YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

EV OTOMASYONUNUN BİLGİSAYAR ARAYÜZÜ İLE GERÇEKLEŞTİRİLMESİ

Elektrik Müh. Levent BİRGÜL

FBE Elektrik Mühendisliği Anabilim Dalı Kontrol ve Otomasyon Programında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı: Prof. Dr. Galip CANSEVER (YTU)

İSTANBUL – 2007

İÇİNDEKİLER

SİMGE LİSTESİ

x	Ortamdan alınan gerçek değer
y	Ortamdan alınan gerçek değer
μ	Üyelik fonksiyonu
w_i	Kuralın çıkış üzerindeki etkisi
r	Fonksiyonun derecesi
z	Çıkış

KISALTMA LİSTESİ

AC	Alternating Current
ADC	Analog-Digital Converter
ANSI	American National Standarts Institue
CPU	Central Processing Unit
D	Derivative
DAC	Digital-Analog Converter
DC	Direct Current
DMA	Direct Memory Access
EEprom	Electronically Erasable Pragrammable Read-Only Memory
FPU	Floating Point Unit
GHz	Giga Hertz
I	Integral
I2C	Inter-Integrated Circuit
IP	Internet Protocol
IRQ	Interrupt Request
ISM	Industrial, Scientific and Medical
KB	Kilo Byte
kSPS	kilo Sample Per Second
LDR	Light Dependent Resistor
MB	Mega Byte
MCS-51	8051 Microcontroller Series
MHz	Mega Hertz
MIPS	Million Instruction Per Second
NB	Negatif Büyük
NK	Negatif Küçük
P	Propotional
PB	Pozitif Büyük
PCB	Printed Ciruit Board
PK	Pozitif Küçük
PLL	Phase Lock Loop
PWM	Pulse Width Modulation
RAM	Random Access Memory
ROM	Read-Only Memory
RS232	Recommended Standart 232 (EIA232)
RS485	Recommended Standart 485(EIA485)
S	Sıfır
SAR	Successive Approximation Register
SPI	Serial Peripheral Interface
TIC	Time Internal Counter
TV	Televizyon
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
V	Volt
W	Watt

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

Bu tez çalışmasında günümüzde gelişmekte olan teknolojinin günlük hayatta insanların yaşamlarının büyük kısmını geçirdikleri ev, büro gibi yapılarda kullanılan ve her geçen gün gelişim gösteren bina otomasyonu hakkında çalışılmıştır.

Bu tezde yardımlarını ve desteklerini esirgemeyen aileme, arkadaşlarıma, Tüm Elektronik Ar-Ge yöneticisi Cengiz Bektaş'a, Tüm Elektronik Proje Yöneticileri Erdem Pamuk ve Murat Ünal'a, Arrow'dan Selim Dilmaç'a ve Yıldız Teknik Üniversitesi Elektrik-Elektronik Fakültesi Dekanı Prof. Dr. Galip Cansever'e teşekürü borç bilirim.

İstanbul-2007

Levent BİRGÜL

ÖZET

Otomasyon sistemleri günümüzde her alanda kullanılmaktadır. Bu tez çalışmasında otomasyonun evlerdeki uygulaması üzerinde çalışılmıştır.

Gerçekleştirilen sistemde evin içerisindeki odalara oda panosu evin girişine ise ana pano olarak bilgisayar yerleştirilmiştir. Oda panolarının görevi odalardaki sıcaklık ve aydınlatma kontrolünü yapmaktır. Aydınlatmada odanın içerisindeki ışık bilgisi LDR'ler yardımıyla okunur. Çıkışta ise ışık grupları kademeli olarak P kontrol algoritmasına göre sürülmektedir. Bu şekilde odanın her noktasında aynı aydınlık seviyesi ekonomik olarak sağlanmış olmaktadır.

Odalardaki sıcaklık kontrolünde ise TMP35'ler ile alınan oda sıcaklık bilgisi bulanık mantık algoritması ile işlenerek ısıtıcı veya soğutucunun kademeli olarak çalıştırılması sağlanmaktadır.

Oda panosunda bulunan mikrodenetleyeci ile odanın pencerelerinde bulunan Rit role yardımıyla pencere durumu öğrenilmekte ve kullanıcı tarafından aktive edilmemiş bir odadan gelen izinsiz pencere konumunun değişikliğinde oda modülü üzerinde bulunan alarm çıkışı aktif edilmektedir.

Ana pano olarak kullanılan bilgisayar üzerine Delphi programı ile yazılmış olan yazılım yardımıyla oda modülerinden gelen oda sıcaklık bilgisi, oda sıcaklık set değer bilgisi ve oda aydınlık seviye bilgisi gösterilmektedir. Ayrıca odaların sıcaklık zaman grafiği bilgisayar yazılımında görülebilmektedir.

Anahtar Kelimeler: Bilgisayar Destekli Ev Otomasyonu, Mikrodenetleyici, ADuC841, Sıcaklık kontrolü, Aydınlık, Güvenlik.

ABSTRACT

Automation systems are being used in all fields that humans are living. In this thesis, Building automation subject is chosen to studied on automation subjects.

In designed building automation system, modules which are designed with microcontrollers are used in rooms and a computer software is used for main control panel. The modules which are used in rooms, controls temperature and lightening of room. Lightening knowledge is taken by LDRs (Light Dependent Resistor). Lightening sources are driven by P control algorithm. By gradually lightening source groups, the same light level obtained and get a good return on financial.

In modules, TMP35 is used for sensing temperature of room. This temperature knowledge put in fuzzy algorithm in microcontroller software. According to this temperature knowledge fuzzy logic algorithm gives the level of heater or cooler output.

By the help of rit relay, windows position is taken. If a window's position is change in room which is not activated by user, alarm output of module will be active.

On computer software which is written in Delphi, shows temperatures of rooms, temperature wihch is set by user, lightening per cent which is set by user. Besides room temperature according time graph is drawn by software.

Key Words: Computer Based Building Automation, Microcontroller, ADuC841, Temperature Control, Lightening, Security.

1. GİRİŞ

Günümüzde binaların hacimleriyle birlikte sorunları da büyümektedir. Bu sorunların başında binanın ısıtılması, soğutulması, havalandırılması, aydınlatması, yangın ve güvenlik önlemlerinin alınması gelmektedir. Ayrıca bu sistemlerin uygun zamanda uygun miktarlarda ve birbirleriyle koordinasyonlu biçimde çalıştırılması diğer bir husustur. Sözü edilen sistemlerin otomatik kontrol olmadan sadece insan kontrolü ile belirtilen biçimde çalıştırılmasının zorluğu açıktır. Ayrıca insan kontrolü bu sistemler için pek çok sakınca doğurmaktadır. Sistemlerin yeterli düzende çalıştırılamaması sonucu enerji sarfiyatının artması ve istenen koşulların tam olarak sağlanamaması bunların başında gelmektedir.

Uzun yıllar binaların yukarıda sayılan özelliklerinin kontrolü ya klasik yöntemlerle veya bizzat insanlar tarafından yapılmıştır. Ancak binaların gün geçtikçe büyümesi, karmaşıklaşması, modernleşmesi ve ihtiyaçlarının çoğalması neticesinde tüm bu enerji harcayan sistemlerin klasik yöntemlerle veya el ile kontrolü çok zor ve masraflı hale gelmiştir. Böylece bina otomasyonu kavramı daha net hale gelmiş ve sistemlerin sadece kontrolü için değil düşük işletme maliyetleri için de gereklilik halini almıştır.Ayrıca günümüzde bina otomasyonu sadece büyük ve karmaşık binalar için değil artık evlerin otomasyonu için de kullanılır hale gelmiştir.

Bilgisayar sistemlerinin hızla gelişmesi ve her iş alanına girmesi bu alanda da kendisini göstermiştir. Klasik kontrol panoları ve kontrol cihazlarından sonra tamamen programlanabilir kontrolörler geliştirilmiş; bunlardan gerekli sayıda olanları binanın gerekli yerlerine yerleştirilerek ve istendiğinde birbirlerine ve bir merkezi bilgisayara bağlanarak tüm kontrollerin gözlemlerin tek bir merkezde toplanması mümkün olmuştur. Otomatik kontrol programları ile cihazlar en uygun biçimde çalıştırılmakta ve büyük miktarlarda enerji tasarrufu sağlanmaktadır. Bina Yönetim Sistemi adı verilen bu sistemlerin nemi ülkemizde de her geçen gün daha iyi anlaşılmakta ve yaygınlaşmaktadır.

Bina yönetim sistemleri adı verilen bu sistemler uzman bir kişinin kontrolü olmadan da çalışabilme kabiliyetlerini her geçen arttırmaktadırlar. Kabiliyetlerinin artması ile bina otomasyon sistemleri her kişinin kendi evinde bulunan sistemler haline gelmektedirler. Bu sistemler evin herhangi bir yerinde kullanıcının gözünün önünde olmayan ancak kullanıcıya ev içerisindeki hayatını kolaylaştıran sistemler haline gelmiştir.

2. EV OTOMASYONU

2.1 Ev Otomasyon Sisteminin Tarihçesi

Bina otomasyonu 1950'lilerin başında ortaya çıkmış, günümüz teknolojisi ile hayallerimizin ötesine geçmiştir. İlk yapılan sistemlerde bilgi ve kontrol noktaları ana kontrol paneline ayrı ayrı kablolarla taşınır, sisteme ana panodan ve operatör panosundan işlemler gerçekleştirilir konumdaydı. Bu şekilde sistemler kontrol edilebiliyordu ancak arızaların giderilmesi ve sistemin kurulumunda zorluklar yaşanıyordu.

1970'lerde bilginin seri olarak taşınabilmesi ve elektronikteki büyük gelişmeler, bina otomasyonun da önünü açmıştır. Birkaç kablo üzerinden yapının değişik noktaları ile bilgiler alınıp, kontrol sağlanmıştı. Bu sayede montajın ve arıza tespitinin kolaylaşmasının yanında sistemler daha üstün özelliklere kavuştular. Basınç, sıcaklık gibi parametreleri ölçerek ana merkeze gönderme gibi işlemleri yapabilir konuma geldiler.

1980lerin sonlarında bilgisayar teknolojisiyle birlikte elektronikte de yaşanan gelişmeler ile bina otomasyonunda bilgi toplayan üniteler artık bilgi toplama değil kontrol noktası haline geldiler. Artık, bulundukları noktada ölçüm yapabilir ve bu ölçüm için yapılması gereken bir işlem varsa onu da gerçekleştirebilir konuma gelmiştir.

2.2 Ev Otomasyonunda Kontrol Edilebilen Parametreler

2.2.1 Sıcaklık-İklimlendirme

Ev veya büro içindeki sıcaklığı optimum tutulması ile temiz hava akışının kontrolünü içerir. Otomatik sıcaklık kontrolü gece ve gündüz sıcaklık ayarları seçenekleri ile optimum sıcaklığı muhafaza edebilmelidir. Kendisini dışarıdaki sıcaklığa göre otomatik olarak ayarlayabilmelidir. Böylece sürekli bir konfor ortamı sağlanırken, aynı zamanda maddi olarak da kazanç sağlayacaktır.

2.2.2 Aydınlatma

Karanlık bir odaya birinin girmesi durumunda açılan, uzun süre odada kimse olmayınca kapanmaya programlı ışıklar bugün her yerde karşımıza çıkmaktadır. Hareket dedektörlü bu sistemlerin yanında kapısı açıldığında veya arabayla eve yaklaşıldığında otomatik olarak açılan ışıklandırma sistemleri, otomatik ışık düzenleri farklı durumlar (çalışma, dinleme, eğlence, uyku) için seçeneklere sahip olan sistemler, güneş ışığının açısına, odanın içerisindeki aydınlık seviyelerine göre ayarlanabilen sistemler günümüzde mevcuttur. Bunların hepsi birer ev otomasyonun aydınlatma alt grubunda yer alır.

2.2.3 Güvenlik

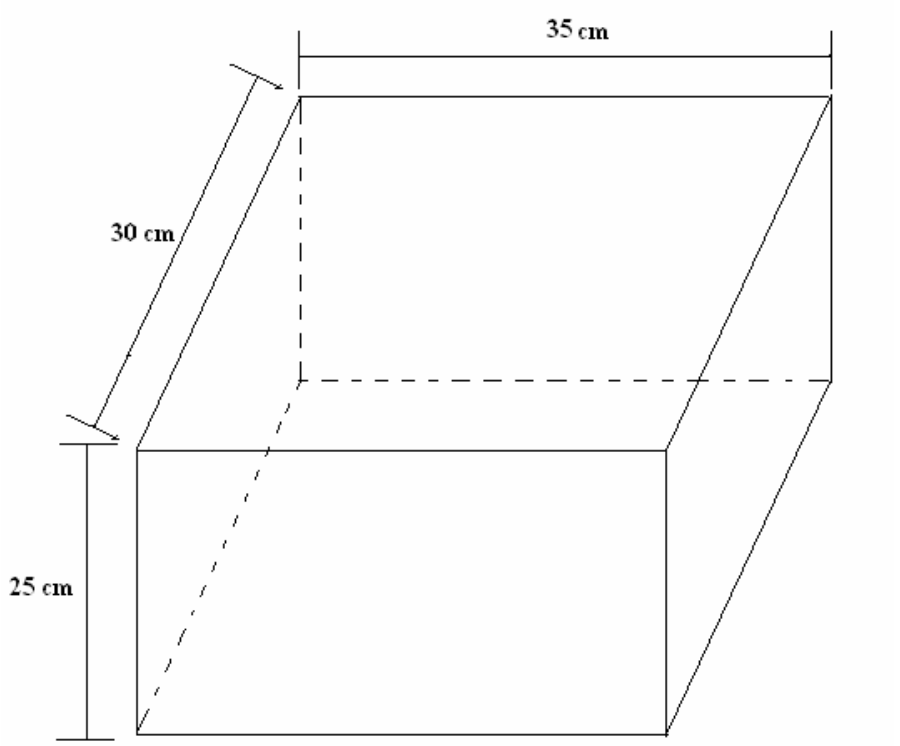
Sistemin yangın, hırsızlık, gaz kaçağı, donma noktasındaki sıcaklıklar ve su kesintisi gibi durumlarda kişileri haberdar etmesini içerir. Klasik güvenlik önlemlerinin yanında arttırılmış önlemler de söz konusu olabilir. Aktif olarak dışarıdan içeriye müdahaleyi durduran bir güvenlik sistemi buna örnektir. Bunun yanında kişiler sistemlerini pasif güvenlik sağlayacak şekilde programlayabilirler. Evde yokken sistemin programlanması sonucu hırsızlar üzerinde caydırıcılık amaçlı önlemlerdir. Işıkların, televizyonun ve müzik setinin çeşitli zaman aralıkları ile açılıp kapanmalarını sağlayarak, ev boşken bile evde biri varmış izlenimi yaratır.

