL   045
..... bosluk();
0051:  CALL   045
.....
..... bit_set(port_d,1);
0052:  BSF    08.1
..... port_b=0x4e;          //N
0053:  MOVLW  4E
0054:  MOVWF  06
..... enb();
0055:  CALL   004
```

```
.....................
..... bit_set(port_d,1);
0056:  BSF    08.1
..... port_b=0x69;          //i
0057:  MOVLW  69
0058:  MOVWF  06
..... enb();
0059:  CALL   004
.....................
..... bit_set(port_d,1);
005A:  BSF    08.1
..... port_b=0x68;          //h
005B:  MOVLW  68
005C:  MOVWF  06
..... enb();
005D:  CALL   004
.....................
..... bit_set(port_d,1);
005E:  BSF    08.1
..... port_b=0x61;          //a
005F:  MOVLW  61
0060:  MOVWF  06
..... enb();
0061:  CALL   004
.....................
..... bit_set(port_d,1);
0062:  BSF    08.1
..... port_b=0x74;          //t
0063:  MOVLW  74
0064:  MOVWF  06
..... enb();
0065:  CALL   004
.....................
..... bosluk();
0066:  CALL   045
..... bosluk();
0067:  CALL   045
.....................
..... bit_set(port_d,1);
0068:  BSF    08.1
..... port_b=0x59;          //Y
0069:  MOVLW  59
006A:  MOVWF  06
..... enb();
006B:  CALL   004
.....................
..... bit_set(port_d,1);
006C:  BSF    08.1
..... port_b=0x55;          //U
006D:  MOVLW  55
006E:  MOVWF  06
..... enb();
006F:  CALL   004
.....................
..... bit_set(port_d,1);
0070:  BSF    08.1
..... port_b=0x4b;          //K
0071:  MOVLW  4B
0072:  MOVWF  06
..... enb();
0073:  CALL   004
.....................
..... bit_set(port_d,1);
0074:  BSF    08.1
..... port_b=0x4c;          //L
0075:  MOVLW  4C
0076:  MOVWF  06
```

```
..... enb();
0077:  CALL   004
.....
..... bit_set(port_d,1);
0078:  BSF    08.1
..... port_b=0x55;          //U
0079:  MOVLW  55
007A:  MOVWF  06
..... enb();
007B:  CALL   004
.....
..... bosluk();
007C:  CALL   045
..... bosluk();
007D:  CALL   045
.....
..... }
007E:  BCF    0A.3
007F:  BCF    0A.4
0080:  GOTO   2EC (RETURN)
.....
.....
..... void rek_inu() {
..... bit_clear(port_d,0);
0081:  BCF    08.0
..... bit_clear(port_d,1);
0082:  BCF    08.1
..... bit_clear(port_d,2);
0083:  BCF    08.2
..... port_b=0xc0;      //adr ikinci satır için
0084:  MOVLW  C0
0085:  MOVWF  06
..... enb();
0086:  CALL   004
.....
..... bosluk();
0087:  CALL   045
.....
..... bit_set(port_d,1);
0088:  BSF    08.1
..... port_b=0x49;          //I
0089:  MOVLW  49
008A:  MOVWF  06
..... enb();
008B:  CALL   004
.....
..... bit_set(port_d,1);
008C:  BSF    08.1
..... port_b=0x4e;          //N
008D:  MOVLW  4E
008E:  MOVWF  06
..... enb();
008F:  CALL   004
.....
..... bit_set(port_d,1);
0090:  BSF    08.1
..... port_b=0x4f;          //O
0091:  MOVLW  4F
0092:  MOVWF  06
..... enb();
0093:  CALL   004
.....
..... bit_set(port_d,1);
0094:  BSF    08.1
..... port_b=0x4e;          //N
0095:  MOVLW  4E
0096:  MOVWF  06
```

```
..... enb();
0097: CALL   004
.....
..... bit_set(port_d,1);
0098: BSF    08.1
..... port_b=0x55;          //U
0099: MOVLW  55
009A: MOVWF  06
..... enb();
009B: CALL   004
.....
..... bosluk();
009C: CALL   045
.....
..... bit_set(port_d,1);
009D: BSF    08.1
..... port_b=0x55;          //U
009E: MOVLW  55
009F: MOVWF  06
..... enb();
00A0: CALL   004
.....
..... bit_set(port_d,1);
00A1: BSF    08.1
..... port_b=0x4e;          //N
00A2: MOVLW  4E
00A3: MOVWF  06
..... enb();
00A4: CALL   004
.....
..... bit_set(port_d,1);
00A5: BSF    08.1
..... port_b=0x49;          //I
00A6: MOVLW  49
00A7: MOVWF  06
..... enb();
00A8: CALL   004
.....
..... bit_set(port_d,1);
00A9: BSF    08.1
..... port_b=0x2e;          //.
00AA: MOVLW  2E
00AB: MOVWF  06
..... enb();
00AC: CALL   004
.....
..... bit_set(port_d,1);
00AD: BSF    08.1
..... port_b=0x32;          //2
00AE: MOVLW  32
00AF: MOVWF  06
..... enb();
00B0: CALL   004
.....
..... bit_set(port_d,1);
00B1: BSF    08.1
..... port_b=0x30;          //0
00B2: MOVLW  30
00B3: MOVWF  06
..... enb();
00B4: CALL   004
.....
..... bit_set(port_d,1);
00B5: BSF    08.1
..... port_b=0x30;          //0
00B6: MOVLW  30
00B7: MOVWF  06
```

```
..... enb();
00B8:  CALL   004
.....
..... bit_set(port_d,1);
00B9:  BSF    08.1
..... port_b=0x36;          //6
00BA:  MOVLW  36
00BB:  MOVWF  06
..... enb();
00BC:  CALL   004
.....
..... bosluk();
00BD:  CALL   045
..... }
00BE:  BCF    0A.3
00BF:  BCF    0A.4
00C0:  GOTO   2ED (RETURN)
.....
.....
..... int segment1(int seg)  {
..... switch(seg) {                    // segmente sayı girmek için
*
024A:  MOVF   43,W
024B:  XORLW  0B
024C:  BTFSC  03.2
024D:  GOTO   26D
024E:  XORLW  0B
024F:  BTFSC  03.2
0250:  GOTO   270
0251:  XORLW  01
0252:  BTFSC  03.2
0253:  GOTO   273
0254:  XORLW  03
0255:  BTFSC  03.2
0256:  GOTO   276
0257:  XORLW  01
0258:  BTFSC  03.2
0259:  GOTO   279
025A:  XORLW  07
025B:  BTFSC  03.2
025C:  GOTO   27C
025D:  XORLW  01
025E:  BTFSC  03.2
025F:  GOTO   27F
0260:  XORLW  03
0261:  BTFSC  03.2
0262:  GOTO   282
0263:  XORLW  01
0264:  BTFSC  03.2
0265:  GOTO   285
0266:  XORLW  0F
0267:  BTFSC  03.2
0268:  GOTO   288
0269:  XORLW  01
026A:  BTFSC  03.2
026B:  GOTO   28B
026C:  GOTO   28E
.....
.....     case 0x0B  : seg=0x20; break;
026D:  MOVLW  20
026E:  MOVWF  43
026F:  GOTO   291
.....     case 0x00  : seg=0x30; break;
0270:  MOVLW  30
0271:  MOVWF  43
0272:  GOTO   291
.....     case 1  : seg=0x31;break;
```

```
0273:  MOVLW  31
0274:  MOVWF  43
0275:  GOTO   291
.....      case 2  : seg=0x32; break;
0276:  MOVLW  32
0277:  MOVWF  43
0278:  GOTO   291
.....      case 3  : seg=0x33; break;
0279:  MOVLW  33
027A:  MOVWF  43
027B:  GOTO   291
.....      case 4  : seg=0x34; break;
027C:  MOVLW  34
027D:  MOVWF  43
027E:  GOTO   291
.....      case 5  : seg=0x35; break;
027F:  MOVLW  35
0280:  MOVWF  43
0281:  GOTO   291
.....      case 6  : seg=0x36; break;
0282:  MOVLW  36
0283:  MOVWF  43
0284:  GOTO   291
.....      case 7  : seg=0x37; break;
0285:  MOVLW  37
0286:  MOVWF  43
0287:  GOTO   291
.....      case 8  : seg=0x38; break;
0288:  MOVLW  38
0289:  MOVWF  43
028A:  GOTO   291
.....      case 9  : seg=0x39; break;
028B:  MOVLW  39
028C:  MOVWF  43
028D:  GOTO   291
.....      default : seg=0x30; break;
028E:  MOVLW  30
028F:  MOVWF  43
0290:  GOTO   291
.....       // seg=0x23; break;  // # işareti
..... }
..... return seg;
0291:  MOVF   43,W
0292:  MOVWF  78
..... }
0293:  RETLW  00
.....
..... //************************ ana program ************************//
.....
.....
..... main()  {
.....
0294:  CLRF   04
0295:  MOVLW  1F
0296:  ANDWF  03,F
0297:  BSF    03.5
0298:  BSF    1F.0
0299:  BSF    1F.1
029A:  BSF    1F.2
029B:  BCF    1F.3
.....  // int16 toplam,toplm2,toplm1;
.....
.....    int toplam;
.....    int deg1,deg2,deg3,deg4,i,j,sayici;                    // sayıcılar için kullanılanlar
.....    int isimin1,isimin2,isimin3,nemmin1,nemmin2,nemmin3;
.....    int isimax1,isimax2,isimax3,nemmax1,nemmax2,nemmax3;
.....    int isi1,isi2,isi3,nem1,nem2,nem3;
```