3. UYGULAMASI GERÇEKLEŞTİRİLEN EV OTOMASYON SİSTEMİ

3.1 Uygulama Hakkında Genel Bilgi

Bilgisayar ve mikroişlemci üzerinde gerçekleştirilen yazılımın test edilmesi için evin bir odasına denk gelen küçük bir oda maketi üzerinde çalışma yapılmıştır. Bu oda maketi üzerinde gerçekleştirilen çalışma istenildiğinde, sadece oda sayısı arttırılarak bir ev şeklinde de çalışma gerçekleştirilebilir.

Üzerinde çalışma yapılan oda maketinde iki ledden oluşan üç led aydınlatma grubu, bir ısıtıcı, bir soğutucu, güvenlik amaçlı bir manyetik röle bulunmaktadır. Oda maketinde yan yüzeylerden biri pencere olarak düşünülmüştür.



Şekil 3.1 Odanın ölçeklendirilmiş çizimi

Şekil 3.2 Oda modelinin resmi

Ev otomasyon sisteminin model üzerinden gerçekleştirilmesinde bir adet ana bilgisayar yazılımı ve buna bağlı olarak çalışan oda panolarının kontrolü esas alınmıştır. Sistemin çalışmasının genel prensibi odaya ait olan panonun bilgisayar yazılımı ile haberleşerek kontrol sağlamasına dayanır. Oda panosundaki kontrol kartlarında aydınlatma, sıcaklık ve güvenlik kontrolü yapılmaktadır. Bilgisayar yazılımında kullanıcının oda içerisinden alınan anlık değerlere ve sıcaklık- zaman grafiğine ulaşması hedeflenmiştir. Burada kullanıcı tanıma, şifreleme, izleme işlemleri yapılmaktadır.

3.2 Tasarım Esnasında Seçilen MCU

8051 ailesi, INTEL firması tarafından 1980'lerin başında piyasaya sunulan dünyanın en popüler 8-bit mikrokontrolör ailesidir. INTEL 'den sonra, bu MCU (Micro Controller Unit) ailesi ile uyumlu olarak, başta PHILIPS, SIEMENS, ATMEL, DALLAS, OKI, HYUNDAI/HYNIX, WINBOND olmak üzere pek çok üretici firma türev işlemciler üretmiştir. Bunlardan başka kendi özgün mikrokontrolör ailelerini üreten ST, TEXAS INS. gibi çeşitli büyük üreticiler bile 8051 uyumlu MCU lar geliştirmiş ve pazara sunmuştur.

Internet'de yapılan bir araştırmada 55'in üzerinde 8051 üreticisi belirlenmiştir. Bu firmalara ait bir liste bölüm sonunda yer almaktadır. KEIL, IAR, NOHAU, TASKING vb başta olmak üzere çok miktarda diğer firma ise geniş bir donanım ve yazılım geliştirme araçları desteği sunmuştur. Bunun sonucu olarak 8051 ailesi, 1980'lerden bugüne bir endüstri standardı olmuştur.

8051 ailesi bazen MCS–51 ailesi olarak da anılmaktadır. 8051 ve MCS–51 tanımlamaları aynı aileyi belirtmek için kullanılır. Ayrıca "8051" bu ailenin ilk üyelerinden biri olan "Mask ROM" lu modelin de adıdır.

Bugün için çeşitli 8-bitlik mikrokontrolör aileleri arasında 8051 ailesi, gelişmiş türev ürünleriyle beraber yaklaşık %40 gibi bir piyasa payına sahiptir. 8051 ailesinin başlıca özellikleri aşağıda maddeler halinde verilmiştir.

1 - Geniş Yelpaze: Pek çok üretici firma, orijinal 8051'e çeşitli ek özellikler katarak türev ürünler geliştirmiştir. Her bir üretici firmanın onlarca, hatta bazılarının elliden fazla türev ürünü olduğu dikkate alınırsa ne kadar geniş bir aileden söz edildiği daha rahat anlaşılabilir. Bütün bu ürünler için çeşitli yazılım ve uygulama geliştirme donanımları üreten firmaların da katkılarıyla 8051 bir endüstri standardı haline gelmiştir. Yeryüzünde "Industry Standard" tanımlamasına sahip tek 8-bitlik mikrodenetleyici ailesidir.

2 - Uyumluluk: Çok değişik 8051 türev ürünler bulunmasına rağmen komut seti ve mimari yapı olarak bütün ürünler uyumludur (code compatible). Diğer mikrodenetleyici aileleri, 8051'in sunmuş olduklarını farklı ve uyumlu olmayan işlemcilerle (genellikle tek üretici firma kaynaklı olarak) sağlayabilmektedirler. Bu uyumluluk, kolaylık ve esneklik, program geliştirme araçlarında, eğitiminde ve yazılım desteğinde de bulunmaktadır.

3 - Hız ve Güç:8051 çekirdek mimarisi kontrol uygulamaları için gayet uygun olup hızlı ve güçlüdür. Piyasaya ilk sunuldukları tarihte 12MHz lik modelleri yaklaşık olarak 1 MIPS de (Mega Instruction Per Second) çalışıyor iken günümüzde 24 MIPS, 50 MIPS hatta 100 MIPS'lik hızlara sahip olan türev işlemcilere sahiptir. Bu performans ile 1 makine çevrimlik bir komutu 40ns veya 20ns, hatta 10 ns. gibi bir sürede icra eder. 8-Bitlik işlemci aileleri arasında bu hıza sahip genel bir işlemci ailesi bulunmamaktadır.

4 - Popülerlik: 8051 kullanıcıları için birçok kitap, teknik dokümanlar, yazılım ve donanım gereçleri, pek çok İnternet Web Sayfası mevcuttur. Ürün kolay bir şekilde bulunmakta ve yaygın bir şekilde desteklenmektedir. Eğitim notlarının sonunda 8051 MCU üreticileri, 8051

ailesi için geliştirme araçları donanım ve yazılımları üreten firmalara ait irtibat bilgileri ayrıca çeşitli internet web sitesi adresleri yer almaktadır.

5 - Sürekli Geliştirilme: 1980'lerden bugüne silikon ve tasarım olarak sürekli geliştirilen 8051'lerin hızları, işlem güçleri, on-chip çevre birimleri sayısı ve çeşitliliği artmıştır. Örneğin Analog Devices firması tarafından üretilen bir ürün (ADUC845) Standart 8051 mimarisinde yer alan özelliklerin yanı sıra:

-10 kanal 24 bit rezolüsyona sahip 10 kanal ADC,

-Programlanabilir Gain Amplifier,

-12 bit voltage output DAC,

-Dual PWM çıkışları,

-Voltage reference,

-Current Source,

-Temperature sensor

-Power supply monitör,

-Power-on reset,

-PLL,

-62KB on-chip Flash ROM (In system & In Application Programmable),

-On-chip download / debug interface,

-On-chip 256 Byte + 2KB data RAM,

-4KB data EEprom,

-16MB external data RAM address space,

-UART, SPI, I2C 3 kanal serial interface,

-3 kanal 16-bitlik timer/counter,

-Timer Interval Counter (Real Time Clock),

-Watchdog timer,

-Baud rate generator timer,

-11 Interrupt Vector

gibi özelliklere de sahiptir.

3.2.1 8051/8052 Tarihçesi

Intel firması 1968 yılında hafıza entegre devreleri (Integrated Circuit / tümdevre) yapmak üzere kuruldu. Üretecekleri bir hesap makinesi için CPU entegresi isteyen, hesap makinesi üreten bir firmanın talebi ve üretecekleri bir terminal için özel bir tümdevre isteyen, diğer bir firma Datapoint Corporation'ın isteklerini karşılamak için, Intel firması 4004 (1971) ve 8008 (1972) CPU'larını tasarladı.

Mikroişlemciler ve mikrobilgisayarların sınıflandırılmasında en temel ölçü, mikroişlemcinin entegre üzerinde (On-Chip) işlem yaptığı en uzun verinin bit sayısı, yani kelime uzunluğudur (word length). 4-bitlik işlemci olan 4004 ve 8-bitlik işlemci olan 8008'den başlayarak, mikroişlemciler ve mikrobilgisayarlar için 4-bit, 8-bit, 16-bit, 32-bit, 64-bit gibi veri uzunluk standartları doğmuştur.

Intel, bu ilk ürünlerini başlangıçta sadece o müşterileri için hazırlamıştı. Fakat ilk siparişleri veren firmalar ürünleri kullanmamaya karar verince, piyasaya tanıtım yapıldı. Ciddi bir satış potansiyeli olduğu görülmesi üzerine, aynı zamanda 8008'in 16K'lık hafıza limitini aşmak amacıyla, Intel firması (1974) yılında genel-amaçlı 8080 CPU'sunu üretti. Şaşırtıcı bir şekilde yüksek ilgi gören mikroişlemciler hızla yaygınlaşmaya başladı ve kısa süre içinde 8080, 8-bit mikroişlemcilerde endüstri standardı oldu.

Diğer yarı iletken üreticileri de kendi ürünlerini piyasaya sürdüler, ancak bunların hepsi başaralı olamadı. Başarılı olanlar arasında MOS Technologies'in 6502'si (Apple II bilgisayarlarda kullanıldı), Motorola 6800 ve Zilog Z80 anılabilir. Intel, bir kaç yıl sonra gelişmiş 8-bitlik 8080 işlemcisi olarak adlandırılabilecek, CPU ve çeşitli çevre birimlerinin tek çip üzerinde gerçekleştirildiği, halen yaygın kullanımda olan güçlü bir komut setine sahip, günümüz modern mikrokontrolörlerinin atası 8051'i satışa sundu.

Intel 1978 yılında ilk 16-bit mikroişlemcisi olan 8086'yı üretti. 8086 daha önceki 8080/8085 ürününe bazı yönlerden benzemesine rağmen, iki işlemci ailesi birbiri ile uyumlu değildi. Bir yıl sonra 1979'da üretilen, 8086'nın 8–bit veri yoluna sahip sürümü olan 8088, 1981 yılında üretilen IBM PC mikrobilgisayarının ilk işlemcisi olmuştur. Kısa sürede endüstrinin 16-bit mikroişlemci standardı olan 8086/8088, günümüze kadar uzanan pek çok değişik ürünüyle, x86 ailesi diye adlandırılan çekirdeği(core) oldu.

8086

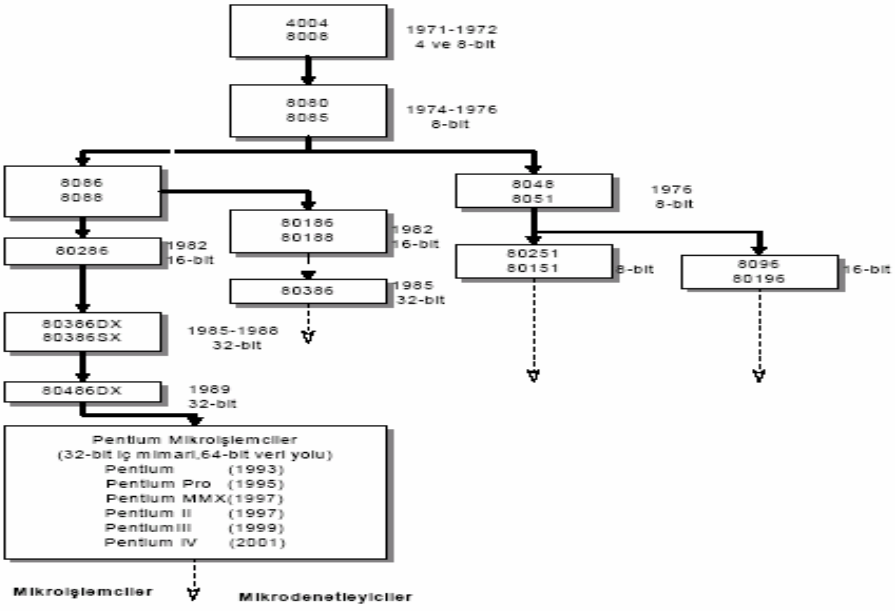
→ 80286 → 80386 → 80486 → PENTIUM →… … …

8088

Şekil 3.3 x86 ailesinin çekirdeği

Daha sonraki yıllarda x86 ailesinin diğer ürünleri,80286, 80186 ve 80188 üretildi. 80186 işlemcisi 8086'nın tümdevre-üzere çeşitli çevre birimlerine sahip olan sürümüdür. 80188 işlemcisi ise, 8-bit veri yoluna sahip bir 80186 işlemcisidir. Tasarımlarda fazla çevre birimi istemeyen 80186/80188 işlemcilerinin, genelde değişmez bir programla, kontrol uygulamaları içinde yer alarak mikrodenetleyici gibi kullanılmaları amaçlanmıştır. Bu iki işlemci yaygın olarak kullanılmıştır.

Uygulama çeşitlerine göre Intel mikroişlemcilerinin sınıflara ayrılması, 80186/80188 ve 8048/8051 işlemcilerden sonra başlamıştır. Genel olarak Intel mikroişlemcileri bugün tekrar programlanabilir mikroişlemciler (Genel Amaçlı İşlemciler) ve mikrodenetleyiciler (Özel Amaçlı İşlemciler) olarak ikiye ayrılır. Intel mikroişlemcileri ve mikrodenetleyicilerinin gelişimi aşağıdaki Şekil 3.4 ve Çizelge 3.1 de verilmiştir.



Şekil 3.4 8051 aile ağacı

8086/8088 işlemcilerinin 1 megabyte hafıza ile sınırlı adres alanı,1980'lerin ortalarına doğru birçok uygulama için ciddi bir problem olmaya başlamıştı. Bu yüzden Intel x86 çekirdeğinin bir üst uyumlu sürümü olan 80286 işlemcisini üretti. Bu işlemci, 16 megabyte'lık adres

alanıyla beraber 8086/8088 komut kümesine sahipti. 80286, IBM PC/AT ve orta model PS/2 bilgisayarlarında kullanıldı ve daha önceki 8088 gibi büyük bir başarı kazandı.

Intel için bir sonraki adım, 1985 yılında üretilen, bir entegre devre-üzerinde gerçek 32- bit CPU olan 80386DX oldu. 80286 gibi bu mikroişlemci de çok yaygın olarak kullanıldı. 1988 yılında, harici 16-bit veri yoluna sahip 80386SX işlemcisi üretildi.