```
.....    int durum;
.....
.....
.....    set_tris_b(all_out);            //çıkış yap
*
029F:  MOVLW  00
02A0:  BSF    03.5
02A1:  MOVWF  06
.....    set_tris_d(all_out);            //
02A2:  MOVWF  08
.....    set_tris_a(all_in);             //giriş yap
02A3:  MOVLW  FF
02A4:  MOVWF  05
.....    set_tris_e(all_in);
02A5:  BSF    09.0
02A6:  BSF    09.1
02A7:  BSF    09.2
.....
..... bit_clear(port_d,7);
02A8:  BCF    03.5
02A9:  BCF    08.7
..... bit_clear(port_d,6);
02AA:  BCF    08.6
..... bit_clear(port_d,5);
02AB:  BCF    08.5
..... bit_clear(port_d,4);                // Roleler sıfırlanır
02AC:  BCF    08.4
..... bit_set(port_d,3);                  //power için ON
02AD:  BSF    08.3
.....
.....
..... isimin =read_eeprom(0x00);          // ısı ve nem bilgileri yardımcı bellege alınır
02AE:  BSF    03.6
02AF:  CLRF   0D
02B0:  BSF    03.5
02B1:  BCF    0C.7
02B2:  BSF    0C.0
02B3:  BCF    03.5
02B4:  MOVF   0C,W
02B5:  BCF    03.6
02B6:  MOVWF  22
..... isimax =read_eeprom(0x01);
02B7:  MOVLW  01
02B8:  BSF    03.6
02B9:  MOVWF  0D
02BA:  BSF    03.5
02BB:  BCF    0C.7
02BC:  BSF    0C.0
02BD:  BCF    03.5
02BE:  MOVF   0C,W
02BF:  BCF    03.6
02C0:  MOVWF  23
..... nemmin =read_eeprom(0x02);
02C1:  MOVLW  02
02C2:  BSF    03.6
02C3:  MOVWF  0D
02C4:  BSF    03.5
02C5:  BCF    0C.7
02C6:  BSF    0C.0
02C7:  BCF    03.5
02C8:  MOVF   0C,W
02C9:  BCF    03.6
02CA:  MOVWF  24
..... nemmax =read_eeprom(0x03);         // ısı ve nem bilgileri yardımcı bellege alınır
02CB:  MOVLW  03
02CC:  BSF    03.6
02CD:  MOVWF  0D
```

```
02CE:  BSF    03.5
02CF:  BCF    0C.7
02D0:  BSF    0C.0
02D1:  BCF    03.5
02D2:  MOVF   0C,W
02D3:  BCF    03.6
02D4:  MOVWF  25
.....................
..................... isimin =55;
02D5:  MOVLW  37
02D6:  MOVWF  22
..................... isimax =44;
02D7:  MOVLW  2C
02D8:  MOVWF  23
..................... nemmin =33;
02D9:  MOVLW  21
02DA:  MOVWF  24
..................... nemmax =22;
02DB:  MOVLW  16
02DC:  MOVWF  25
.....................
..................... sil:                                  // tüm değerler sıfırlanır
.....................
..................... toplam=durum=0;                        // global degisken
02DD:  CLRF   42
02DE:  MOVF   42,W
02DF:  MOVWF  28
..................... deg1=deg2=deg3=deg4=seg=0;
02E0:  CLRF   21
02E1:  MOVF   21,W
02E2:  MOVWF  2C
02E3:  MOVWF  2B
02E4:  MOVWF  2A
02E5:  MOVWF  29
.....................
..................... lcd_set();                             //lcd nin çalışması için gerekli olan temel girdileri girmektedir
02E6:  GOTO   024
..................... lcd_sil();
02E7:  CALL   03E
..................... delay_ms(100);
02E8:  MOVLW  64
02E9:  MOVWF  44
02EA:  CALL   011
..................... rek_nihat();                           //sistem için reklam girilmektedir.
02EB:  GOTO   04A
..................... rek_inu();
02EC:  GOTO   081
..................... delay_ms(2000);
02ED:  MOVLW  08
02EE:  MOVWF  43
02EF:  MOVLW  FA
02F0:  MOVWF  44
02F1:  CALL   011
02F2:  DECFSZ 43,F
02F3:  GOTO   2EF
.....................
.....................
.....................  basla:                                                //##############################Program  BASLANGICI
******************************
.....................
..................... if (sayici>=250) {                //amaç 2 saniyede 1 defa veri okumak
02F4:  MOVF   2F,W
02F5:  SUBLW  F9
02F6:  BTFSC  03.0
02F7:  GOTO   305
..................... isi = read_ext_eeprom(0x00011);    //SHT11 deki ilgili ısı ve nem verileri okunur
02F8:  CLRF   44
```

```
02F9:  MOVLW  11
02FA:  MOVWF  43
02FB:  CALL   17C
02FC:  MOVF   78,W
02FD:  MOVWF  26
..... nem = read_ext_eeprom(0x00101);
02FE:  MOVLW  01
02FF:  MOVWF  44
0300:  MOVWF  43
0301:  CALL   17C
0302:  MOVF   78,W
0303:  MOVWF  27
..... sayici=0;
0304:  CLRF   2F
..... }
..... sayici=sayici+1;
0305:  MOVLW  01
0306:  ADDWF  2F,F
.....
.....
..... for (i=1;i<7;++i) {
0307:  MOVWF  2D
0308:  MOVF   2D,W
0309:  SUBLW  06
030A:  BTFSS  03.0
030B:  GOTO   394
.....
..... deg1=deg2=deg3=deg4=0;
030C:  CLRF   2C
030D:  MOVF   2C,W
030E:  MOVWF  2B
030F:  MOVWF  2A
0310:  MOVWF  29
..... toplam=0;
0311:  CLRF   28
.....
..... if (i==1)
0312:  DECFSZ 2D,W
0313:  GOTO   316
..... toplam = isimin;
0314:  MOVF   22,W
0315:  MOVWF  28
.....
..... if (i==2)
0316:  MOVF   2D,W
0317:  SUBLW  02
0318:  BTFSS  03.2
0319:  GOTO   31C
..... toplam = isimax;
031A:  MOVF   23,W
031B:  MOVWF  28
.....
..... if (i==3)
031C:  MOVF   2D,W
031D:  SUBLW  03
031E:  BTFSS  03.2
031F:  GOTO   322
..... toplam = nemmin;
0320:  MOVF   24,W
0321:  MOVWF  28
.....
..... if (i==4)
0322:  MOVF   2D,W
0323:  SUBLW  04
0324:  BTFSS  03.2
0325:  GOTO   328
..... toplam = nemmax;
```