80486, 80386'nın bir üst uyumlu modeliydi. Bütün 80386 programları, 80486 makinelerinde bir değişiklik yapılmadan çalışabilecekti. Bu iki işlemci arasındaki temel fark, 80486'nın 80386'ın özelliklerine ek olarak, yardımcı işlemcisi olan bir kayan-nokta birimine (Floating Point Unit-FPU), 8 kilobyte ön hafıza (cache) ve bir hafıza yönetim birimine tümdevre-üzerinde sahip olmasıdır. Ayrıca 80486, 80386'dan çok daha hızlıdır.

Çizelge 3.1 İntel işlemcilerin yıllara göre üretimleri ve özellikleri [18]

İşlemci	Yıl	Saklayıcı/Veri YoluGenişliği	Adres Alanı	Önemli Özellikler
4004	1971	4\4	640 byte	İlk mikroişlemci,2300 transistör,10 mikron
8008	1972	8\8	16K	İlk 8-bit işlemci,108Khz
8080	1974	8\8	64K	İlk genel amaçlı CPU, 6 mikron,6000 transistör
8085	1976	8\8	64K	Gelişmiş 8080,6200 transistör
8086	1978	16\16	1M	İlk 16-bit CPU, 5-10MHz,29000 transistör,3 mikron
8088	1979	16\8	1M	1981'de üretilen IBM PC'deki ilk işlemci,8-bit veri yolu 8086
80186	1982	16\16	1M	8086+I/0
80188	1982	16\8	1M	8088+ I/0
80286	1982	16\16	16M/1G	134000 transistör, 1.5 mikron,IBM PC/AT'nin ilk işlemcisi
80386DX	1985	32\32	4G/64T	Intel'in ilk 32-bit işlmecisi,275000 transistör,1 mikron

80386SX	1988	32\16	4G/64T	80286 yoluna sahip 80386
80486DX	1989	32\32	4G/64T	80386+FPU+cache,1.2 milyon transistör,0.8 mikron
Pentium	1993	32\64	4G/64T	3.1 milyon transistör,0,8 mikron,60-200 MHz, superscalar mimari
Pentium Pro	1995	32\64	64G/64T	5.5 milyon transistör,0.32 mikron, tümdevre üzeri L2 cache, P6 mimarisi: çoklu dallanma tahmin, veri akışı analizi ve tahmini yürütme
Pentium MMX	1997	32\64	4G/64T	4.5 milyon transistör, multi-medya ekleri
Pentium II	1997	32\64	64G/64T	7.5 milyon transistör, 0.25 mikron, 300-450 MHz, MMX+ Pro Teknolojisi
Pentium III	1999	32\64	64G/64T	9.5 milyon transistör, 0.18 mikron,450-500 MHz, 3D grafikler ve daha fazla multi-medya desteği

1993 yılında piyasaya sürülen Pentium, temel mimari olarak çok farklı bir mikroişlemci olmayıp, Intel' in her 2-3 yılda bir ürettiği yeni bir x86 işlemcisidir. Bu yapı IA-32 (Intel Architecture) olarak belirtilen 386/486 ile başlayan 32-bit mimarinin bir uzantısıdır.

3.2.2 Genel 8052 Mikrodenetleyicisinin Özellikleri

8051 mikrodenetleyici ailesinin başlıca özellikleri aşağıda verilmiştir;

• Kontrol uygulamaları için optimize edilmiş 8 bitlik CPU

• Genişletilmiş Boolean işleme komutları (tek bitlik lojik komutlar)

• On-chip program hafızası (Program ROM)

• On-chip veri hafızası (Data RAM)

• 4 adet 8 bitlik I/O portu

• Çift yönlü kullanılabilen ve tek tek adreslenebilen I/O pinleri

• 3 kanal 16 bitlik Timer/Counter

• Full Duplex UART (Seri haberleşme kanalı)

• Çok kaynaklı / vektörlü / öncelik seviyeli kesme yapısı (Interrupts)

• On-chip saat osilatör

3.2.3 8052 Türevi Mikrodenetleyiciler

Aşağıda 8051/8052 mimarisini kullanarak mikroişlemci üreten firmaların bazıları bulunmaktadır. Aynı zamanda bu firmalar 8051/8052 yapısına kendi bünyelerinde yeni özelliklerde eklemişlerdir.

Çizelge 3.2 8051 üreten firmalar ve chipleri [18]

NO:	Firma:	İnternet Adresi:	Genel Özellikler
1	ACER LABS	http://www.ali.com.tw/	8051
2	AEROFLEX	http://www.utmc.com/	8051+Flexible Core Clock İşlemi(1Hz-20Mhz'e harici Clock),(2MHz-20MHz Dahili osilatör)
3	AMD	http://www.amd.com/	8051+Kablosuz Telefon Chip seti
4	ANALOG DEVİCE	http://www.analog.com/	8051+ yüksek çözünürlüklü ADC
5	ATMEL	http://www.atmel.com/	8051
6	CAST	http://www.cast-inc.com	8051+ ortalama 8 kat daha hızlı çekirdek yapısı+ onchip debug
7	CHIPCON	http://www.chipcon.com	8051+ yüksek frekanslı RF Tranciever
8	CONTROL CHIPS	http://www.controlchips.com	Standart 8052
9	CML	http://www.cmlmicro.com	8051+LCD Kontrol Ara yüzü+ İntegral Modem+A/D converter+8*16 klavye+2 Adet PWM Çıkışı
10	CYBRA TECH.	http://www.cybratech.com	8051
11	CYGNAL	http://www.cygnal.com	8051+128Kbyte'a kadar Flash Memory+5 adet 16 bit Timer+ USB 2.0
12	CYPRESS	http://www.cypress.com	8051+Full speed USB bağlantısı

13	DAEWOO	http://www.daewoosemicon.co.kr	8051+2 kademe programlanabilir seri port+ devre üzerinde emülasyon
14	DALLAS	http://www.dalsemi.com	8051+ dual DPTR+ OnChip-Debug+ 16Kbyte EPROM+kristal frekansını 64'den 1024'e kadar bölerek çalışma+ 2 adet seri port+
15	DOLPHIN	http://www.dolphin.fr	8051+ ortalama 9 kat daha hızlı çekirdek yapısı
16	DOMOSYS	http://www.domosys.com	8051+ ortalama 8 kat daha hızlı çekirdek yapısı+ 8 bit A/D conveter ara yüzü SPI'dan
17	GOAL	http://www.goalasic.com	8051+2 capture ve yakalma ünitesi+4 PWM çıkışı+ 2 seri UART+ 4 kanal ADC
18	GOLDSTAR		
19	HONEYWELL	http://www.ssec.honeywell.com	8051+900Hz RF tranciever
20	HYNIX	http://www.hynix.com	8051+ 40MHz'e kadar hız+USART
21	ICSI	http://www.icsi.com.tw/english/	
22	INFINEON	http://www.infineon.com/	8051+ 29 kanal PWM çıkışı+USART+4 kaynaklı 19 interrupt
23	INNOVASIC	http://www.innovasic.com	8051+48MHz'e kadar hız+3.3V ile besleme
24	INTEL	http://www.intel.com	8051
25	ISSI	http://www.issi.com/	8051+iki level öncelik kademeli Flash Memory
26	MACRONIX	http://www.macronix.com/	8051+ dokunmatik ekran controller
27	MAXIM		
28	MICRONAS	http://www.micronas.com	8051+Teletext decoder+tek karakter seti++batı ,doğu avrupa,fars,arap tabanlı dil+programlanabilir ekran boyutu+Ekran kaydırma donanımı+ 14 bit iki kanal PWM
29	MICROTUNE	http://www.microtune.com	8051+bluetooth+ 2GHz frekans tranciever+USB 1.1
30	MENTOR GRAPH.	http://www.mentor.com	8051+On-Chip debug

31	MOSEL-VİTELİC	http://www.moselvitelic.com	8051+LCD'li Voice Smart+ Mouse Controller
32	MYSON CENTRY	http://www.century-semi.com	8051+dijital video encoder+çeşitli monitör tiplerine emülasyon+ üç öncelik kademeli 8 interrupt kaynağı
33	NORDIC	http://www.nvlsi.no/	8051+ 2.4 GHz ISM band radio+ 9 kanal 12 bit ADC
34	OREGANO	http://www.oergano.at	Full senkron devre dizaynı
35	OKI	http://www.oki-europe.de/	8051
36	PHILIPS	http://www.philips.com	
37	RDC	http://www.rdc.com..tw	8051+8 bit IP+ 16 bit network işlemcisi
38	QUICKCORES	http://www.quickcores.com	8051+ Evolution Board
39	SAMSUNG	http://www.intl.samsungsemi.com/	
40	SANYO	http://www.sanyo.com/	8051+10-12 bit ADC Converter
41	SHARP	http://sharp-world.com	8051+ 4 kanal DMA Controller+ üç kanal programlanabilir TIC+ Floppy disk controller
42	SİLİCON STOR.	http://www.sst.com/	
43	SILICONIAS	http://www.siliconias.com	8051
44	SMSC	http://www.smsc.com/	
45	ST	http://www.st.com	8051+ 128Kbyte Flash Memory+ 32 Kbyte Flash Memory+DDC+LVD+ 8Kbyte Battery Backup+PLD+USB1.1+DDCPROM+ keypad ara yüzü+üstün entegre(sadece enerji ve kristal eklenir)
46	SYNCMOS	http://www.syncmos.com.tw	8051+ 40Mhz Clock Frekansı
47	TDK	http://www.tdksemiconductor.com	8051
48	TEMIC	http://www.temic-semi.de	
49	TEXAS INS.	http://www.ti.com	8051+ 8 kanal analog giriş 24 bit çözünürlük+ burn out sensor +16 bit PWM+ 21 interrupt kaynağı
50	TRISCEND	http://www.triscend.com	8051+üç harici 10 dahili interrupt+40Mhz Core+ 10MIPS

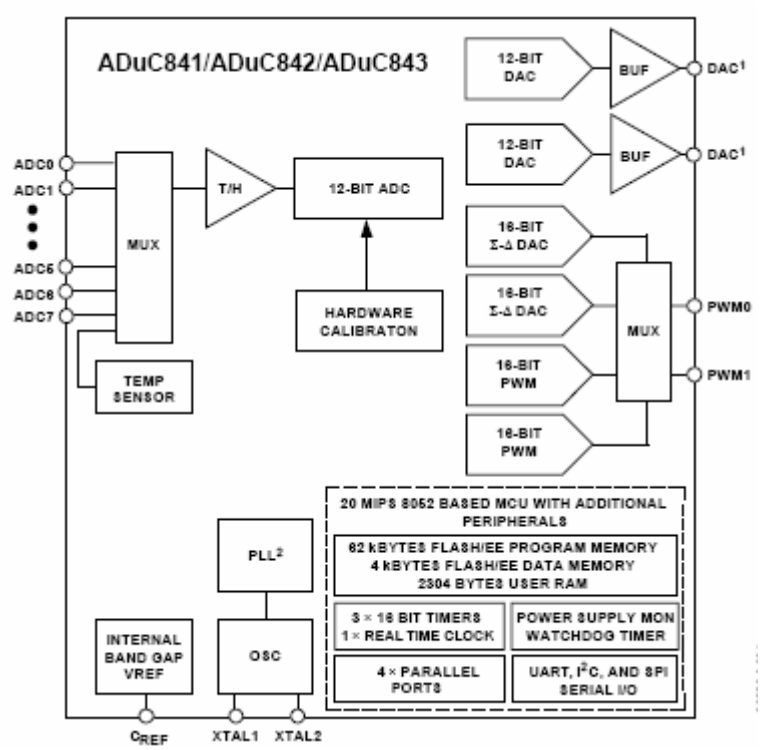
51	WINBOND	http://www.winbond.com	8051+1280 byte RAM+64Kbyte ROM+ 1.8V ile 5.5V arası besleme+PWM+ Extra I/0+ wait state control signal
52	WINEDGE	http://www.winedge.com.sg	8051+12 kat hız +hızlı interrupt tepki+ (12+2) interrupt kaynağı + 2 DPTR +4 Kbyte Data RAM
53	WELTREND	http://www.well/trend.com.tw	8051+SYNC sinyal işlemcisi H/V ayrımlı+SYNC sinyal çıkışı (ayrı çalışan)+harici IRQ+14 kanal 8 bit PWM+CRD ve LCDler için karakter görüntüleme+ LCD kontrolü+USB 1.0
54	VERSACHIPS	http://www.vchips.co.kr	8051+ 40 DIP+ 4 kademeli interrupt önceliği+6 interrupt kaynağı+ A/D converter

3.2.4 Tasarlanan Sistemde Kullanılan ADuC841

Tasarlanan sistemde mikroişlemci seçimi yaparken öncelikle işlem hızının yüksek olmasına ve ADC çözünürlüğünün yüksek olması dikkate alındı. Tasarımda kullanılan ADuC841 mikrodenetleyicisi Analog Devices tarafından 8052 ana yapısı üzerine özellikle 8 kanal 12 bit 420 kSPS (kilo sample per second) ADC eklenmiş bir modelidir. Tasarımda ADuC841'in 8 kanal ADCsi kullanılmıştır. Bu sekiz kanal ADC ile oda sıcaklığı üç farklı sensörden, odanın içindeki ışık şiddeti üç farklı noktadaki sensörden ölçülmüş ve kullanıcı ayarlarının girilmesi için de iki ADC kanalı kullanılmıştır.

ADuC841 mikrodenetleyicisinin 8052 yapısı üzerine eklediği PLL (Phase Lock Loop) yapısı ile 8052 ana yapısına 20 MIPS (Million Instruction Per Second) işlem yapabilme özelliği eklenmiştir. Bu özelliği sayesinde ADuC841 ile işlemler hızlı bir şekilde gerçekleştirilmiştir [17].

ADuC841 8052 den farklı olarak üzerinde 2304 byte RAM bulundurmakta ve 62 Kbyte a kadar çıkabilen program hafızasını içinde barındırmaktadır. Büyük RAM ve hafıza alanı yazılım için daha rahat, daha kullanışlı bir geliştirme ortamı sağlamaktadır. Ayrıca mikrodenetleyici üzerinde bulunan on-chip debug özelliği ile mikrodenetleyicinizin üzerinde olduğu karttan, gerçek ortam içinde rahatlıkla hataları ayıklayıp, işlemlerinizi düzenleyebilir [17].



Şekil 3.5 ADuC841 Mikrodenetleyicisinin İç Yapısı [17]

3.3 Kullanılan Cihazların Kontrolü

3.3.1 Isıtıcı Kontrolü

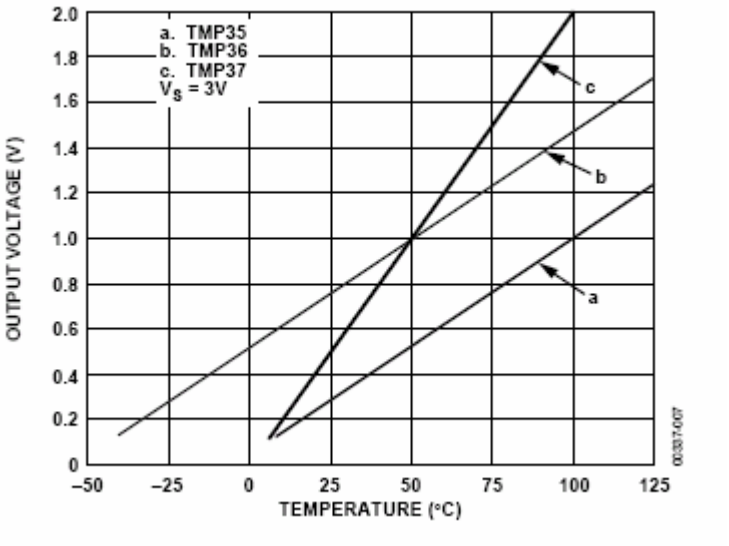
İnsanların bulunduğu bir ortamda konfor için önemli bir nokta da ortam sıcaklığıdır. Ortam ısısı ısıtıcının ve soğutucunun gerekli zamanlarda, gerekli oranlarda devreye sokulmasıyla ayarlanır. İnsan vücudu ortam sıcaklığının küçük değişimlerini algılayamaz. Bu yüzden ısıtıcının veya soğutucunun her ortamda aynı seviyede çalışması yerine kademeli bir çalışma şekli seçilmelidir. Bu kademenin derecesi bulanık mantık kontrolü ile sağlanabilir.

Tasarlanan sistemde her oda içersinde üç farklı noktada ayrı ayrı sıcaklık sensörleri bulunmaktadır. Bu üç sıcaklık sensöründen alınan sıcaklık değerleri ortalama alma işlemi ile odanın sıcaklık bilgisine dönüştürülür. Bu şekilde odanın her noktasındaki sıcaklığın aynı tutulması sağlanmıştır.

3.3.2 Kullanılan Sıcaklık Sensörü: TMP35

Sistem tasarımında sıcaklık sensörü olarak Analog Device'ın TMP35 adlı sıcaklık sensörü kullanıldı. TMP35 2.7 ile 5.5 V aralığında çalışma gerilimine sahip lineer bir sıcaklık sensörüdür. TMP35 10mV/ °C ölçüm faktörüne sahiptir. Ayrıca 50 µA den az bir akım

sarfiyatı bulunmaktadır. Şekil 3.6 daki grafikte (a) ile gösterilmiş grafikte TMP35'ın ortam sıcaklığına göre verdiği çıkış görülmektedir.



Şekil 3.6 TMP35'in sıcaklık-çıkış voltaj grafiği [17]

3.3.3 SAR ve Sigma Delta ADC Yapısı

ADuC 841 üzerinde sekiz kanal 12 bit Σ-Δ yapısında Analog-Dijital çevirici (ADC-Analog Digital Converter) bulundurmaktadır. Analog dijital çevirileri Σ-Δ (Sigma-Delta), SAR (Successive Approximation Register), Flash gibi birçok benzer yapıdadır. Analog Devices tasarladığı microdenetleyiciler veya onların tabiri ile mikrodönüştürücüler (microconverter) üzerinde SaAR veya Σ-Δ ADC yapısını kullanmaktadır [16].

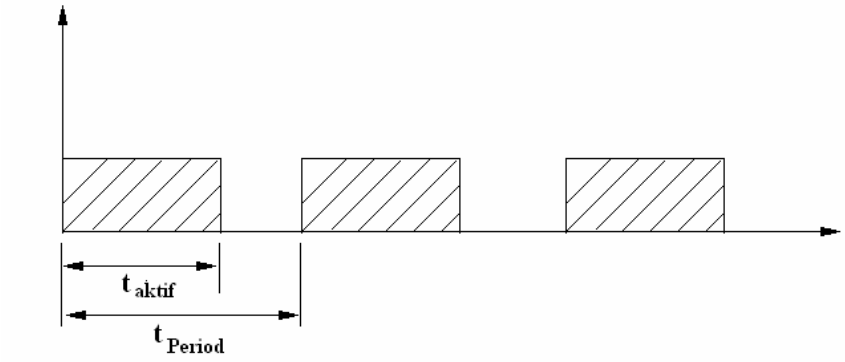
SAR-(Successive Approximation Register) yani Ardışıl Yaklaşım Yöntemi, DAC'a giriş üretilen bu teknik, bilinmeyen bir maddenin (örneğin 1gramdan daha az) ½ gr, ¼ gr, gibi kesirli ağırlıklarla bir terazide tartılmasına benzer. Tartma işlemi en ağır ½ gr ile başlar ve denge bozulana kadar daha düşük ağırlıklar eklenir. Dengeyi bozan ağırlık çıkartılır ve işlem en küçük ağırlık kullanılıncaya kadar devam eder.

Σ-Δ yapısındaki ADClerde ise ADC işlemi SAR yapısından biraz daha yavaş ancak daha hassastır. Σ-Δ yapısındaki dönüşümü bir tartım işlemi yapılırken tartılacak olan madde ne olursa olsun, bu tip ADC ler kullanabildikleri en küçük ağırlığı tek tek terazinin diğer kefesine koyup karşılaştırırlar. Ne zamanki en iyi sonuç elde edilir veya denge ters yönde bozulursa dengeyi bozan son en küçük parçayı alarak sonucu elde ederler.

3.3.4 Sıcaklık Kontrol Devresinde Kullanılan PWM Kontrolü

ADuC841 mikrodenetleyicisi Dual 8/16 bit PWM (Pulse Width Modulation-Darbe Genlik Modülasyonu) modülüne sahiptir.ADuC841 işlemcisinin PWM pin çıkışları başka amaç için kullanıldığından dolayı mikrodenetleyicinin üzerinde bulunan 2 adet PWM çıkışı kullanılamamıştır. Ayrıca PWM çıkışı olmayan durumlarda da alternatif yol olarak kullanılabilen Timer kontrollü PWM çıkışı yazılım ile gerçekleştirilmiştir. Sıcaklık kontrol yazılımında bu yöntem tercih edilmiştir.

PWM genel çalışma prensibi sinyalin belirli bir süre içerisinde aktif, diğer süre zarfında ise pasif olmasıdır. PWM için gerekli olan süreler Timer modülü ile elde edilir. Sıcaklık kontrolünde bulanık mantık kullanılmıştır. Bulanık mantık kontrolünde çıkış değeri PWM'in aktif olma süresini vermektedir. Yazılan yazılımda Timer0 modülü 100 milisaniyeye ayarlanmıştır. Bulanık mantık çıkışı örneğin 70 olarak elde edilmiş ise, bu ısıtıcının veya soğutucunun 100 milisaniyelik periyot içerisinde %70, aktif %30 pasif olması anlamına gelmektedir.



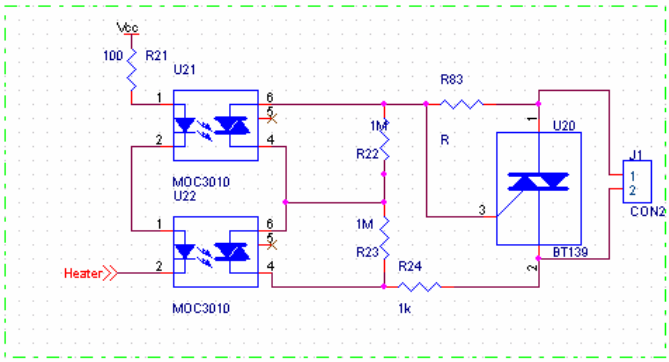
Şekil 3.7 PWM sinyali

3.3.5 Isıtıcı Sürme Devresi.

Piyasada bulunan ısıtıcıların çalışma gerilimi 220V AC mikroişlemcinin çalışma geriliminin 5V DC dır. İki çalışma gerilimi arasında büyük farklılıklar olduğu için, iki sistem arasında ilişki kurabilecek bir devreye ihtiyaç duyulmuştur.

220 V Alternatif akımlı gerilim kısmını kontrol etmek için triyak kullanılmalıdır. Sistemde kullanılan triyak, sistemde kullanılacak olan ısıtıcının gücüne göre belirlenmelidir. Bu sistemde kullanılacak olan triac 50W ile 100W arasında kalacağı düşünülerek BT139 Triac'ı seçilmiştir. Triac üzerinde 220V gerilim olduğu ve Triac'ı tetiklemek için 220V dan biraz

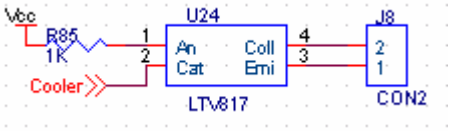
daha yüksek bir gerilime ihtiyaç duyulduğu için seçilen triac'ı tetiklemek için 2 adet birbirine seri bağlı MOC3010 dan oluşan bir tetikleme devresi tasarlanmıştır. Şekil 3.8 de 220V kontrol devresi görülmektedir.



Şekil 3.8 220V kontrol devresi,

3.3.6 Soğutucu Kontrolü

Sıcaklık kontrolünde odanın soğutulmasında 12 Volt gerilim ile çalışan fan kullanılmıştır. Fanlar, oda duvarına monte edilerek çalıştırıldıkların da hava akımı ile oda sıcaklığını düşürürler. Fan kontrolünde şekildeki kontrol devresi kullanılmıştır. Soğutucuların kontrolünde de PWM kullanılmıştır. PWM'in aktif olma süresi soğutucular için bulanık mantık algoritmasından elde edilir.



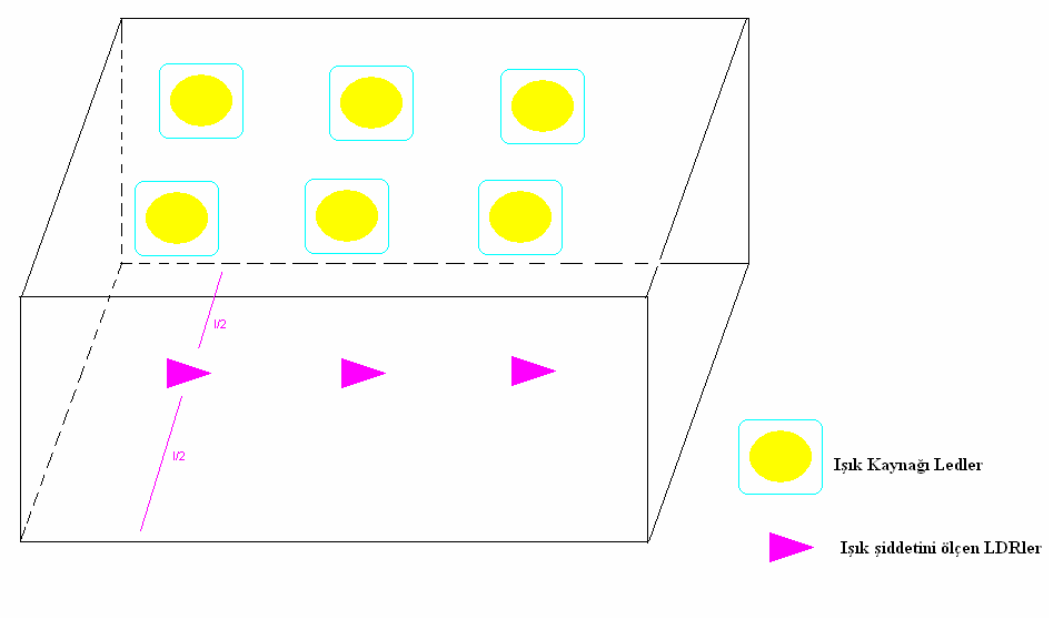
Şekil 3.9 Soğutucu devresi

3.3.7 Işık Kontrolü

Oda aydınlatmasında ışık kaynağı olarak tek bir kaynak seçilmesi durumunda odanın her noktasında aynı aydınlık seviyesi sağlanamamaktadır. Tek bir ışık kaynağının yerine grup ışık kaynakları seçilmesi durumunda ise odanın her yeri aynı şekilde aydınlatılabilir, ama bu durum ekonomik değildir. Çünkü gündüz ışığında oda pencerelerden içeriye ışık almaktadır. Işık şiddetinin kontrol edilmeden ışık kaynaklarının sürülmesi, aydınlık seviyesi düzgün olan bir noktayı aydınlatmak için fazladan enerji kullanılmasına neden olur.

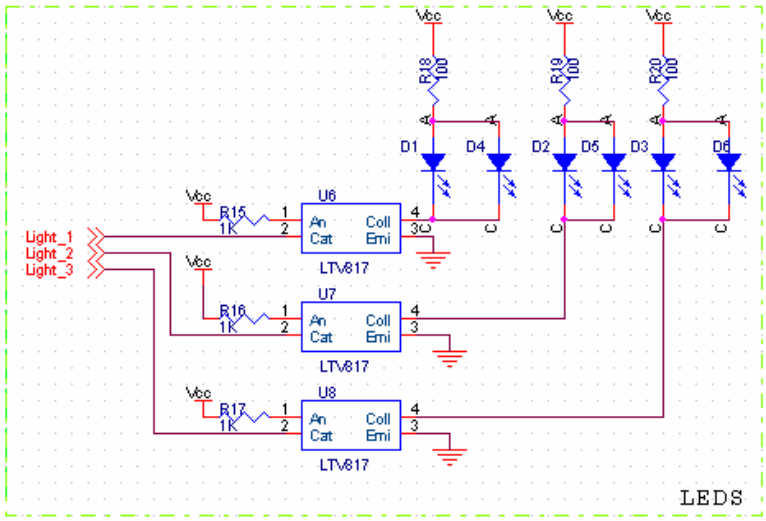
Tasarlanan sistemde oda içerisinde her noktada aynı ışık şiddetinin olması amaçlanmıştır. Bunun için odanın içerisinde farklı ışık şiddetlerine maruz kalan noktalara yerleştirilen LDR(Light Dependent Resistance) ile bu noktalardaki ışık şiddeti hakkında bilgiler elde edilir. Bu bilgiler genel bir filtreden geçirildikten sonra P kontrol algoritmasıyla her noktadaki aydınlık seviyesi eşit, ancak ışık kaynaklarından yayılan ışık şiddeti ölçüm yapılan noktaya göre değişiklik gösterir. Bu şekilde gereksiz yere sürülen ışık şiddeti ile hem enerji tasarrufu yapılmış hem de sağlıklı bir aydınlatma sağlanmış olur.

Tasarlanan sistemde üç adet LDR ile ışık bilgisi alınmış ve bu bilgiye göre iki adet ledden oluşan üç led grubu sürülmüştür. Bu led grupları pencereye yakın, odanın ortasında ve pencereden uzak olacak şekilde konumlandırılmıştır.