```
0326:  MOVF   25,W
0327:  MOVWF  28
.....................
..................... if (i==5)
0328:  MOVF   2D,W
0329:  SUBLW  05
032A:  BTFSS  03.2
032B:  GOTO   32E
..................... toplam = isi;
032C:  MOVF   26,W
032D:  MOVWF  28
.....................
..................... if (i==6)
032E:  MOVF   2D,W
032F:  SUBLW  06
0330:  BTFSS  03.2
0331:  GOTO   334
..................... toplam = nem;
0332:  MOVF   27,W
0333:  MOVWF  28
.....................
..................... toplam=toplam+1;
0334:  MOVLW  01
0335:  ADDWF  28,F
.....................
..................... don1a:            //hexadesimal sayıyı desimale çevirir.
..................... deg1=deg1+1;
0336:  MOVLW  01
0337:  ADDWF  29,F
..................... if (deg1>9) {
0338:  MOVF   29,W
0339:  SUBLW  09
033A:  BTFSC  03.0
033B:  GOTO   33F
..................... deg1=0;
033C:  CLRF   29
..................... deg2=deg2+1;
033D:  MOVLW  01
033E:  ADDWF  2A,F
..................... }
..................... if (deg2>9) {
033F:  MOVF   2A,W
0340:  SUBLW  09
0341:  BTFSC  03.0
0342:  GOTO   346
..................... deg2=0;
0343:  CLRF   2A
..................... deg3=deg3+1;
0344:  MOVLW  01
0345:  ADDWF  2B,F
..................... }
..................... if (deg3>9) {
0346:  MOVF   2B,W
0347:  SUBLW  09
0348:  BTFSC  03.0
0349:  GOTO   34D
..................... deg3=0;
034A:  CLRF   2B
..................... deg4=deg4+1;
034B:  MOVLW  01
034C:  ADDWF  2C,F
..................... }
..................... if (deg4>9) {
034D:  MOVF   2C,W
034E:  SUBLW  09
034F:  BTFSC  03.0
0350:  GOTO   352
```

..... deg4=0;
0351: CLRF 2C
..... }
..... toplam=toplam-1;
0352: MOVLW 01
0353: SUBWF 28,F
..... if (toplam>1)
0354: MOVF 28,W
0355: SUBLW 01
0356: BTFSS 03.0
..... goto don1a;
0357: GOTO 336
.....
..... if (i==1) {
0358: DECFSZ 2D,W
0359: GOTO 360
..... isimin1 = deg1;
035A: MOVF 29,W
035B: MOVWF 30
..... isimin2 = deg2;
035C: MOVF 2A,W
035D: MOVWF 31
..... isimin3 = deg3;
035E: MOVF 2B,W
035F: MOVWF 32
..... }
..... if (i==2) {
0360: MOVF 2D,W
0361: SUBLW 02
0362: BTFSS 03.2
0363: GOTO 36A
..... isimax1 = deg1;
0364: MOVF 29,W
0365: MOVWF 36
..... isimax2 = deg2;
0366: MOVF 2A,W
0367: MOVWF 37
..... isimax3 = deg3;
0368: MOVF 2B,W
0369: MOVWF 38
..... }
..... if (i==3) {
036A: MOVF 2D,W
036B: SUBLW 03
036C: BTFSS 03.2
036D: GOTO 374
..... nemmin1 = deg1;
036E: MOVF 29,W
036F: MOVWF 33
..... nemmin2 = deg2;
0370: MOVF 2A,W
0371: MOVWF 34
..... nemmin3 = deg3;
0372: MOVF 2B,W
0373: MOVWF 35
..... }
..... if (i==4) {
0374: MOVF 2D,W
0375: SUBLW 04
0376: BTFSS 03.2
0377: GOTO 37E
..... nemmax1 = deg1;
0378: MOVF 29,W
0379: MOVWF 39
..... nemmax2 = deg2;
037A: MOVF 2A,W
037B: MOVWF 3A

..... nemmax3 = deg3;
037C: MOVF 2B,W
037D: MOVWF 3B
..... }
..... if (i==5) {
037E: MOVF 2D,W
037F: SUBLW 05
0380: BTFSS 03.2
0381: GOTO 388
..... isi1 = deg1;
0382: MOVF 29,W
0383: MOVWF 3C
..... isi2 = deg2;
0384: MOVF 2A,W
0385: MOVWF 3D
..... isi3 = deg3;
0386: MOVF 2B,W
0387: MOVWF 3E
..... }
..... if (i==6) {
0388: MOVF 2D,W
0389: SUBLW 06
038A: BTFSS 03.2
038B: GOTO 392
..... nem1 = deg1;
038C: MOVF 29,W
038D: MOVWF 3F
..... nem2 = deg2;
038E: MOVF 2A,W
038F: MOVWF 40
..... nem3 = deg3;
0390: MOVF 2B,W
0391: MOVWF 41
..... }
..... } // hex cevirim tablosu
0392: INCF 2D,F
0393: GOTO 308
.....
..... if (!bit_test(durum,0)) { // sistem setup modunda değilse
0394: BTFSC 42.0
0395: GOTO 44F
.....
..... lcd_sil();
0396: CALL 03E
.....
..... bit_clear(port_d,0); // ****************** 1. satır için ISI DEGERI yazılır ******************
0397: BCF 08.0
..... bit_clear(port_d,1);
0398: BCF 08.1
..... bit_clear(port_d,2);
0399: BCF 08.2
..... port_b=0x80; //adr
039A: MOVLW 80
039B: MOVWF 06
..... enb();
039C: CALL 004
.....
..... bit_set(port_d,1);
039D: BSF 08.1
..... port_b=0x49; //i
039E: MOVLW 49
039F: MOVWF 06
..... enb();
03A0: CALL 004
.....
..... bit_set(port_d,1);

```
03A1:  BSF    08.1
..... port_b=0x53;         //s
03A2:  MOVLW  53
03A3:  MOVWF  06
..... enb();
03A4:  CALL   004
.....
..... bit_set(port_d,1);
03A5:  BSF    08.1
..... port_b=0x49;         //i
03A6:  MOVLW  49
03A7:  MOVWF  06
..... enb();
03A8:  CALL   004
.....
..... bosluk();
03A9:  CALL   045
.....
..... bit_set(port_d,1);
03AA:  BSF    08.1
..... port_b=0x44;         //d
03AB:  MOVLW  44
03AC:  MOVWF  06
..... enb();
03AD:  CALL   004
.....
..... bit_set(port_d,1);
03AE:  BSF    08.1
..... port_b=0x65;         //e
03AF:  MOVLW  65
03B0:  MOVWF  06
..... enb();
03B1:  CALL   004
.....
..... bit_set(port_d,1);
03B2:  BSF    08.1
..... port_b=0x67;         //g
03B3:  MOVLW  67
03B4:  MOVWF  06
..... enb();
03B5:  CALL   004
.....
..... bit_set(port_d,1);
03B6:  BSF    08.1
..... port_b=0x65;         //e
03B7:  MOVLW  65
03B8:  MOVWF  06
..... enb();
03B9:  CALL   004
.....
..... bit_set(port_d,1);
03BA:  BSF    08.1
..... port_b=0x72;         //r
03BB:  MOVLW  72
03BC:  MOVWF  06
..... enb();
03BD:  CALL   004
.....
..... bit_set(port_d,1);
03BE:  BSF    08.1
..... port_b=0x69;         //i
03BF:  MOVLW  69
03C0:  MOVWF  06
..... enb();
03C1:  CALL   004
.....
..... bit_set(port_d,1);
```

```
03C2:  BSF    08.1
..... port_b=0x3d;          //=
03C3:  MOVLW  3D
03C4:  MOVWF  06
..... enb();
03C5:  CALL   004
.....
..... bit_set(port_d,1);
03C6:  BSF    08.1
..... seg=isi3;             //isi3
03C7:  MOVF   3E,W
03C8:  MOVWF  21
..... port_b=segment1(seg);
03C9:  MOVF   21,W
03CA:  MOVWF  43
03CB:  CALL   24A
03CC:  MOVF   78,W
03CD:  MOVWF  06
..... enb();
03CE:  CALL   004
.....
..... bit_set(port_d,1);
03CF:  BSF    08.1
..... seg=isi2;             //isi2
03D0:  MOVF   3D,W
03D1:  MOVWF  21
..... port_b=segment1(seg);
03D2:  MOVF   21,W
03D3:  MOVWF  43
03D4:  CALL   24A
03D5:  MOVF   78,W
03D6:  MOVWF  06
..... enb();
03D7:  CALL   004
.....
..... bit_set(port_d,1);
03D8:  BSF    08.1
..... seg=isi1;             //isi1
03D9:  MOVF   3C,W
03DA:  MOVWF  21
..... port_b=segment1(seg);
03DB:  MOVF   21,W
03DC:  MOVWF  43
03DD:  CALL   24A
03DE:  MOVF   78,W
03DF:  MOVWF  06
..... enb();
03E0:  CALL   004
.....
..... bosluk();
03E1:  CALL   045
.....
..... //role çekili ise bu harf yazılır
.....
..... if ((bit_test(durum,4)) | (bit_test(durum,5))) {
03E2:  MOVLW  00
03E3:  BTFSC  42.4
03E4:  MOVLW  01
03E5:  MOVWF  43
03E6:  MOVLW  00
03E7:  BTFSC  42.5
03E8:  MOVLW  01
03E9:  IORWF  43,W
03EA:  XORLW  00
03EB:  BTFSC  03.2
03EC:  GOTO   3F2
..... bit_set(port_d,1);
```

```
03ED:  BSF    08.1
..... port_b=0x52;          //r
03EE:  MOVLW  52
03EF:  MOVWF  06
..... enb();
03F0:  CALL   004
..... }
..... else
03F1:  GOTO   3F3
..... bosluk();
03F2:  CALL   045
.....
.....
..... satr2:
.....                               // ************ 2. satır için NEM DEGERI yazılır ****************
..... bit_clear(port_d,0);
03F3:  BCF    08.0
..... bit_clear(port_d,1);
03F4:  BCF    08.1
..... bit_clear(port_d,2);
03F5:  BCF    08.2
..... port_b=0xc0;      //adr
03F6:  MOVLW  C0
03F7:  MOVWF  06
..... enb();
03F8:  CALL   004
.....
..... bit_set(port_d,1);
03F9:  BSF    08.1
..... port_b=0x4e;          //N
03FA:  MOVLW  4E
03FB:  MOVWF  06
..... enb();
03FC:  CALL   004
.....
..... bit_set(port_d,1);
03FD:  BSF    08.1
..... port_b=0x45;          //E
03FE:  MOVLW  45
03FF:  MOVWF  06
..... enb();
0400:  CALL   004
.....
..... bit_set(port_d,1);
0401:  BSF    08.1
..... port_b=0x4d;          //M
0402:  MOVLW  4D
0403:  MOVWF  06
..... enb();
0404:  CALL   004
.....
..... bosluk();
0405:  CALL   045
.....
..... bit_set(port_d,1);
0406:  BSF    08.1
..... port_b=0x44;          //d
0407:  MOVLW  44
0408:  MOVWF  06
..... enb();
0409:  CALL   004
.....
..... bit_set(port_d,1);
040A:  BSF    08.1
..... port_b=0x65;          //e
040B:  MOVLW  65
040C:  MOVWF  06
```

..... enb();
040D: CALL 004
.....
..... bit_set(port_d,1);
040E: BSF 08.1
..... port_b=0x67; //g
040F: MOVLW 67
0410: MOVWF 06
..... enb();
0411: CALL 004
.....
..... bit_set(port_d,1);
0412: BSF 08.1
..... port_b=0x65; //e
0413: MOVLW 65
0414: MOVWF 06
..... enb();
0415: CALL 004
.....
..... bit_set(port_d,1);
0416: BSF 08.1
..... port_b=0x72; //r
0417: MOVLW 72
0418: MOVWF 06
..... enb();
0419: CALL 004
.....
..... bit_set(port_d,1);
041A: BSF 08.1
..... port_b=0x69; //i
041B: MOVLW 69
041C: MOVWF 06
..... enb();
041D: CALL 004
.....
..... bit_set(port_d,1);
041E: BSF 08.1
..... port_b=0x3d; //=
041F: MOVLW 3D
0420: MOVWF 06
..... enb();
0421: CALL 004
.....
..... bit_set(port_d,1);
0422: BSF 08.1
..... seg=nem3; //NEM3
0423: MOVF 41,W
0424: MOVWF 21
..... port_b=segment1(seg);
0425: MOVF 21,W
0426: MOVWF 43
0427: CALL 24A
0428: MOVF 78,W
0429: MOVWF 06
..... enb();
042A: CALL 004
.....
..... bit_set(port_d,1);
042B: BSF 08.1
..... seg=nem2; //NEM2
042C: MOVF 40,W
042D: MOVWF 21
..... port_b=segment1(seg);
042E: MOVF 21,W
042F: MOVWF 43
0430: CALL 24A
0431: MOVF 78,W