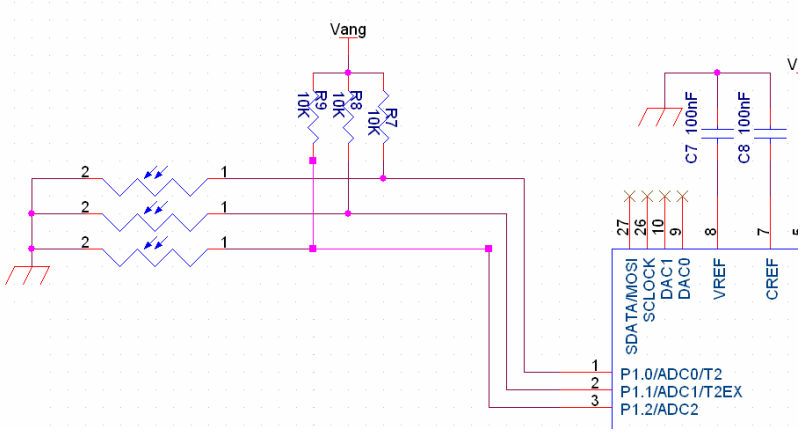


Şekil 3.10 Odadaki ledler ve LDR'ler.

Işık kontrolünde de PWM kullanılmıştır. PWM'in aktif olma süresi P kontrol algoritmasından elde edilir. P kontrol algoritması LDR'lerden aldığı ışık bilgisini ortalamasını alarak kontrol algoritmasına sokmaktadır. Kontrol algoritmasında ışık şiddeti ile kullanıcının ayarlamış olduğu değer arasındaki hataya göre işlem yaparak led grubunun sürülmesinde kullanılan PWM aktif olma süresi belirlenir. Bu PWM sürelerine göre led grupları ayrı ayrı sürülür.



Şekil 3.11 Led gruplarını sürme devresi



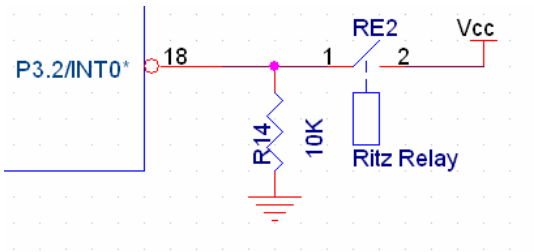
Şekil 3.12 LDRler ile ışık şiddeti ölçme devresi

3.3.8 Güvenlik Kontrolü

Bir evde yaşayan insanların yaşam kalitesini artıran önemli unsurlardan biri de o evde insanın kendisini güvende hissetmesidir. Aynı şekilde evde olmadığında da evinin güvende olduğundan emin olmak ister. Bu yüzden yapılan akıllı ev otomasyonunda güvenlik kontrolü de işlenmiştir.

Odalarda bulunan pencereler ve ev giriş kapısı güvenliğin zayıf olduğu noktalardır. Bu yüzden bu noktaların kontrolü ile yaşanılan ev ortamının güvenliği sağlanmış olur. Pencerelerin evde kimse yok iken konumlarındaki değişim, eve izinsiz bir giriş yapıldığı hakkında bilgi verir. Pencerelerdeki değişim, manyetik bir role çeşidi olan rit röle yardımı ile oda kontrolünü yapan mikroişlemciye iletilir. Bu işlemcide ana bilgisayar ile haberleşerek pencerenin açıldığını bildirir. Ana bilgisayar ise evde kullanıcı yok ise alarm durumuna geçer.

Pencerelerde kullanılan rit rölenin çalışma prensibi, manyetik bir alana girdiğinde kısa devre olması, manyetik alan etkisi dışında kaldığında ise açık devre konumunda olmasına dayanır. Tasarlanan sistemde pencere pervazına rit röle, pencereye rit rölenin karşısına gelecek şekilde mıknatıs yerleştirilmiştir. Pencere açıldığında rit role çıkış verir. Bu çıkış mikroişlemcinin dış kesmesine sinyal göndererek, sistemin pencerenin konumunun değiştiğini anlamasını sağlar. Bu şekilde sistem evin güvenliğinin düşük olduğu kapı pencere gibi noktalarda güvenliği arttırmış ve insanlar için daha rahat ve güvenli bir ortam sağlamış olur.



Şekil 3.13 Rit role ile güvenlik kontrol devresi

3.3.9 Haberleşme RS232

Tasarlanan sistemde ana kontrol ünitesi bilgisayar olduğu için tasarımı gerçekleştirilen yazılımın bilgisayar ile iletişim halinde olması gerekir. Bu iletişim için Ethernet, RS485, Wireless gibi haberleşme protokol ve donanımları kullanılabilir ancak sistem tek bir bilgisayar ile iletişim halinde olacağı düşünülerek RS232 üzerinden bilgisayar ile iletişim sağlanmıştır.

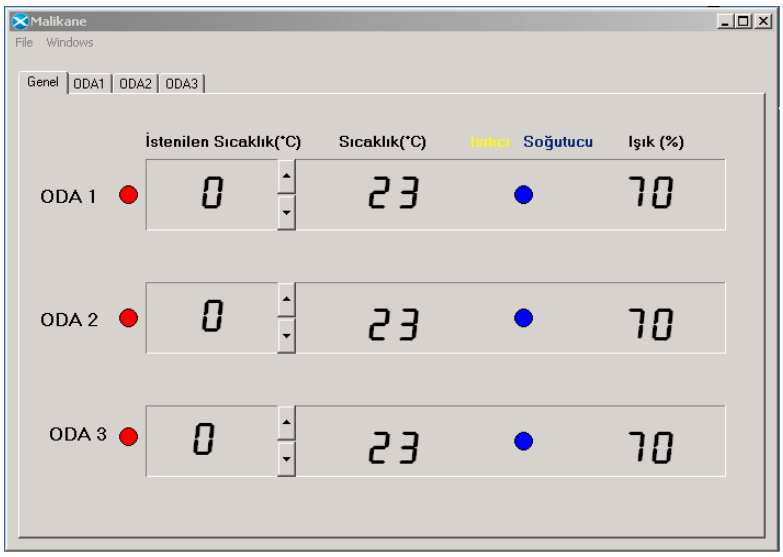
Sistem odanın sıcaklığını, kullanıcı sıcaklık set değerini, ışık değerini ve güvenlik amaçlı pencere ve kapının durumunu RS232 üzerinden bilgisayar tarafından sorgu yapıldığında gönderir.

RS232 haberleşmesi asenkron seri haberleşme protokolüdür. Asenkron denmesi bilgi aktarımına başlamak için belirli bir koşulun olmaması demektir. Herhangi bir anda, veri aktarımına başlanabilir. RS232 haberleşme protokolü ±12 volt seviyesinde çalışır.

RS232 haberleşmesinde uzun mesafeler kullanılamaz. En fazla 20 metre mesafede bilgi akışı sağlıklı yapılabilmektedir. Daha uzun mesafeler için farklı bir haberleşme protokolü seçilmelidir. RS485 veya Ethernet gibi.

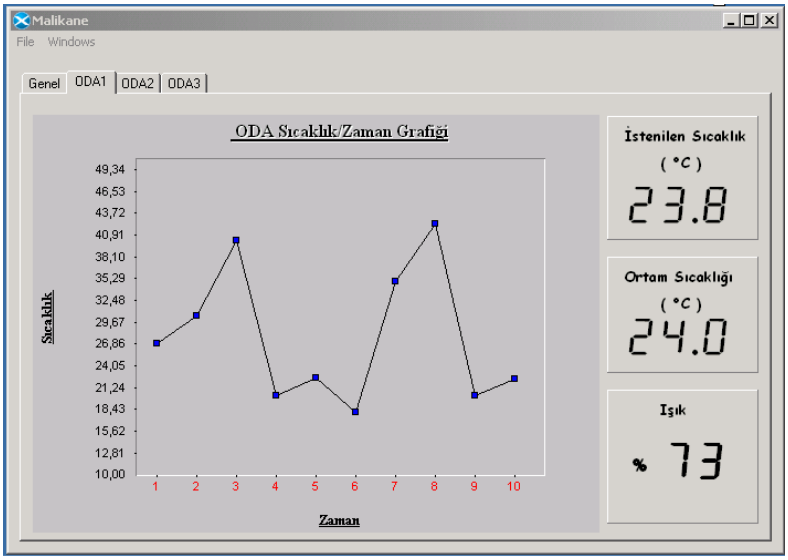
Tasarımı yapılan elektronik kart ile bilgisayar arasındaki haberleşme 9600 baud ratede 8 bit data 1 bit stop biti ve no parity ile gerçekleştirilmektedir.

Bilgisayar yazılımı saniyede bir tasarımı yapılan sisteme oda bilgilerini sorar ve cevap bekler. Gelen cevaba göre bilgisayar yazılımında kullanılan arayız tazelenir. Ana pencerede sisteme bağlı olan odaları sıcaklıkları görülür.



Şekil 3.14 Bilgisayar Yazılımında ana gösterge paneli

Kullanıcı odaların ayrıntılı bilgilerini görmek istediklerinde ilgili odaya tıkladıklarında, oda sistemin açılışından o ana kadar ki sıcaklık değişimini grafikte görebilirler.



Şekil 3.15 Bilgisayar yazılımında oda gösterge paneli

3.3.10 Tasarlanan Sistemin Devre Kartının Tasarlanması (PCB tasarımı)

Tasarlanan sistemin elektronik kart tasarımı sistem içindeki yazılım kadar önemlidir. Eğer yapılan kartta görülemeyen hatalardan dolayı sistem düzgün çalışmayabilir. PCB, printed circuit board, basılmış devre kartı anlamına gelir. PCB üzerine elektronik, mekanik, yarı elektronik yarı mekanik elemanlar yerleştirilerek sistem tasarlanır.

PCB tasarımı yapılmadan önce elektronik olarak kullanılması düşünülen elemanlar ve kullanma yapıları önceden belirlenmiş olması gerekir.

Yapılan tasarımda analog ve dijital sistemler aynı kart üzerinde bulunmakla beraber kart üzerinde farklı gerilimlerde bulunmaktadır. 220V AC ısıtıcı için, 12V DC soğutucu için ve 5 volt mikrodenetleyici ve gösterge kısmı için gereklidir.

Tasarımda Analog kısmın toprak bağlantısı ile dijital kısmın toprak noktasının bağlantı noktasına özellikle dikkat etmiştir. Dijital kısımda çıkış ünitelerinin sürülmesi sırasında oluşan gürültülerin Analog kısmı etkilemesi gerekir. Eğer dijital kısımda oluşan gürültünün analog kısımda sıcaklık sensörlerinin beslemesine veya LDR lerin üzerine gelmesi ADC den alınan sıcaklık veya ışık bilgisinde hatalar ile karşılaşılır. Analog ve dijital kısıma besleme tek bir noktadan ve yüksek değerlikli kapasite üzerinden verilmesi oluşabilecek gürültü etkilerini en aza indirebilir. Bu kapasitenin yanında yazılımda da bazı filtreleme işlemlerinin de yapılması gerekir.

3.3.11 Tasarımda MCU Yazılımında Kullanılan C Programlama Dili

C dili, 1970li yılların başında Amerika'da, Bell Laboratuarlarında (Bell Labs) Dennis Ritchie tarafından geliştirilmiştir [10].

C programlama dilinin doğuşu, Unix'in doğuşu ile paralellik gösterir. C, Unix sistemlerinin geliştirilmesinden sonra bu sistemlerde programlama yapmak amacıyla 1969-1973 yılları arasında geliştirilmiştir. C, en hızlı gelişimini 1972 yılında yaşadığı için, dilin doğum yılı birçok yayında 1972 olarak geçer. Unix sistemlerin geliştirilmesi ile birlikte, C dili de 1977 - 1979 yılları arasında ikinci bir değişime uğramış ve en son 1980'li yılların ortasında ANSI (The American National Standarts Institue) isimli organizasyonun dili standartlaştırması ile bugünkü şeklini almıştır.

C Dilinden önce, yerine Unix sistemlerini programlama amacıyla kullanılan BCPL ve B dilleri vardı. Bu diller, C dilinin geliştirildiği laboratuarlarda Martin Richards ve Kan Thompson tarafından 1960'lı yılların sonlarına doğru geliştirilmişti. Ancak bir süre sonra bu diller yetersiz kaldı ve böylece C dili doğdu. Kısa sürede büyük gelişmeler gösteren C dili,1980'li yıllarda standartlaştırıldıktan sonra tüm dünyada ilgi gördü. C dili başlarda Unix sistemler için geliştirilmiş olmasına rağmen, bugün her türlü sistem için hazırlanmış olan derleyiciler (Compiler) sayesinde günümüzde tüm sistemlerde çalışabilecek duruma gelmiştir. C'nin bu derece ilgi görmesinde " C Programming Language" isimli kitabında etkisi büyüktür. Bu kitap ANSI tarafından standartlaştırılarak tek kaynak olarak kullanılmıştır [10].

3.4 Ev Otomasyonunda Oda Model Üzerinde Parametrelerin Kontrolü

Tasarımı gerçekleştirilen sistemin birebir gerçek ortamda simülasyonu ve gerçek ortam sorunlarının gözlenebilmesi için küçük bir oda modeli yapılmıştır. Evin bir odası örnek alınmış, ana sistem kontrolü de bilgisayar üzerine yazılmış olan arayız programı ile gerçekleştirilmiştir. Örnek olarak hazırlanmış oda modelinde ışık kaynağı olarak ikili gruplardan oluşan üç grup led ışık kaynağı, ışık şiddetini ölçebilmek için üç adet LDR, oda sıcaklığını ölçmek için üç adet sıcaklık sensörü, oda sıcaklığını arttırabilmek için 220 volt AC gerilim ile çalışan ısıtıcı, oda sıcaklığını düşürebilmek için 12 volt DC gerilim ile çalışan fan ve güvenlik kontrolü için rit role kullanılmıştır. Kontrol edilen parametreler, ısı, ışık ve güvenliktir.

3.4.1 Sıcaklık

Oda içerisinde kontrol edilen parametrelerden biri de ısıdır. Isı kontrolünde çeşitli yöntemler kullanılabilir. On/Off kontrolü en çok tercih edilen yöntemdir. Bu yöntemde odanın ısısına bakılmaksızın sistem belirli bir ısıtma kademesine ayarlanır ve kullanıcı bu kademeyi değiştirene kadar oda aynı kademede ısıtılır veya soğutulur. Bunun yerine geri beslemeli bir sistem kullanılırsa, yani odanın ısısı ölçülüp, buna göre işlemler gerçekleştirilirse ekonomik açıdan kazanç sağlanmış olur. Isı kontrol sistemlerinde geri beslemeli kontrol için çeşitli yöntemler kullanılabilir. P, PI, PD, PID kontroller, piyasada yaygın olarak seçilen kontrol yöntemleridir.

Günümüzde iklimlendirme sistemi geleneksel yöntemler ile kontrol edilmektedir. Bu da çok büyük enerji kayıplarına yol açmaktadır. Bulanık mantık ile yapılan uygulama bu enerji kaybını neredeyse yarı yarıya azaltmaktadır.

Pratikte kullanılmak için tasarlanan sistemler her zaman belli bir modele indirgenemezler veya çok karmaşık sistemler ortaya çıkabilir. Bu gibi durumlarda bu sistemlerin kontrolü için denetleyici tasarlamak zorlaşmakta hatta imkansıza yaklaşmaktadır. Sistemlerin kontrolünde uzman kişilerin bilgisine başvurulmakta, uzman kişi az, biraz, çok, daha az gibi dilsel ifadeler ile denetleyici ile sistemi kontrol etmektedir. Oysa her sistem için gerekli olduğunda bir uzman kişi bulmak mümkün olmayabilir veya uzman kişiler arasında fikir ayrılıkları olabilir.

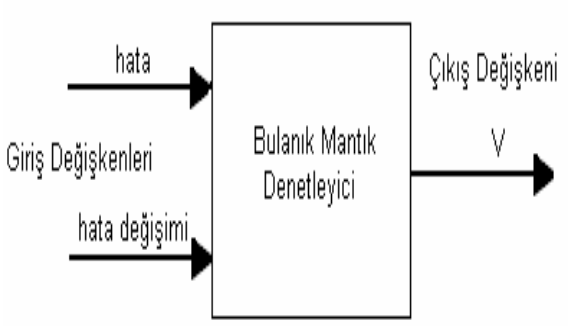
Bu dilsel ifadeler denetleyiciye direkt olarak verilirse ve denetleyici bu ifadeler ile işlemlerini gerçekleştirilirse uzman kişiye ihtiyaç ortadan kalkar. Böylece her an değişen sistemin onu her an denetleyebilecek özel bir uzman kişisi olur.

Geleneksel yöntemlerle iklimlendirme kontrolü dilsel ifadelerle değil birler ve sıfırlar ile gerçekleştirilmektedir. Yani ortam ya soğuktur ya da sıcaktır; ortam nemi ya fazladır ya da değildir. Oysa insan böyle katı ayrımlar yapmaz, ortamı serin, çok sıcak, ılık gibi dilsel ifadelerle tanımlar. İşte bu şekildeki dilsel ifadeleri yorumlamak için bulanık mantık kullanılmaktadır.

Bulanık mantık ile yapılan iklimlendirme çalışmaları %40 enerji tasarrufu sağlamaktadır. Bunun yanında bulanık mantık ile klima kontrolü yapmak tasarımcıya da büyük kolaylıklar sağlar. Klima cihazları lineer olmayan yapıdadırlar, bu yüzden belli bir transfer fonksiyonuna sahip değillerdir. Bulanık mantık kullanarak sadece istenen parametreler kontrol edilmektedir. Bu çalışmada da bulanık mantık kontrol algoritması kullanılarak ısı kontrolü gerçekleştirilmiştir.

3.4.1.1 Bulanık Mantık Kontrolü

Bulanık mantık ilk kez Berkeley üniversitesinde bilgisayar profesörü olan Lotfi A. Zadeh tarafından öne atılmıştır [6][7][8] .Lotfi Zadeh, bulanık mantığın kurucusu kümelerin bir çoğunun keskin sınırları olmadığını iddaa etmiştir. Zadeh 2 değerli mantığı ({0,1}) sürekli değerlere arttırmıştır [2].Bulanık mantık denetleyicinin temel yapısında gerçek girişi bulandıran bir bulandırma birimi, bilgi tabanını ve bulanık bilgiyi kullanarak karar veren karar verme birimi ve bulanık çıkışı gerçek çıkışa döndüren durulama birimi bulunmaktadır.



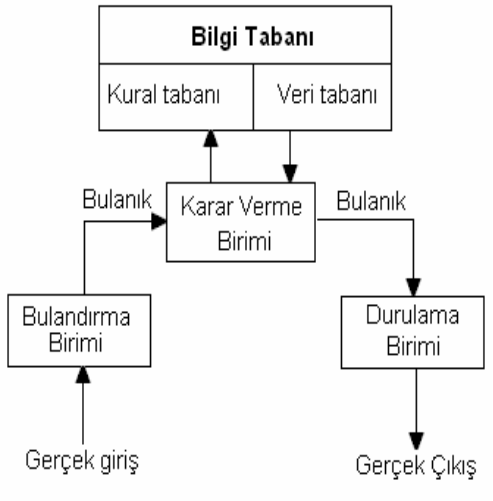
Şekil 3.16 Bulanık mantık blok gösterimi

3.4.1.1.1 Bulandırma Birimi

Bulandırma, sistemden alınan denetim giriş bilgilerini dilsel niteleyiciler olan sembolik değerlere dönüştürme işlemidir[1]. Bulanık mantık sistemi nümerik veriler ve dilsel bilgiler ile çalışmada benzersizdir.[3]. Bulanık mantık matematiğin bilgisayara gerçek dünyanın insan gibi modelleme yeteneği verilmiş dalıdır[5]. Bu sayede günümüz bilgisayar teknolojisi insan diline göre işlemler yapabilme yeteneğini kazanmıştır. Kavramlar uzun veya çok hızlı olarak matematiksel olarak formülüze edilebilir, bilgisayarlarda işlenebilir, insan gibi düşünen programlama yapıları uyugulanabilir[11]. Bulanık mantık kontrol ve sınıflandırma problemlerine farklı açıdan yaklaşmayı sağlar [12]. Aynı zamanda Bulanık mantık kuralları ve bulanık mantık kontrolcüsü gerçek zamanlı çalışabilmektedir [4].

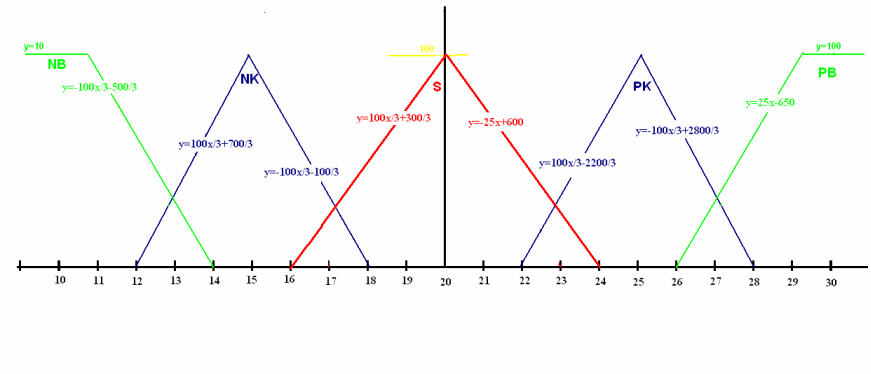
Bulandırma biriminde yapılan işlem gerçek girişi küçük, daha küçük, çok, çok fazla gibi dilsel değişkenlere dönüştürme işidir. Bunun yapılması için bulanık küme seçilmesi gerekir. En çok kullanılan bulanık küme şekilleri üçgen, yamuk, çan eğrisi, gaussiandır. Yapılan çalışmada üçgen şeklinde bulanık küme seçilmiştir. Sistemin gerçek girişi olarak hata ve hatanın değişimi alınmıştır.

İklimlendirme sisteminde bulandırma birimi sayısal giriş ve çıkış değerlerini sembolik değerlere dönüştürür. “e” girişi odanın ayarlanan sıcaklık değeridir.”ce”girişi ise odanın o andaki sıcaklık değeridir. “e” ve ”ce” için üyelik işlev fonksiyonu olarak üçgen giriş seçilmiştir.



Şekil 3.17 Bulanık mantık blok diyagramı

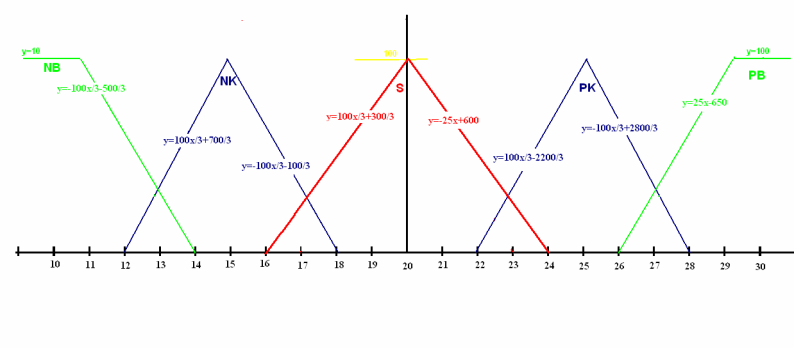
Hata bulanık kümesi beş üçgen üyelik fonksiyonundan oluşmaktadır. Bu beş üçgen üyelik fonksiyonu soldan başlamak üzere Negatif Büyük, Negatif Küçük, Sıfır, Pozitif Küçük, Pozitif Büyük olarak tanımlanmıştır. Şekil 3.18 de hatanın üyelik fonksiyonları gösterilmektedir.



Şekil 3.18 Hatanın üyelik fonksiyonları

Hatanın değişimi bulanık kümesi, hata gibi beş üçgen üyelik fonksiyonundan oluşmaktadır. Bu beş üçgen üyelik fonksiyonu soldan başlamak üzere Negatif Büyük, Negatif Küçük, Sıfır,

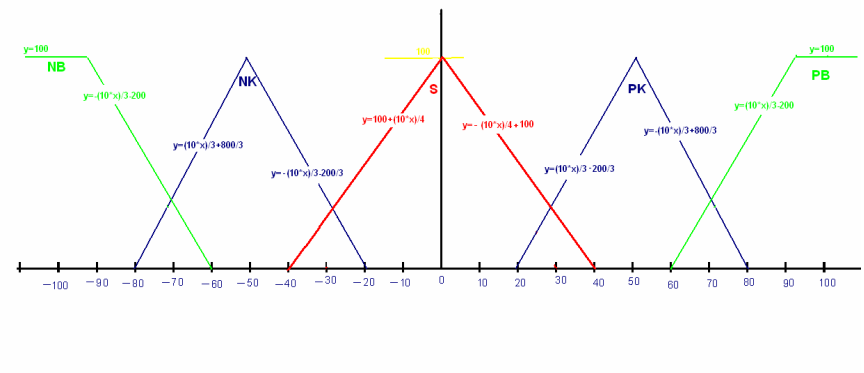
Pozitif Küçük, Pozitif Büyük olarak tanımlanmıştır. Şekil 3.19 da hatanın değişiminin üyelik fonksiyonları gösterilmektedir.



Şekil 3.19 Hatanın değişiminin üyelik fonksiyonları

Çıkış bulanık kümesi, hata gibi beş üçgen üyelik fonksiyonundan oluşmaktadır. Bu beş üçgen üyelik fonksiyonu soldan başlamak üzere Negatif Büyük, Negatif Küçük, Sıfır, Pozitif Küçük, Pozitif Büyük olarak tanımlanmıştır. Şekil 3.20 de çıkışın üyelik fonksiyonları gösterilmektedir.

Çıkış bilgisi, kullanılan ısıtıcı ve soğutucuyu sürmede kullanılan PWM'ın aktif olma süresi olarak elde edilir.



Şekil 3.20 Çıkışın üyelik fonksiyonları

Aşağıda yazılımda kullanılan hata, hatanın değişiminin ve çıkış üyelik fonksiyonlarının yazılan mikrodenetleyici programında C dili ile yapılan tanımları bulunmaktadır.

```
//üyelik fonksiyonlari
#define NB              0
#define NK              1
#define Zero    2
#define PK              3
#define PB              4

//Kural Tablosu
unsigned  const code RuleLine1[5]={Zero, PK , PB , PB , PB };
unsigned  const code RuleLine2[5]={ NK ,Zero, PK , PB , PB };
unsigned  const code RuleLine3[5]={ NB , NK ,Zero, PK , PB };
unsigned  const code RuleLine4[5]={ NB , NB , NK ,Zero, PK };
unsigned  const code RuleLine5[5]={ NB , NB , NB , NK , Zero};
```

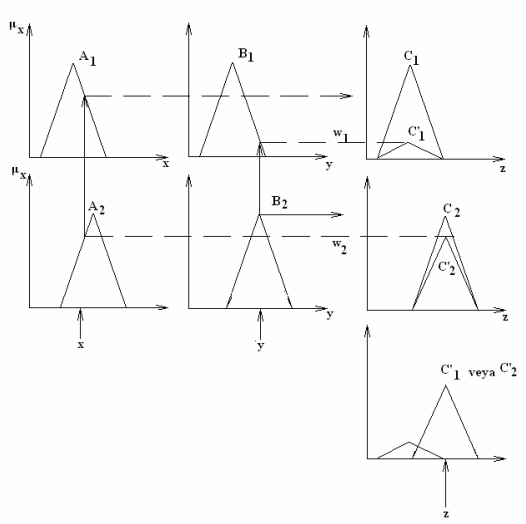
3.4.1.1.2 Karar Verme Birimi

Her bir giriş değeri bir veya birden fazla, üyelik fonksiyonuna ait olabilir. Bu üyelik fonksiyonuna aitlik derecesine (μ)mü denir. Her bir giriş değerinin (Hata ve Hatanın değişimi için) bulanıksal ifade ile ifade edilmesi gerekir. Her değişkenin bir veya birden fazla bulanıksal ifadeye sahip olabilme özelliği vardır. Bulanıksal ifade için üyelik fonksiyon isimleri ve bu üyelik fonksiyonları ile değişken arasında ilişkilendirilmiş (μ)mü değerine ihtiyaç vardır.

Mesela iki üyelik fonksiyonuna ait olan bir giriş değeri için, seçilen çıkarım yöntemi ile o giriş değerine ait olan μ (mü) değerleri bulunmuş, bulanıksal ifade elde edilmiş, bulanık çıkarım yapılmış olur. Bulanık çıkarım yöntemleri, Max-Dot, Min-Max, Tsukamato, Takagi-Sugeno olarak sınıflandırılabilir.