```
0432:  MOVWF  06
..... enb();
0433:  CALL   004
.....
..... bit_set(port_d,1);
0434:  BSF    08.1
..... seg=nem1;               //NEM1
0435:  MOVF   3F,W
0436:  MOVWF  21
..... port_b=segment1(seg);
0437:  MOVF   21,W
0438:  MOVWF  43
0439:  CALL   24A
043A:  MOVF   78,W
043B:  MOVWF  06
..... enb();
043C:  CALL   004
.....
..... bosluk();
043D:  CALL   045
..... //role çekili ise bu harf yazılır
.....
..... if ((bit_test(durum,6)) | (bit_test(durum,7))) {
043E:  MOVLW  00
043F:  BTFSC  42.6
0440:  MOVLW  01
0441:  MOVWF  43
0442:  MOVLW  00
0443:  BTFSC  42.7
0444:  MOVLW  01
0445:  IORWF  43,W
0446:  XORLW  00
0447:  BTFSC  03.2
0448:  GOTO   44E
..... bit_set(port_d,1);
0449:  BSF    08.1
..... port_b=0x52;          //r
044A:  MOVLW  52
044B:  MOVWF  06
..... enb();
044C:  CALL   004
..... }
..... else
044D:  GOTO   44F
..... bosluk();
044E:  CALL   045
.....
..... }     // sistem setup modunda değilse
.....
.....
.....
..... if ( isi<=isimin){          //isi belirtilen değerin altında veya eşitse
044F:  MOVF   26,W
0450:  SUBWF  22,W
0451:  BTFSS  03.0
0452:  GOTO   456
..... bit_set(port_d,4);         //isiticiyi çalıştır.
0453:  BSF    08.4
..... bit_set(durum,4);
0454:  BSF    42.4
..... }
..... else
0455:  GOTO   458
..... {
..... bit_clear(port_d,4);
0456:  BCF    08.4
..... bit_clear(durum,4);
```

```
0457:  BCF    42.4
..... }
.....
..... if ( isi>isimax){          //isi belirtilen değerin üstünde ise
0458:  MOVF   26,W
0459:  SUBWF  23,W
045A:  BTFSC  03.0
045B:  GOTO   45F
..... bit_set(port_d,5);         //fanı çalıştır.
045C:  BSF    08.5
..... bit_set(durum,5);
045D:  BSF    42.5
..... }
..... else
045E:  GOTO   461
..... {
..... bit_clear(port_d,5);
045F:  BCF    08.5
..... bit_clear(port_d,5);
0460:  BCF    08.5
..... }
.....
..... if ( nem<=nemmin){          //nem belirtilen değerin altında veya eşitse
0461:  MOVF   27,W
0462:  SUBWF  24,W
0463:  BTFSS  03.0
0464:  GOTO   468
..... bit_set(port_d,6);         //spreyi çalıştır.
0465:  BSF    08.6
..... bit_set(durum,6);
0466:  BSF    42.6
..... }
..... else
0467:  GOTO   46A
..... {
..... bit_clear(port_d,6);
0468:  BCF    08.6
..... bit_clear(durum,6);
0469:  BCF    42.6
..... }
.....
..... if ( nem>nemmax){          //nem belirtilen değerin üstünde ise
046A:  MOVF   27,W
046B:  SUBWF  25,W
046C:  BTFSC  03.0
046D:  GOTO   471
..... bit_set(port_d,7);         //havalandırma fanını çalıştır.
046E:  BSF    08.7
..... bit_set(durum,7);
046F:  BSF    42.7
..... }
..... else
0470:  GOTO   473
..... {
..... bit_clear(port_d,7);
0471:  BCF    08.7
..... bit_clear(durum,7);
0472:  BCF    42.7
..... }
.....
..... if ((!bit_test(port_e,0)) & (!bit_test(port_e,1)) & (!bit_test(durum,0)))       // setup modu aktif olur
0473:  MOVLW  00
0474:  BTFSS  09.0
0475:  MOVLW  01
0476:  MOVWF  43
0477:  MOVLW  00
0478:  BTFSS  09.1
```

```
0479:  MOVLW  01
047A:  ANDWF  43,W
047B:  MOVWF  44
047C:  MOVLW  00
047D:  BTFSS  42.0
047E:  MOVLW  01
047F:  ANDWF  44,W
0480:  XORLW  00
0481:  BTFSS  03.2
..... bit_set(durum,0);
0482:  BSF    42.0
.....
..... if ((!bit_test(port_a,4)) &  (bit_test(durum,0)))  {       // setup modundan çıkar
0483:  MOVLW  00
0484:  BTFSS  05.4
0485:  MOVLW  01
0486:  MOVWF  43
0487:  MOVLW  00
0488:  BTFSC  42.0
0489:  MOVLW  01
048A:  ANDWF  43,W
048B:  XORLW  00
048C:  BTFSC  03.2
048D:  GOTO   503
..... j=0;
048E:  CLRF   2E
..... bit_clear(durum,0);
048F:  BCF    42.0
..... write_eeprom(0x00,isimin);      // ısı ve nem bilgileri eeprom a alınır
0490:  BSF    03.6
0491:  CLRF   0D
0492:  BCF    03.6
0493:  MOVF   22,W
0494:  BSF    03.6
0495:  MOVWF  0C
0496:  BSF    03.5
0497:  BCF    0C.7
0498:  BSF    0C.2
0499:  BCF    03.5
049A:  BCF    03.6
049B:  MOVF   0B,W
049C:  MOVWF  77
049D:  BCF    0B.7
049E:  BSF    03.5
049F:  BSF    03.6
04A0:  MOVLW  55
04A1:  MOVWF  0D
04A2:  MOVLW  AA
04A3:  MOVWF  0D
04A4:  BSF    0C.1
04A5:  BTFSC  0C.1
04A6:  GOTO   4A5
04A7:  BCF    0C.2
04A8:  MOVF   77,W
04A9:  BCF    03.5
04AA:  BCF    03.6
04AB:  IORWF  0B,F
..... write_eeprom(0x01,isimax);
04AC:  MOVLW  01
04AD:  BSF    03.6
04AE:  MOVWF  0D
04AF:  BCF    03.6
04B0:  MOVF   23,W
04B1:  BSF    03.6
04B2:  MOVWF  0C
04B3:  BSF    03.5
04B4:  BCF    0C.7
```

```
04B5:  BSF    0C.2
04B6:  BCF    03.5
04B7:  BCF    03.6
04B8:  MOVF   0B,W
04B9:  MOVWF  77
04BA:  BCF    0B.7
04BB:  BSF    03.5
04BC:  BSF    03.6
04BD:  MOVLW  55
04BE:  MOVWF  0D
04BF:  MOVLW  AA
04C0:  MOVWF  0D
04C1:  BSF    0C.1
04C2:  BTFSC  0C.1
04C3:  GOTO   4C2
04C4:  BCF    0C.2
04C5:  MOVF   77,W
04C6:  BCF    03.5
04C7:  BCF    03.6
04C8:  IORWF  0B,F
..................... write_eeprom(0x02,nemmin);
04C9:  MOVLW  02
04CA:  BSF    03.6
04CB:  MOVWF  0D
04CC:  BCF    03.6
04CD:  MOVF   24,W
04CE:  BSF    03.6
04CF:  MOVWF  0C
04D0:  BSF    03.5
04D1:  BCF    0C.7
04D2:  BSF    0C.2
04D3:  BCF    03.5
04D4:  BCF    03.6
04D5:  MOVF   0B,W
04D6:  MOVWF  77
04D7:  BCF    0B.7
04D8:  BSF    03.5
04D9:  BSF    03.6
04DA:  MOVLW  55
04DB:  MOVWF  0D
04DC:  MOVLW  AA
04DD:  MOVWF  0D
04DE:  BSF    0C.1
04DF:  BTFSC  0C.1
04E0:  GOTO   4DF
04E1:  BCF    0C.2
04E2:  MOVF   77,W
04E3:  BCF    03.5
04E4:  BCF    03.6
04E5:  IORWF  0B,F
..................... write_eeprom(0x03,nemmax);
04E6:  MOVLW  03
04E7:  BSF    03.6
04E8:  MOVWF  0D
04E9:  BCF    03.6
04EA:  MOVF   25,W
04EB:  BSF    03.6
04EC:  MOVWF  0C
04ED:  BSF    03.5
04EE:  BCF    0C.7
04EF:  BSF    0C.2
04F0:  BCF    03.5
04F1:  BCF    03.6
04F2:  MOVF   0B,W
04F3:  MOVWF  77
04F4:  BCF    0B.7
04F5:  BSF    03.5
```