3.4.1.1.2.1 Max-Dot Bulanık Çıkarım Yöntemi

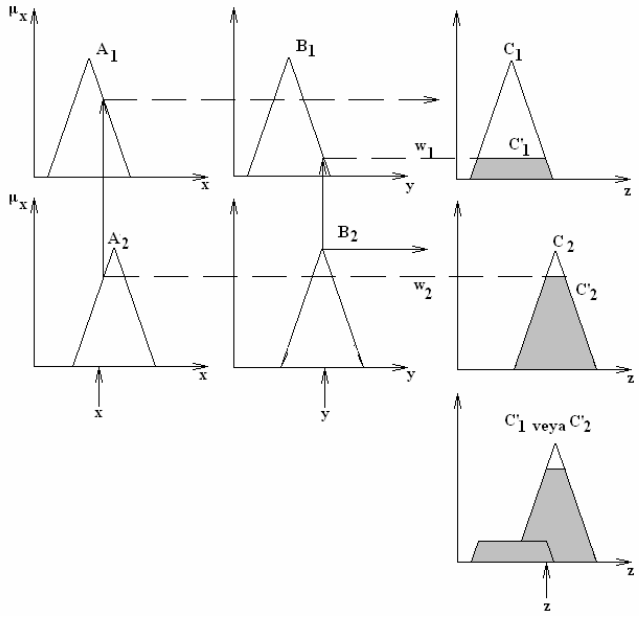
Her bir giriş değeri ait olduğu üyelik işlevindeki üyelik derecesine bağlı olarak, ilgili bulanık kümeyi yeniden ölçeklendirir. Çıkış değeri tüm girişiler için yeniden ölçeklendirilmiş bulanık kümeler içerisindeki maksimum değer alınarak bulunur [1].



Şekil 3.21 Max-dot çıkarım [1]

3.4.1.1.2.2 Min-Max Bulanık Çıkarım Yöntemi

Min-Max çıkarım yönteminde çıkış değeri elde edilen bulanık kümelere ağırlık ortalama yönteminin uygulanması ile bulunur [1].



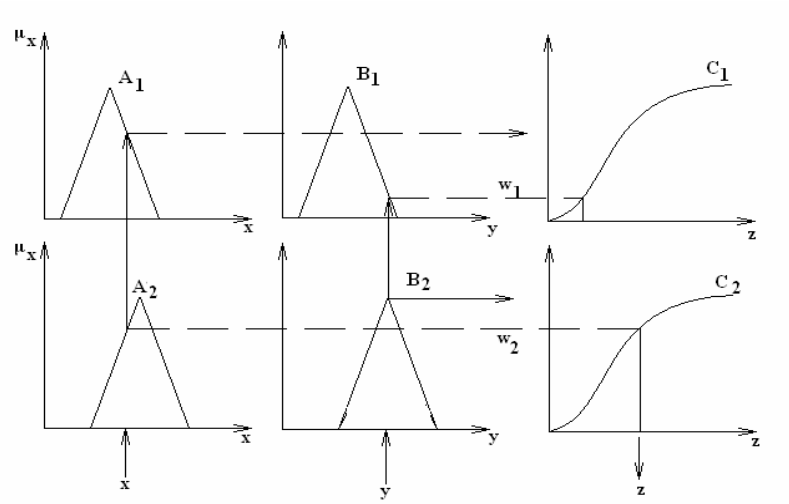
Şekil 3.22 Min-Max çıkarım [1]

$$z = \frac{\sum r_i z_i}{\sum z_i} \tag{3.1}$$

3.4.1.1.2.3 Tsukamato Bulanık Çıkarım Yöntemi:

Çıkış üyelik işlevi tek yönlü artan bir işlev olarak seçilir. Çıkış değeri ise her bir kuralın keskin çıkış değerinin ağırlık ortalaması alınarak bulunur [1].

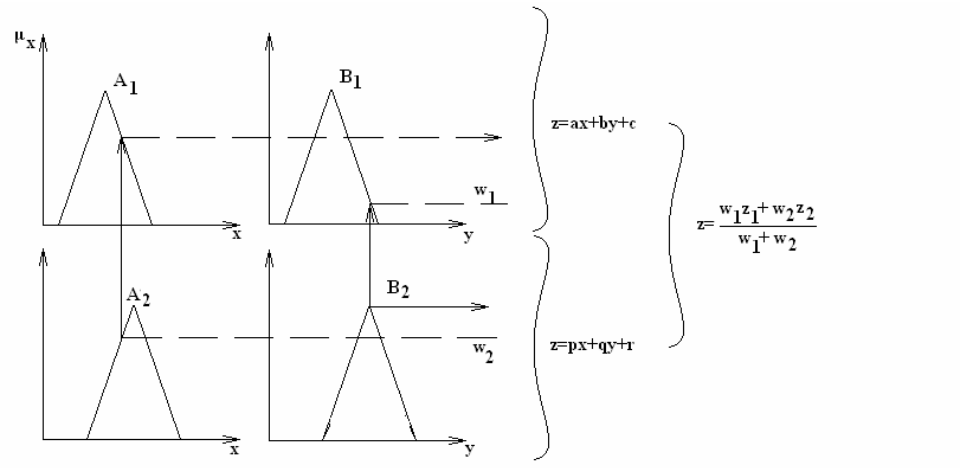
$$z = \frac{\omega_1 z_1 + \omega_2 z_2}{\omega_1 + \omega_2} \tag{3.2}$$



Şekil 3.23 Tsukamato çıkarım [1]

3.4.1.1.2.4 Takagi-Sugeno Bulanık Çıkarım Yöntemi

Her bir kuralın çıkışı giriş değerlerinin doğrusal birleşimi ile bulunur. Keskin çıkış değeri ise ağırlık ortalaması ile bulunur [1].



Şekil 3.24 Takagi-Sugeno çıkarım [1]

$$z_1 = f_1(x, y) = ax + by + c \tag{3.3}$$

$$z_2 = f_2(x, y) = ax + by + c \tag{3.4}$$

$$z = \frac{\omega_1 z_1 + \omega_2 z_2}{\omega_1 + \omega_2} \tag{3.5}$$

3.4.1.1.2.5 Mikrodenetleyici Üzerine Yazılan Bulanıklaştırma C kodu:

Tasarlanan sistemde bulanık çıkarım yöntemi olarak max-dot bulanık çıkarım yöntemi seçilmiştir.

Aşağıda mikroişlemci üzerine C programlama dilinde yazılan hata ve hatanın değişiminin bulanıklaştırmasını yapan "FuzzificationOfError" ve FuzzificationOfDeltaError" Fonksiyonları bulunmaktadır.

```
/*==================================================================
Fonksiyon Adi: FuzzificationOfError
Açiklama:      NewError degerlerini bulaniklastirir.
degerler milisantigrad
==================================================================*/
void FuzzificationOfError (void){
signed int MemberShip;

MemberShip=0;
if (NewError < ErrorTopOfNB){
        DegreeOfError[0]=ResolutionOfDegree;
        DegreeOfError[1]=NB;

        DegreeOfError[2]=NotADegree;
        DegreeOfError[3]=NotADegree; }
else if (NewError < ErrorEndOfNK) {
                MemberShip=0;
                MemberShip=(25*NewError)*(-1); //y=-25x+350
```

```
                MemberShip=MemberShip+350;
                DegreeOfError[0]=MemberShip;
                DegreeOfError[1]=NB;
                DegreeOfError[2]=NotADegree;
DegreeOfError[3]=NotADegree; }
else if (NewError < ErrorStartOfNB)  {
        MemberShip=0;
        MemberShip=(25*NewError)*(-1);  // y=-25x+350
        MemberShip=MemberShip+350;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=NB;



        MemberShip=0;
        MemberShip=100*NewError;                // y=100x-1200
        MemberShip=MemberShip-1200;
        DegreeOfError[2]=MemberShip;
        DegreeOfError[3]=NK; }
else if (NewError < ErrorTopOfNK) {
        MemberShip=0;
        MemberShip=100*NewError;                // y=100x-1200
        MemberShip=MemberShip-1200;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=NK;

        DegreeOfError[2]=NotADegree;
        DegreeOfError[3]=NotADegree;     }
else if (NewError < ErrorStartOfZero) {
        MemberShip=0;
        MemberShip=(100*NewError)*(-1);// y=-(100*x)/3+600
        MemberShip=MemberShip/3+600;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=NK;

        DegreeOfError[2]=NotADegree;
```

```
        DegreeOfError[3]=NotADegree; }
else if (NewError < ErrorStartOfNK) {
        MemberShip=0;
        MemberShip=(100*NewError)*(-1);// y=-(100*x)/3+600
        MemberShip=MemberShip/3+600;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=NK;

        MemberShip=0;
        MemberShip=25*NewError;                 // y=25x-400
        MemberShip=MemberShip-400;
        DegreeOfError[2]=MemberShip;
        DegreeOfError[3]=Zero;      }
else if (NewError < ErrorTopOfZero) {
        MemberShip=0;
        MemberShip=25*NewError;                 // y=25x-400
        MemberShip=MemberShip-400;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=Zero;
        DegreeOfError[2]=NotADegree;
        DegreeOfError[3]=NotADegree;  }
else if (NewError < ErrorStartOfPK)  {
        MemberShip=0;
        MemberShip=(25*NewError)*(-1);  // y=-25*x+600
        MemberShip=MemberShip+600;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=Zero;
        DegreeOfError[2]=NotADegree;
        DegreeOfError[3]=NotADegree;      }
else if (NewError < ErrorEndOfZero) {
        MemberShip=0;
        MemberShip=(25*NewError)*(-1);
        MemberShip=MemberShip+600;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=Zero;
```

```
            MemberShip=0;
            MemberShip=100*NewError;
            MemberShip=MemberShip/3-733;
            DegreeOfError[2]=MemberShip;
            DegreeOfError[3]=PK;                    }

    else if (NewError < ErrorTopOfPK) {
            MemberShip=0;
            MemberShip=100*NewError;
            MemberShip=MemberShip/3-733;
            DegreeOfError[0]=MemberShip;
            DegreeOfError[1]=PK;
            DegreeOfError[2]=NotADegree;
            DegreeOfError[3]=NotADegree;        }
    else if (NewError < ErrorStartOfPB) {
            MemberShip=0;
            MemberShip=(100*NewError)*(-1);
            MemberShip=MemberShip/3+933;
            DegreeOfError[0]=MemberShip;
            DegreeOfError[1]=PK;
            DegreeOfError[2]=NotADegree;
            DegreeOfError[3]=NotADegree;        }
    else if (NewError < ErrorEndOfPK) {
            MemberShip=0;
            MemberShip=(100*NewError)*(-1);
            MemberShip=MemberShip/3+933;
            DegreeOfError[0]=MemberShip;
            DegreeOfError[1]=PK;


            MemberShip=0;
            MemberShip=25*NewError;
            MemberShip=MemberShip-650;
            DegreeOfError[2]=MemberShip;
            DegreeOfError[3]=PB;                            }
    else if (NewError < ErrorTopOfPB) {
```

```
        MemberShip=0;
        MemberShip=25*NewError;
        MemberShip=MemberShip-650;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=PB;
        DegreeOfError[2]=NotADegree;
        DegreeOfError[3]=NotADegree;    }
else if (NewError >= ErrorTopOfPB)      {
        DegreeOfError[0]=ResolutionOfDegree;
        DegreeOfError[1]=PB;
        DegreeOfError[2]=NotADegree;
        DegreeOfError[3]=NotADegree;    }
}
/*====================================================================
Fonksiyon Adi: FuzzificationOfDeltaError
Açiklama:    DeltaError degerlerini bulaniklastirir.
degerler milisantigrad
====================================================================*/
void FuzzificationOfDeltaError (void)
{
signed int MemberShipDeltaError;
MemberShipDeltaError=0;

if (DeltaError < DeltaErrorTopOfNB){
        DegreeOfDeltaError[0]=ResolutionOfDegree;
        DegreeOfDeltaError[1]=NB;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;}
else if (DeltaError < DeltaErrorEndOfNK){
        MemberShipDeltaError=0;
        MemberShipDeltaError=(25*NewError)*(-1);
        MemberShipDeltaError=MemberShipDeltaError+350;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=NB;
        DegreeOfDeltaError[2]=NotADegree;
```

```
        DegreeOfDeltaError[3]=NotADegree;}
else if (DeltaError < DeltaErrorStartOfNB){
        MemberShipDeltaError=0;
        MemberShipDeltaError=(25*NewError)*(-1);
        MemberShipDeltaError=MemberShipDeltaError+350;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=NB;

        MemberShipDeltaError=0;
        MemberShipDeltaError=100*NewError;
        MemberShipDeltaError=MemberShipDeltaError-1200;
        DegreeOfDeltaError[2]=MemberShipDeltaError;
        DegreeOfDeltaError[3]=NK; }
else if (DeltaError < DeltaErrorTopOfNK){
        MemberShipDeltaError=0;
        MemberShipDeltaError=100*NewError;
        MemberShipDeltaError=MemberShipDeltaError-1200;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=NK;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;       }
else if (DeltaError < DeltaErrorStartOfZero){
        MemberShipDeltaError=0;
        MemberShipDeltaError=(100*NewError)*(-1);
        MemberShipDeltaError=MemberShipDeltaError/3+600;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=NK;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;       }
else if (DeltaError < DeltaErrorStartOfNK){
        MemberShipDeltaError=0;
        MemberShipDeltaError=(100*NewError)*(-1);
        MemberShipDeltaError=MemberShipDeltaError/3+600;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=NK;
```

```
        MemberShipDeltaError=0;
        MemberShipDeltaError=25*NewError;
        MemberShipDeltaError=MemberShipDeltaError-400;
        DegreeOfDeltaError[2]=MemberShipDeltaError;
        DegreeOfDeltaError[3]=Zero;                }
else if (DeltaError < DeltaErrorTopOfZero){
        MemberShipDeltaError=0;
        MemberShipDeltaError=25*NewError;
        MemberShipDeltaError=MemberShipDeltaError-400;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=Zero;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;        }
else if (DeltaError < DeltaErrorStartOfPK){
        MemberShipDeltaError=0;
        MemberShipDeltaError=(25*NewError)*(-1);
        MemberShipDeltaError=MemberShipDeltaError+600;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=Zero;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;        }
else if (DeltaError < DeltaErrorEndOfZero){
        MemberShipDeltaError=0;
        MemberShipDeltaError=(25*NewError)*(-1);
        MemberShipDeltaError=MemberShipDeltaError+600;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=Zero;


        MemberShipDeltaError=0;
        MemberShipDeltaError=100*NewError;
        MemberShipDeltaError=MemberShipDeltaError/3-633;
        DegreeOfDeltaError[2]=MemberShipDeltaError;
        DegreeOfDeltaError[3]=PK; }

else if (DeltaError < DeltaErrorTopOfPK){
        MemberShipDeltaError=0;
```

```
        MemberShipDeltaError=100*NewError;
        MemberShipDeltaError=MemberShipDeltaError/3-633;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=PK;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;       }
else if (DeltaError < DeltaErrorStartOfPB){
        MemberShipDeltaError=0;
        MemberShipDeltaError=(100*NewError)*(-1);
        MemberShipDeltaError=MemberShipDeltaError/3+933;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=PK;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;}
else if (DeltaError < DeltaErrorEndOfPK){
        MemberShipDeltaError=0;
        MemberShipDeltaError=(100*NewError)*(-1);
        MemberShipDeltaError=MemberShipDeltaError/3+933;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=PK;

        MemberShipDeltaError=0;
        MemberShipDeltaError=25*NewError;
        MemberShipDeltaError=MemberShipDeltaError-650;
        DegreeOfDeltaError[2]=MemberShipDeltaError;
        DegreeOfDeltaError[3]=PB;  }
else if (DeltaError < DeltaErrorTopOfPB){
        MemberShipDeltaError=0;
        MemberShipDeltaError=25*NewError;
        MemberShipDeltaError=MemberShipDeltaError-650;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=PB;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;}
else if (DeltaError >= DeltaErrorTopOfPB){
        DegreeOfDeltaError[0]=ResolutionOfDegree;
```

```
        DegreeOfDeltaError[1]=PB;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;}
}
```

3.4.1.1.3 Kural Tablosu

Hata ve hatanın değişimi için beşer üyelik fonksiyonundan toplam 25 kural oluşturulmuştur. Hata ve hatanın değişimi giriş olarak seçilmiş çıkış, çıkış için tanımlamış olan kendine ait üyelik fonksiyonu yardımı ile gerçek değer dönüştürülmüştür.