```
04F6:  BSF    03.6
04F7:  MOVLW  55
04F8:  MOVWF  0D
04F9:  MOVLW  AA
04FA:  MOVWF  0D
04FB:  BSF    0C.1
04FC:  BTFSC  0C.1
04FD:  GOTO   4FC
04FE:  BCF    0C.2
04FF:  MOVF   77,W
0500:  BCF    03.5
0501:  BCF    03.6
0502:  IORWF  0B,F
..................... }
.....................
..................... if ((!bit_test(port_a,5)) &  (bit_test(durum,0)) &  (!bit_test(durum,1))) {
0503:  MOVLW  00
0504:  BTFSS  05.5
0505:  MOVLW  01
0506:  MOVWF  43
0507:  MOVLW  00
0508:  BTFSC  42.0
0509:  MOVLW  01
050A:  ANDWF  43,W
050B:  MOVWF  44
050C:  MOVLW  00
050D:  BTFSS  42.1
050E:  MOVLW  01
050F:  ANDWF  44,W
0510:  XORLW  00
0511:  BTFSC  03.2
0512:  GOTO   51A
..................... j=j+1;
0513:  MOVLW  01
0514:  ADDWF  2E,F
..................... bit_set(durum,1);
0515:  BSF    42.1
..................... if (j>=4)
0516:  MOVF   2E,W
0517:  SUBLW  03
0518:  BTFSS  03.0
..................... j=0;
0519:  CLRF   2E
..................... }
.....................
..................... if ((bit_test(port_a,5)) & (bit_test(durum,1)))
051A:  MOVLW  00
051B:  BTFSC  05.5
051C:  MOVLW  01
051D:  MOVWF  43
051E:  MOVLW  00
051F:  BTFSC  42.1
0520:  MOVLW  01
0521:  ANDWF  43,W
0522:  XORLW  00
0523:  BTFSS  03.2
..................... bit_clear(durum,1);
0524:  BCF    42.1
.....................
.....................
..................... if ((!bit_test(port_e,1)) &  (bit_test(durum,0)) & (j==0) & (!bit_test(durum,2))) {
0525:  MOVLW  00
0526:  BTFSS  09.1
0527:  MOVLW  01
0528:  MOVWF  43
0529:  MOVLW  00
052A:  BTFSC  42.0
```

```
052B:  MOVLW  01
052C:  ANDWF  43,W
052D:  MOVWF  44
052E:  MOVF   2E,F
052F:  BTFSC  03.2
0530:  GOTO   533
0531:  MOVLW  00
0532:  GOTO   534
0533:  MOVLW  01
0534:  ANDWF  44,W
0535:  MOVWF  45
0536:  MOVLW  00
0537:  BTFSS  42.2
0538:  MOVLW  01
0539:  ANDWF  45,W
053A:  XORLW  00
053B:  BTFSC  03.2
053C:  GOTO   543
..... bit_set(durum,2);    //sistem setup modunda ise isi miktarını azaltır
053D:  BSF    42.2
..... if (isimin>0)
053E:  MOVF   22,F
053F:  BTFSC  03.2
0540:  GOTO   543
..... isimin=isimin-1;
0541:  MOVLW  01
0542:  SUBWF  22,F
..... }
.....
..... if ((!bit_test(port_e,0)) &  (bit_test(durum,0)) & (j==0) & (!bit_test(durum,2))) {
0543:  MOVLW  00
0544:  BTFSS  09.0
0545:  MOVLW  01
0546:  MOVWF  43
0547:  MOVLW  00
0548:  BTFSC  42.0
0549:  MOVLW  01
054A:  ANDWF  43,W
054B:  MOVWF  44
054C:  MOVF   2E,F
054D:  BTFSC  03.2
054E:  GOTO   551
054F:  MOVLW  00
0550:  GOTO   552
0551:  MOVLW  01
0552:  ANDWF  44,W
0553:  MOVWF  45
0554:  MOVLW  00
0555:  BTFSS  42.2
0556:  MOVLW  01
0557:  ANDWF  45,W
0558:  XORLW  00
0559:  BTFSC  03.2
055A:  GOTO   562
..... bit_set(durum,2);    //sistem setup modunda ise isi miktarını artırır
055B:  BSF    42.2
..... if (isimin<100)
055C:  MOVF   22,W
055D:  SUBLW  63
055E:  BTFSS  03.0
055F:  GOTO   562
..... isimin=isimin+1;
0560:  MOVLW  01
0561:  ADDWF  22,F
..... }
.....
..... if ((!bit_test(port_e,0)) &  (bit_test(durum,0)) & (j==1) & (!bit_test(durum,2))) {
```

```
0562:  MOVLW  00
0563:  BTFSS  09.0
0564:  MOVLW  01
0565:  MOVWF  43
0566:  MOVLW  00
0567:  BTFSC  42.0
0568:  MOVLW  01
0569:  ANDWF  43,W
056A:  MOVWF  44
056B:  DECFSZ 2E,W
056C:  GOTO   56E
056D:  GOTO   570
056E:  MOVLW  00
056F:  GOTO   571
0570:  MOVLW  01
0571:  ANDWF  44,W
0572:  MOVWF  45
0573:  MOVLW  00
0574:  BTFSS  42.2
0575:  MOVLW  01
0576:  ANDWF  45,W
0577:  XORLW  00
0578:  BTFSC  03.2
0579:  GOTO   581
..... bit_set(durum,2);    //sistem setup modunda ise isi miktarını artırır
057A:  BSF    42.2
..... if (isimax<100)
057B:  MOVF   23,W
057C:  SUBLW  63
057D:  BTFSS  03.0
057E:  GOTO   581
..... isimax=isimax+1;
057F:  MOVLW  01
0580:  ADDWF  23,F
..... }
.....
..... if ((!bit_test(port_e,1)) &  (bit_test(durum,0)) & (j==1) & (!bit_test(durum,2))) {
0581:  MOVLW  00
0582:  BTFSS  09.1
0583:  MOVLW  01
0584:  MOVWF  43
0585:  MOVLW  00
0586:  BTFSC  42.0
0587:  MOVLW  01
0588:  ANDWF  43,W
0589:  MOVWF  44
058A:  DECFSZ 2E,W
058B:  GOTO   58D
058C:  GOTO   58F
058D:  MOVLW  00
058E:  GOTO   590
058F:  MOVLW  01
0590:  ANDWF  44,W
0591:  MOVWF  45
0592:  MOVLW  00
0593:  BTFSS  42.2
0594:  MOVLW  01
0595:  ANDWF  45,W
0596:  XORLW  00
0597:  BTFSC  03.2
0598:  GOTO   59F
..... bit_set(durum,2);    //sistem setup modunda ise isi miktarını azaltır
0599:  BSF    42.2
..... if (isimax>0)
059A:  MOVF   23,F
059B:  BTFSC  03.2
059C:  GOTO   59F
```

..... isimax=isimax-1;
059D: MOVLW 01
059E: SUBWF 23,F
..... }
.....
..... if ((!bit_test(port_e,1)) & (bit_test(durum,0)) & (j==2) & (!bit_test(durum,2))) {
059F: MOVLW 00
05A0: BTFSS 09.1
05A1: MOVLW 01
05A2: MOVWF 43
05A3: MOVLW 00
05A4: BTFSC 42.0
05A5: MOVLW 01
05A6: ANDWF 43,W
05A7: MOVWF 44
05A8: MOVF   2E,W
05A9: SUBLW 02
05AA: BTFSC 03.2
05AB: GOTO   5AE
05AC: MOVLW 00
05AD: GOTO   5AF
05AE: MOVLW 01
05AF: ANDWF 44,W
05B0: MOVWF 45
05B1: MOVLW 00
05B2: BTFSS 42.2
05B3: MOVLW 01
05B4: ANDWF 45,W
05B5: XORLW 00
05B6: BTFSC 03.2
05B7: GOTO   5BE
..... bit_set(durum,2);   //sistem setup modunda ise nem miktarını azaltır
05B8: BSF   42.2
..... if (nemmin>0)
05B9: MOVF   24,F
05BA: BTFSC 03.2
05BB: GOTO   5BE
..... nemmin=nemmin-1;
05BC: MOVLW 01
05BD: SUBWF 24,F
..... }
.....
..... if ((!bit_test(port_e,0)) & (bit_test(durum,0)) & (j==2) & (!bit_test(durum,2))) {
05BE: MOVLW 00
05BF: BTFSS 09.0
05C0: MOVLW 01
05C1: MOVWF 43
05C2: MOVLW 00
05C3: BTFSC 42.0
05C4: MOVLW 01
05C5: ANDWF 43,W
05C6: MOVWF 44
05C7: MOVF   2E,W
05C8: SUBLW 02
05C9: BTFSC 03.2
05CA: GOTO   5CD
05CB: MOVLW 00
05CC: GOTO   5CE
05CD: MOVLW 01
05CE: ANDWF 44,W
05CF: MOVWF 45
05D0: MOVLW 00
05D1: BTFSS 42.2
05D2: MOVLW 01
05D3: ANDWF 45,W
05D4: XORLW 00
05D5: BTFSC 03.2

```
05D6:  GOTO   5DE
..... bit_set(durum,2);    //sistem setup modunda ise nem miktarını artırır
05D7:  BSF    42.2
..... if (nemmin<100)
05D8:  MOVF   24,W
05D9:  SUBLW  63
05DA:  BTFSS  03.0
05DB:  GOTO   5DE
..... nemmin=nemmin+1;
05DC:  MOVLW  01
05DD:  ADDWF  24,F
..... }
.....
..... if ((!bit_test(port_e,0)) &  (bit_test(durum,0)) & (j==3) & (!bit_test(durum,2))) {
05DE:  MOVLW  00
05DF:  BTFSS  09.0
05E0:  MOVLW  01
05E1:  MOVWF  43
05E2:  MOVLW  00
05E3:  BTFSC  42.0
05E4:  MOVLW  01
05E5:  ANDWF  43,W
05E6:  MOVWF  44
05E7:  MOVF   2E,W
05E8:  SUBLW  03
05E9:  BTFSC  03.2
05EA:  GOTO   5ED
05EB:  MOVLW  00
05EC:  GOTO   5EE
05ED:  MOVLW  01
05EE:  ANDWF  44,W
05EF:  MOVWF  45
05F0:  MOVLW  00
05F1:  BTFSS  42.2
05F2:  MOVLW  01
05F3:  ANDWF  45,W
05F4:  XORLW  00
05F5:  BTFSC  03.2
05F6:  GOTO   5FE
..... bit_set(durum,2);    //sistem setup modunda ise nem miktarını artırır
05F7:  BSF    42.2
..... if (nemmax<100)
05F8:  MOVF   25,W
05F9:  SUBLW  63
05FA:  BTFSS  03.0
05FB:  GOTO   5FE
..... nemmax=nemmax+1;
05FC:  MOVLW  01
05FD:  ADDWF  25,F
..... }
.....
..... if ((!bit_test(port_e,1)) &  (bit_test(durum,0)) & (j==3) & (!bit_test(durum,2))) {
05FE:  MOVLW  00
05FF:  BTFSS  09.1
0600:  MOVLW  01
0601:  MOVWF  43
0602:  MOVLW  00
0603:  BTFSC  42.0
0604:  MOVLW  01
0605:  ANDWF  43,W
0606:  MOVWF  44
0607:  MOVF   2E,W
0608:  SUBLW  03
0609:  BTFSC  03.2
060A:  GOTO   60D
060B:  MOVLW  00
060C:  GOTO   60E
```

```
060D:  MOVLW  01
060E:  ANDWF  44,W
060F:  MOVWF  45
0610:  MOVLW  00
0611:  BTFSS  42.2
0612:  MOVLW  01
0613:  ANDWF  45,W
0614:  XORLW  00
0615:  BTFSC  03.2
0616:  GOTO   61D
..... bit_set(durum,2);    //sistem setup modunda ise nem miktarını azaltır
0617:  BSF    42.2
..... if (nemmax>0)
0618:  MOVF   25,F
0619:  BTFSC  03.2
061A:  GOTO   61D
..... nemmax=nemmax-1;
061B:  MOVLW  01
061C:  SUBWF  25,F
..... }
.....
..... if ((bit_test(port_e,1)) & (bit_test(port_e,0)) & (bit_test(durum,2)))
061D:  MOVLW  00
061E:  BTFSC  09.1
061F:  MOVLW  01
0620:  MOVWF  43
0621:  MOVLW  00
0622:  BTFSC  09.0
0623:  MOVLW  01
0624:  ANDWF  43,W
0625:  MOVWF  44
0626:  MOVLW  00
0627:  BTFSC  42.2
0628:  MOVLW  01
0629:  ANDWF  44,W
062A:  XORLW  00
062B:  BTFSS  03.2
..... bit_clear(durum,2);
062C:  BCF    42.2
.....
.....
..... if ((bit_test(durum,0)) & (j==0)) {        // sistem setup modunda ise göster
062D:  MOVLW  00
062E:  BTFSC  42.0
062F:  MOVLW  01
0630:  MOVWF  43
0631:  MOVF   2E,F
0632:  BTFSC  03.2
0633:  GOTO   636
0634:  MOVLW  00
0635:  GOTO   637
0636:  MOVLW  01
0637:  ANDWF  43,W
0638:  XORLW  00
0639:  BTFSC  03.2
063A:  GOTO   68F
.....
..... lcd_sil();
063B:  CALL   03E
.....
.....  bit_clear(port_d,0);                        //  ******************  1.  satır  için  ISI  DEGERI  yazılır
******************
063C:  BCF    08.0
..... bit_clear(port_d,1);
063D:  BCF    08.1
..... bit_clear(port_d,2);
063E:  BCF    08.2
```

```
..... port_b=0x80;     //adr
063F:  MOVLW  80
0640:  MOVWF  06
..... enb();
0641:  CALL   004
.....
..... bit_set(port_d,1);
0642:  BSF    08.1
..... port_b=0x49;     //i
0643:  MOVLW  49
0644:  MOVWF  06
..... enb();
0645:  CALL   004
.....
..... bit_set(port_d,1);
0646:  BSF    08.1
..... port_b=0x53;     //s
0647:  MOVLW  53
0648:  MOVWF  06
..... enb();
0649:  CALL   004
.....
..... bit_set(port_d,1);
064A:  BSF    08.1
..... port_b=0x49;     //i
064B:  MOVLW  49
064C:  MOVWF  06
..... enb();
064D:  CALL   004
.....
..... bosluk();
064E:  CALL   045
.....
..... bit_set(port_d,1);
064F:  BSF    08.1
..... port_b=0x4d;     //m
0650:  MOVLW  4D
0651:  MOVWF  06
..... enb();
0652:  CALL   004
.....
..... bit_set(port_d,1);
0653:  BSF    08.1
..... port_b=0x69;     //i
0654:  MOVLW  69
0655:  MOVWF  06
..... enb();
0656:  CALL   004
.....
..... bit_set(port_d,1);
0657:  BSF    08.1
..... port_b=0x6e;     //n
0658:  MOVLW  6E
0659:  MOVWF  06
..... enb();
065A:  CALL   004
.....
..... bit_set(port_d,1);
065B:  BSF    08.1
..... port_b=0x3d;     //=
065C:  MOVLW  3D
065D:  MOVWF  06
..... enb();
065E:  CALL   004
.....
..... bit_set(port_d,1);
065F:  BSF    08.1
```

..... seg=isimin3; //isimin3
0660: MOVF   32,W
0661: MOVWF 21
..... port_b=segment1(seg);
0662: MOVF   21,W
0663: MOVWF 43
0664: CALL   24A
0665: MOVF   78,W
0666: MOVWF 06
..... enb();
0667: CALL   004
.....
..... bit_set(port_d,1);
0668: BSF   08.1
..... seg=isimin2; //isimin2
0669: MOVF   31,W
066A: MOVWF 21
..... port_b=segment1(seg);
066B: MOVF   21,W
066C: MOVWF 43
066D: CALL   24A
066E: MOVF   78,W
066F: MOVWF 06
..... enb();
0670: CALL   004
.....
..... bit_set(port_d,1);
0671: BSF   08.1
..... seg=isimin1; //isimin1
0672: MOVF   30,W
0673: MOVWF 21
..... port_b=segment1(seg);
0674: MOVF   21,W
0675: MOVWF 43
0676: CALL   24A
0677: MOVF   78,W
0678: MOVWF 06
..... enb();
0679: CALL   004
.....
..... bosluk();
067A: CALL   045
..... bosluk();
067B: CALL   045
..... bosluk();
067C: CALL   045
..... bosluk();
067D: CALL   045
.....
..... //role çekili ise bu harf yazılır
.....
..... if ((bit_test(durum,4)) | (bit_test(durum,5))) {
067E: MOVLW 00
067F: BTFSC 42.4
0680: MOVLW 01
0681: MOVWF 43
0682: MOVLW 00
0683: BTFSC 42.5
0684: MOVLW 01
0685: IORWF 43,W
0686: XORLW 00
0687: BTFSC 03.2
0688: GOTO   68E
..... bit_set(port_d,1);
0689: BSF   08.1
..... port_b=0x52; //r
068A: MOVLW 52

```
068B:  MOVWF  06
..... enb();
068C:  CALL   004
..... }
..... else
068D:  GOTO   68F
..... bosluk();
068E:  CALL   045
..... }
.....
..... if ((bit_test(durum,0)) & (j==1)) {        // sistem setup modunda ise göster
068F:  MOVLW  00
0690:  BTFSC  42.0
0691:  MOVLW  01
0692:  MOVWF  43
0693:  DECFSZ 2E,W
0694:  GOTO   696
0695:  GOTO   698
0696:  MOVLW  00
0697:  GOTO   699
0698:  MOVLW  01
0699:  ANDWF  43,W
069A:  XORLW  00
069B:  BTFSC  03.2
069C:  GOTO   6F1
.....
..... lcd_sil();
069D:  CALL   03E
.....
.....  bit_clear(port_d,0);                    // ****************** 1. satır için ISI DEGERI yazılır
******************
069E:  BCF    08.0
..... bit_clear(port_d,1);
069F:  BCF    08.1
..... bit_clear(port_d,2);
06A0:  BCF    08.2
..... port_b=0x80;     //adr
06A1:  MOVLW  80
06A2:  MOVWF  06
..... enb();
06A3:  CALL   004
.....
..... bit_set(port_d,1);
06A4:  BSF    08.1
..... port_b=0x49;          //i
06A5:  MOVLW  49
06A6:  MOVWF  06
..... enb();
06A7:  CALL   004
.....
..... bit_set(port_d,1);
06A8:  BSF    08.1
..... port_b=0x53;          //s
06A9:  MOVLW  53
06AA:  MOVWF  06
..... enb();
06AB:  CALL   004
.....
..... bit_set(port_d,1);
06AC:  BSF    08.1
..... port_b=0x49;          //i
06AD:  MOVLW  49
06AE:  MOVWF  06
..... enb();
06AF:  CALL   004
.....
..... bosluk();
```

```
06B0:  CALL   045
.....................
..................... bit_set(port_d,1);
06B1:  BSF    08.1
..................... port_b=0x4d;          //m
06B2:  MOVLW  4D
06B3:  MOVWF  06
..................... enb();
06B4:  CALL   004
.....................
..................... bit_set(port_d,1);
06B5:  BSF    08.1
..................... port_b=0x61;          //a
06B6:  MOVLW  61
06B7:  MOVWF  06
..................... enb();
06B8:  CALL   004
.....................
..................... bit_set(port_d,1);
06B9:  BSF    08.1
..................... port_b=0x78;          //x
06BA:  MOVLW  78
06BB:  MOVWF  06
..................... enb();
06BC:  CALL   004
.....................
..................... bit_set(port_d,1);
06BD:  BSF    08.1
..................... port_b=0x3d;          //=
06BE:  MOVLW  3D
06BF:  MOVWF  06
..................... enb();
06C0:  CALL   004
.....................
..................... bit_set(port_d,1);
06C1:  BSF    08.1
..................... seg=isimax3;              //isimax3
06C2:  MOVF   38,W
06C3:  MOVWF  21
..................... port_b=segment1(seg);
06C4:  MOVF   21,W
06C5:  MOVWF  43
06C6:  CALL   24A
06C7:  MOVF   78,W
06C8:  MOVWF  06
..................... enb();
06C9:  CALL   004
.....................
..................... bit_set(port_d,1);
06CA:  BSF    08.1
..................... seg=isimax2;              //isimax2
06CB:  MOVF   37,W
06CC:  MOVWF  21
..................... port_b=segment1(seg);
06CD:  MOVF   21,W
06CE:  MOVWF  43
06CF:  CALL   24A
06D0:  MOVF   78,W
06D1:  MOVWF  06
..................... enb();
06D2:  CALL   004
.....................
..................... bit_set(port_d,1);
06D3:  BSF    08.1
..................... seg=isimax1;              //isimax1
06D4:  MOVF   36,W
06D5:  MOVWF  21
```

```
..... port_b=segment1(seg);
06D6:  MOVF   21,W
06D7:  MOVWF  43
06D8:  CALL   24A
06D9:  MOVF   78,W
06DA:  MOVWF  06
..... enb();
06DB:  CALL   004
.....
..... bosluk();
06DC:  CALL   045
..... bosluk();
06DD:  CALL   045
..... bosluk();
06DE:  CALL   045
..... bosluk();
06DF:  CALL   045
.....
..... //role çekili ise bu harf yazılır
.....
..... if ((bit_test(durum,4)) | (bit_test(durum,5))) {
06E0:  MOVLW  00
06E1:  BTFSC  42.4
06E2:  MOVLW  01
06E3:  MOVWF  43
06E4:  MOVLW  00
06E5:  BTFSC  42.5
06E6:  MOVLW  01
06E7:  IORWF  43,W
06E8:  XORLW  00
06E9:  BTFSC  03.2
06EA:  GOTO   6F0
..... bit_set(port_d,1);
06EB:  BSF    08.1
..... port_b=0x52;          //r
06EC:  MOVLW  52
06ED:  MOVWF  06
..... enb();
06EE:  CALL   004
..... }
..... else
06EF:  GOTO   6F1
..... bosluk();
06F0:  CALL   045
..... }
.....
.....
..... if ((bit_test(durum,0)) & (j==2)) {        // sistem setup modunda ise göster
06F1:  MOVLW  00
06F2:  BTFSC  42.0
06F3:  MOVLW  01
06F4:  MOVWF  43
06F5:  MOVF   2E,W
06F6:  SUBLW  02
06F7:  BTFSC  03.2
06F8:  GOTO   6FB
06F9:  MOVLW  00
06FA:  GOTO   6FC
06FB:  MOVLW  01
06FC:  ANDWF  43,W
06FD:  XORLW  00
06FE:  BTFSC  03.2
06FF:  GOTO   753
.....
.....
.....                                  // ************ 2. satır için NEM DEGERI yazılır ****************
..... bit_clear(port_d,0);
```

```
0700:  BCF    08.0
..................... bit_clear(port_d,1);
0701:  BCF    08.1
..................... bit_clear(port_d,2);
0702:  BCF    08.2
..................... port_b=0xc0;      //adr
0703:  MOVLW  C0
0704:  MOVWF  06
..................... enb();
0705:  CALL   004
.....................
..................... bit_set(port_d,1);
0706:  BSF    08.1
..................... port_b=0x4e;          //N
0707:  MOVLW  4E
0708:  MOVWF  06
..................... enb();
0709:  CALL   004
.....................
..................... bit_set(port_d,1);
070A:  BSF    08.1
..................... port_b=0x45;          //E
070B:  MOVLW  45
070C:  MOVWF  06
..................... enb();
070D:  CALL   004
.....................
..................... bit_set(port_d,1);
070E:  BSF    08.1
..................... port_b=0x4d;          //M
070F:  MOVLW  4D
0710:  MOVWF  06
..................... enb();
0711:  CALL   004
.....................
..................... bosluk();
0712:  CALL   045
.....................
..................... bit_set(port_d,1);
0713:  BSF    08.1
..................... port_b=0x4d;          //m
0714:  MOVLW  4D
0715:  MOVWF  06
..................... enb();
0716:  CALL   004
.....................
..................... bit_set(port_d,1);
0717:  BSF    08.1
..................... port_b=0x69;          //i
0718:  MOVLW  69
0719:  MOVWF  06
..................... enb();
071A:  CALL   004
.....................
..................... bit_set(port_d,1);
071B:  BSF    08.1
..................... port_b=0x6e;          //n
071C:  MOVLW  6E
071D:  MOVWF  06
..................... enb();
071E:  CALL   004
.....................
..................... bit_set(port_d,1);
071F:  BSF    08.1
..................... port_b=0x3d;          //=
0720:  MOVLW  3D
0721:  MOVWF  06
```

```
..... enb();
0722:  CALL   004
.....
..... bit_set(port_d,1);
0723:  BSF    08.1
..... seg=nemmin3;              //nemmin3
0724:  MOVF   35,W
0725:  MOVWF  21
..... port_b=segment1(seg);
0726:  MOVF   21,W
0727:  MOVWF  43
0728:  CALL   24A
0729:  MOVF   78,W
072A:  MOVWF  06
..... enb();
072B:  CALL   004
.....
..... bit_set(port_d,1);
072C:  BSF    08.1
..... seg=nemmin2;              //nemmin2
072D:  MOVF   34,W
072E:  MOVWF  21
..... port_b=segment1(seg);
072F:  MOVF   21,W
0730:  MOVWF  43
0731:  CALL   24A
0732:  MOVF   78,W
0733:  MOVWF  06
..... enb();
0734:  CALL   004
.....
..... bit_set(port_d,1);
0735:  BSF    08.1
..... seg=nemmin1;              //nemmin1
0736:  MOVF   33,W
0737:  MOVWF  21
..... port_b=segment1(seg);
0738:  MOVF   21,W
0739:  MOVWF  43
073A:  CALL   24A
073B:  MOVF   78,W
073C:  MOVWF  06
..... enb();
073D:  CALL   004
.....
..... bosluk();
073E:  CALL   045
..... bosluk();
073F:  CALL   045
..... bosluk();
0740:  CALL   045
..... bosluk();
0741:  CALL   045
.....
..... //role çekili ise bu harf yazılır
.....
..... if ((bit_test(durum,6)) | (bit_test(durum,7))) {
0742:  MOVLW  00
0743:  BTFSC  42.6
0744:  MOVLW  01
0745:  MOVWF  43
0746:  MOVLW  00
0747:  BTFSC  42.7
0748:  MOVLW  01
0749:  IORWF  43,W
074A:  XORLW  00
074B:  BTFSC  03.2
```

```
074C:  GOTO   752
..... bit_set(port_d,1);
074D:  BSF    08.1
..... port_b=0x52;          //r
074E:  MOVLW  52
074F:  MOVWF  06
..... enb();
0750:  CALL   004
..... }
..... else
0751:  GOTO   753
..... bosluk();
0752:  CALL   045
..... }
.....
..... if ((bit_test(durum,0)) & (j==3)) {        // sistem setup modunda ise göster
0753:  MOVLW  00
0754:  BTFSC  42.0
0755:  MOVLW  01
0756:  MOVWF  43
0757:  MOVF   2E,W
0758:  SUBLW  03
0759:  BTFSC  03.2
075A:  GOTO   75D
075B:  MOVLW  00
075C:  GOTO   75E
075D:  MOVLW  01
075E:  ANDWF  43,W
075F:  XORLW  00
0760:  BTFSC  03.2
0761:  GOTO   7B5
.....
.....
.....                                  // ************ 2. satır için NEM DEGERI yazılır ****************
..... bit_clear(port_d,0);
0762:  BCF    08.0
..... bit_clear(port_d,1);
0763:  BCF    08.1
..... bit_clear(port_d,2);
0764:  BCF    08.2
..... port_b=0xc0;      //adr
0765:  MOVLW  C0
0766:  MOVWF  06
..... enb();
0767:  CALL   004
.....
..... bit_set(port_d,1);
0768:  BSF    08.1
..... port_b=0x4e;          //N
0769:  MOVLW  4E
076A:  MOVWF  06
..... enb();
076B:  CALL   004
.....
..... bit_set(port_d,1);
076C:  BSF    08.1
..... port_b=0x45;          //E
076D:  MOVLW  45
076E:  MOVWF  06
..... enb();
076F:  CALL   004
.....
..... bit_set(port_d,1);
0770:  BSF    08.1
..... port_b=0x4d;          //M
0771:  MOVLW  4D
0772:  MOVWF  06
```

```
..... enb();
0773:  CALL   004
.....
..... bosluk();
0774:  CALL   045
.....
..... bit_set(port_d,1);
0775:  BSF    08.1
..... port_b=0x4d;          //m
0776:  MOVLW  4D
0777:  MOVWF  06
..... enb();
0778:  CALL   004
.....
..... bit_set(port_d,1);
0779:  BSF    08.1
..... port_b=0x61;          //a
077A:  MOVLW  61
077B:  MOVWF  06
..... enb();
077C:  CALL   004
.....
..... bit_set(port_d,1);
077D:  BSF    08.1
..... port_b=0x78;          //x
077E:  MOVLW  78
077F:  MOVWF  06
..... enb();
0780:  CALL   004
.....
..... bit_set(port_d,1);
0781:  BSF    08.1
..... port_b=0x3d;          //=
0782:  MOVLW  3D
0783:  MOVWF  06
..... enb();
0784:  CALL   004
.....
..... bit_set(port_d,1);
0785:  BSF    08.1
..... seg=nemmax3;              //nemmax3
0786:  MOVF   3B,W
0787:  MOVWF  21
..... port_b=segment1(seg);
0788:  MOVF   21,W
0789:  MOVWF  43
078A:  CALL   24A
078B:  MOVF   78,W
078C:  MOVWF  06
..... enb();
078D:  CALL   004
.....
..... bit_set(port_d,1);
078E:  BSF    08.1
..... seg=nemmax2;              //nemmax2
078F:  MOVF   3A,W
0790:  MOVWF  21
..... port_b=segment1(seg);
0791:  MOVF   21,W
0792:  MOVWF  43
0793:  CALL   24A
0794:  MOVF   78,W
0795:  MOVWF  06
..... enb();
0796:  CALL   004
.....
..... bit_set(port_d,1);
```

```
0797:  BSF    08.1
..... seg=nemmax1;               //nemmax1
0798:  MOVF   39,W
0799:  MOVWF  21
..... port_b=segment1(seg);
079A:  MOVF   21,W
079B:  MOVWF  43
079C:  CALL   24A
079D:  MOVF   78,W
079E:  MOVWF  06
..... enb();
079F:  CALL   004
.....
..... bosluk();
07A0:  CALL   045
..... bosluk();
07A1:  CALL   045
..... bosluk();
07A2:  CALL   045
..... bosluk();
07A3:  CALL   045
.....
..... //role çekili ise bu harf yazılır
.....
..... if ((bit_test(durum,6)) | (bit_test(durum,7))) {
07A4:  MOVLW  00
07A5:  BTFSC  42.6
07A6:  MOVLW  01
07A7:  MOVWF  43
07A8:  MOVLW  00
07A9:  BTFSC  42.7
07AA:  MOVLW  01
07AB:  IORWF  43,W
07AC:  XORLW  00
07AD:  BTFSC  03.2
07AE:  GOTO   7B4
..... bit_set(port_d,1);
07AF:  BSF    08.1
..... port_b=0x52;          //r
07B0:  MOVLW  52
07B1:  MOVWF  06
..... enb();
07B2:  CALL   004
..... }
..... else
07B3:  GOTO   7B5
..... bosluk();
07B4:  CALL   045
..... }
.....
..... if ((!bit_test(port_e,1)) & (!bit_test(durum,0)) & (!bit_test(port_a,4))) {      // sistemin enerjisi kapatılır
07B5:  MOVLW  00
07B6:  BTFSS  09.1
07B7:  MOVLW  01
07B8:  MOVWF  43
07B9:  MOVLW  00
07BA:  BTFSS  42.0
07BB:  MOVLW  01
07BC:  ANDWF  43,W
07BD:  MOVWF  44
07BE:  MOVLW  00
07BF:  BTFSS  05.4
07C0:  MOVLW  01
07C1:  ANDWF  44,W
07C2:  XORLW  00
07C3:  BTFSC  03.2
07C4:  GOTO   7F0
```

```
..... kapt:
.....
..... bit_clear(port_d,7);          // role kontrol portları kapatılır.
07C5:  BCF    08.7
..... bit_clear(port_d,6);          // + ve set tuslarına basılı ise ve sistem set modunda değilse aktif hale gelir.
07C6:  BCF    08.6
..... bit_clear(port_d,5);
07C7:  BCF    08.5
..... bit_clear(port_d,4);
07C8:  BCF    08.4
..... bit_clear(port_d,3);          // enerji kontrol portları kapatılır.
07C9:  BCF    08.3
.....
..... lcd_sil();
07CA:  CALL   03E
.....
.....  bit_clear(port_d,0);                    // ****************** 1. satır için ISI DEGERI yazılır
******************
07CB:  BCF    08.0
..... bit_clear(port_d,1);
07CC:  BCF    08.1
..... bit_clear(port_d,2);
07CD:  BCF    08.2
..... port_b=0x80;     //adr
07CE:  MOVLW  80
07CF:  MOVWF  06
..... enb();
07D0:  CALL   004
.....
..... for (i=0;i<=17;++i)
07D1:  CLRF   2D
07D2:  MOVF   2D,W
07D3:  SUBLW  11
07D4:  BTFSS  03.0
07D5:  GOTO   7D9
..... bosluk();
07D6:  CALL   045
07D7:  INCF   2D,F
07D8:  GOTO   7D2
.....
..... bit_clear(port_d,0);
07D9:  BCF    08.0
..... bit_clear(port_d,1);
07DA:  BCF    08.1
..... bit_clear(port_d,2);
07DB:  BCF    08.2
..... port_b=0xc0;     //adr
07DC:  MOVLW  C0
07DD:  MOVWF  06
..... enb();
07DE:  CALL   004
.....
..... for (i=0;i<=17;++i)
07DF:  CLRF   2D
07E0:  MOVF   2D,W
07E1:  SUBLW  11
07E2:  BTFSS  03.0
07E3:  GOTO   7E7
..... bosluk();
07E4:  CALL   045
07E5:  INCF   2D,F
07E6:  GOTO   7E0
.....
..... sleep();
07E7:  SLEEP
..... delay_ms(2000);
07E8:  MOVLW  08
```

```
07E9:  MOVWF  43
07EA:  MOVLW  FA
07EB:  MOVWF  44
07EC:  CALL   011
07ED:  DECFSZ 43,F
07EE:  GOTO   7EA
..... goto kapt;
07EF:  GOTO   7C5
..... }
.....
..... delay_ms(5);  // 300 ms olmalı toplam cevirim 2,5 msn olsun diye saniyede 2 defa ekran tazelenir.
07F0:  MOVLW  05
07F1:  MOVWF  44
07F2:  CALL   011
..... goto basla;
07F3:  GOTO   2F4
.....
..... }
.....
.....
.....
07F4:  SLEEP

Configuration Fuses:
   Word  1: 3F7A   HS NOWDT NOPUT NOPROTECT BROWNOUT NOLVP NOCPD NOWRT NODEBUG
```

ÖZGEÇMİŞ

1958 yılında Malatya' da doğdu. İlk ve orta öğrenimini Malatya' da tamamladı. Ocak – 1985' de O.D.T.Ü. Mühendislik Fakültesi Elektrik Elektronik Mühendisliğinden mezun oldu. Özel bir şirkette hidrolik santrallerin aydınlatma ve topraklama projelerinin hazırlanmasında çalıştı. Tekel' e ait bir sigara fabrikasında Elektrik Elektronik Mühendisi olarak görev yaptı. YÖK – Dünya Bankası Endüstriyel Eğitim Projesi kapsamında açılan sınavı kazanarak kasım – 1990' da 9 ay süreyle A.B.D.' ye giderek Ferris State University ve New River Community College' de kendi alanındaki çalışmalara misafir öğretim görevlisi olarak katıldı. Yurt dışı dönüşü 1.5 yıl süreyle Fırat Üniversitesi Elazığ MYO' da görev yaptı.

Mart – 1993' de kendi isteğiyle naklen İnönü Üniversitesi Malatya MYO' na atanarak Sayısal Elektronik, Devre Analizi, Analog Elektronik, Mikroişlemciler, Optoelektronik, PLC, Güç Elektroniği, Elektrik Makinaları, Enstrümantasyon ve Proses Kontrol derslerini verdi. 2000 ve 2004 yılları arasında 4 yıl süre ile Mühendislik Fakültesi Elektrik Elektronik Mühendisliği Bölümünde görevlendirilerek bu fakültede ise Fiziksel Elektronik, Elektrik Elektronik Bilgisi, Elektrik Tesisatı ve Aydınlatma, Teknik İngilizce derslerini verdi. 2004 yılından itibaren de Malatya MYO' da öğretim görevlisi olarak görev yapmakta olup, PIC ve mikrodenetleyici kontrol konularında çalışmalar yapmaktadır.