Çizelge 3.3te kural tablosu gösterilmektedir. Örneğin, hata Negatif Büyük, hatanın değişimi Pozitif küçük ise tabloda satır sütün işlemi yapılarak çıkış Negatif Küçük elde edilmiş olur.

Çizelge 3.3 Bulanık Mantık Kural Tablosu

		HATA				
		NB	NK	S	PK	PB
HATANIN DEĞİŞMİ	NB	S	PK	PK	PK	PB
	NK	NK	S	PK	PK	PB
	S	NK	NK	S	PK	PK
	PK	NK	NK	NK	S	PK
	PB	NB	NK	NK	NK	S

3.4.1.1.4 Durulama Birimi

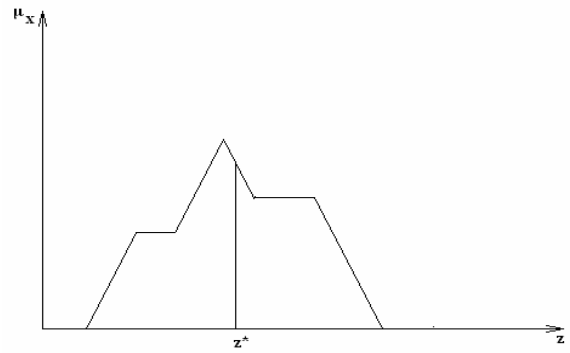
Karar merci tarafında hata ve hatanın değişimi için elde edilmiş olan bulanık ifadelerin kural tablosuna dayanarak işleme koyulmasının sonunda tekrar bir bulanık ifade elde edilir. Bu bulanık ifadeyi gerçek, kesin bir değere dönüştürme işlemine durulama denir. Durulama işlemi için dört çeşit yöntemden biri seçilebilir. Bu yöntemler maksimum üyelik yöntemi, ağırlık merkezi yöntemi, ağırlık ortalaması yöntemi, mean max üyelik yöntemidir.

3.4.1.1.4.1 Ağırlık Merkezi Yöntemi

Ağırlık merkezi yöntemi en yaygın olarak kullanılan duruluma yöntemidir. Gerçek çıkış değeri, alanın merkezi ve uygulanan bulanık kümenin alanı ile aşağıdaki formüle göre hesaplanır [13]

Şu formül ile ifade edilir

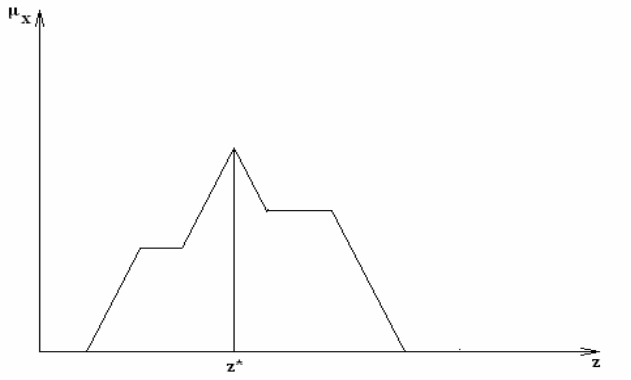
$$z^* = \frac{\int \mu_c(z) z dz}{\int \mu_C(z) sz} \qquad (3.6)\ [14]$$



Şekil 3.25 Ağırlık merkezi yöntemi

3.4.1.1.4.2 Maksimum Üyelik Merkezi Yöntemi

Yükseklik yöntemi olarak da adlandırılmaktadır. Bütün üyelik dereceleri içinde en büyük olana eşittir.

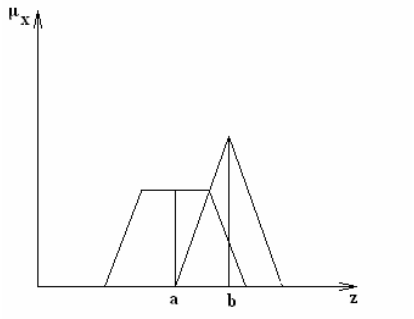


Şekil 3.26 Maksimum Üyelik Merkezi

$$\mu_C(z^*) \geq \mu_C(z), z \in Z \qquad (3.7)$$

3.4.1.1.4.3 Ağırlık Ortalaması Yöntemi

Girişlerden elde edilen bütün bulanık değerler ile üyelik değerleri kullanılarak durulama yapılmaktadır [13].



Şekil 3.27 Ağırlık ortalaması yöntemi

Şu formül ile ifade edilir:

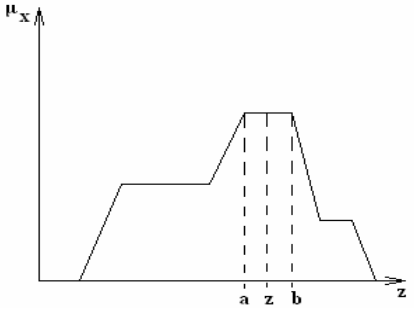
$$z^* = \frac{\sum \mu_C(\overline{z}).\overline{z}}{\sum \mu_C(\overline{z})} \tag{3.8}$$

3.4.1.1.4.4 Mean-Max Yöntemi

Maksimum üyelik derecesi tek bir nokta olmayan, düz olan sistemlerde kullanılır.

Şu formül ile ifade edilir;

$$z^* = \frac{a+b}{2} \tag{3.9}$$



Şekil 3.28 Mean-Max yöntemi

Tasarlanan sistemde kullanılan durulama yöntemi, Ağırlık ortalaması yöntemi olarak uygulanmıştır. Bu şekilde elde edilen çıkış gerçek değeri uygulamada kullanılan sürme devresine PWM aktif oma süresi olarak aktarılmıştır.

3.4.1.2 Mikrodenetleyici Üzerinde Yazılan Bulanık Mantık C Kodu

Aşağıda mikrodenetleyici üzerine yazılmış olan fuzzy kodu ve alt programları bulunmaktadır. Bu program mirodenetleyici üzerinde sürekli olarak koşmamaktadır. Tasarımcı tarafından gerekli görülen sürede bir bulanık mantık algoritması koşturmaktadır. Bulanık mantık algoritmasının sürekli olarak koşmamasının sebebi, sistemde sıcaklık kontrol edilmesidir. Sıcaklığın ani değişimlerde bulunamamasından dolayı sürekli olarak bu kod çalıştırıldığında optimum kontrol yapılamayacaktır. Kullanılan ısıtıcı, soğutucu ve odanın ısı alış-veriş durumuna göre bu süre değişiklik gösterebilir. Aşağıda mikrodenetleyici üzerine yazılmış olan bulanık mantık algoritmasını göreceksiniz.

```
/*==================================================================
Fonksiyon Adi: Fuzzy
Açiklama:     Fuzzy Algoritmasının çalışması
==================================================================*/
                CalculateError();
                CalculateDeltaError();
                FuzzificationOfError();
                FuzzificationOfDeltaError();
                DecisionCycle();
                Fuzzy_CripValue=Defuzzificaiton();

                if (Fuzzy_CripValue > 0)
                        {
                        Heater=1*((unsigned char)Fuzzy_CripValue);
                        Cooler=0;
                        Cooler_Out=1;
                        }
                else if (Fuzzy_CripValue < 0)
                        {
                        Fuzzy_CripValue=(-1)*Fuzzy_CripValue;
                        Cooler=1*((unsigned char)Fuzzy_CripValue);
```

```
                    Heater=0;
                    Heater_Out=1;
                    }

/*=====================================================================
Fonksiyon Adi: CalculateError
Açiklama:   Kontrol algoritmasi için gerekli hatayi bulur.
                    -Sicaklik degerleri düzeltilip LastTemperature, PreviousTemperature
                    içerisine koyulmuş olmalidir.
                    -Hata=Set-Ölçülen
Cikis:  NewError ve OldError  degerler milisantigrad
=====================================================================*/
void CalculateRoomTemperature (void){
unsigned char AvaregeOfTMPs;

AvaregeOfTMPs=TMP35_1;
AvaregeOfTMPs=AvaregeOfTMPs+TMP35_2;
AvaregeOfTMPs=AvaregeOfTMPs+TMP35_3;
AvaregeOfTMPs=AvaregeOfTMPs/3;
RoomTemperature=AvaregeOfTMPs;}
/*=====================================================================
Fonksiyon Adi: CalculateError
Açiklama:   Kontrol algoritmasi için gerekli hatayi bulur.
                    -Sicaklik degerleri düzeltilip LastTemperature, PreviousTemperature
içerisine koyulmuş olmalidir.
                    -Hata=Set-Ölçülen
Cikis:  NewError ve OldError  degerler milisantigrad
=====================================================================*/
void CalculateError (void){
OldError=NewError;
NewError=SetTemperature-RoomTemperature;
NewError=SetTemperature;
}
/*=====================================================================
Fonksiyon Adi: CalculateDeltaError
```

```
Açiklama:   Kontrol algoritmasi için gerekli hatanin degisimini bulur.
                                degerler milisantigrad
==========================================================================*/
void CalculateDeltaError (void)
{ DeltaError=(NewError-OldError);
 DeltaError=RoomTemperature; }
/*==========================================================================
Fonksiyon Adi:  DecisionCycle
Açiklama:       BUlanklastirilmis degerleri tabloya göre bulanık sonuç verir.
degerler Bulaniktir.
==========================================================================*/
void DecisionCycle (void){
unsigned char Error;
unsigned char DeltaError;
Error=DegreeOfError[1];
DeltaError=DegreeOfDeltaError[1];
Decisions[0]=MakeDecisionFromTable(Error,DeltaError);

Error=DegreeOfError[0];
DeltaError=DegreeOfDeltaError[0];
Decisions[1]=MaxDotFunction(Error,DeltaError);

Error=DegreeOfError[1];
DeltaError=DegreeOfDeltaError[3];
Decisions[2]=MakeDecisionFromTable(Error,DeltaError);
Error=DegreeOfError[0];
DeltaError=DegreeOfDeltaError[2];
Decisions[3]=MaxDotFunction(Error,DeltaError);

Error=DegreeOfError[3];
DeltaError=DegreeOfDeltaError[1];
Decisions[4]=MakeDecisionFromTable(Error,DeltaError);
Error=DegreeOfError[2];
DeltaError=DegreeOfDeltaError[0];
Decisions[5]=MaxDotFunction(Error,DeltaError);
```

```
Error=DegreeOfError[3];
DeltaError=DegreeOfDeltaError[3];
Decisions[6]=MakeDecisionFromTable(Error,DeltaError);

Error=DegreeOfError[2];
DeltaError=DegreeOfDeltaError[2];
Decisions[7]=MaxDotFunction(Error,DeltaError);  }

 /*====================================================================
Fonksiyon Adi:  MakeDecisionFromTable
Açiklama:       Girilen Error ve Delta Error'e göre tabloda karar çikartir.
degerler Bulaniktir.                s
====================================================================*/
unsigned char MakeDecisionFromTable (unsigned char Error,unsigned char DeltaError){
unsigned char Decision;

Decision=0;


if ((DeltaError==NotADegree)||(Error==NotADegree)){
        Decision=NotADegree;}
else{
        if ((DeltaError)==NB){
                Decision=RuleLine1[Error];}
        else if ((DeltaError)==NK){
                Decision=RuleLine2[Error];  }
        else if ((DeltaError)==Zero){
                Decision=RuleLine3[Error];}
        else if ((DeltaError)==PK){
                Decision=RuleLine4[Error];}
        else if ((DeltaError)==PB){
                Decision=RuleLine5[Error];}
        }
return Decision;
```

```
}
/*==========================================================================
Fonksiyon Adi:  MaxDotFunction
Açiklama:       u degerleri içinden büyük olani alir.
degerler Bulaniktir.
==========================================================================*/
unsigned char MaxDotFunction (unsigned char Error,unsigned char DeltaError){
unsigned char MaxDot;

if ((DeltaError==NotADegree)||(Error==NotADegree)){
        MaxDot=NotADegree;}
else{
         if (DeltaError>Error)
                {MaxDot=DeltaError;}
        else
                {MaxDot=Error;}
        }
return MaxDot;
}

/*==========================================================================
Fonksiyon Adi:  Defuzzification
Açiklama:       Bulanik degerleri kesin degerlere dönüştürür.
                                Agırlıklı Ortalama Methodu ile dönüsüm yapilmistir.
==========================================================================*/
signed char Defuzzificaiton (void){
signed int Mul1;
signed int Mul2;
signed int Mul3;
signed int Mul4;
signed int Crisp;
signed int Addition;

signed int Center;
Center=0;
```

```
Crisp=0;
Mul1=0;
Mul2=0;
Mul3=0;
Mul4=0;

if      (Decisions[0]!=NotADegree){
        Center=FindCenterOfMembershipFunction(Decisions[0]);
        Mul1=Center * Decisions[1];}
else
        {Mul1=0;}

if      (Decisions[2]!=NotADegree)
        { Center=FindCenterOfMembershipFunction(Decisions[2]);
          Mul2=Center * Decisions[3]; }
else{Mul2=0;}

if      (Decisions[4]!=NotADegree){
        Center=FindCenterOfMembershipFunction(Decisions[4]);
        Mul3=Center * Decisions[5]; }
else{Mul3=0;}

if      (Decisions[7]!=NotADegree){
        Center=FindCenterOfMembershipFunction(Decisions[6]);
        Mul4=Center * Decisions[7];}
else {Mul4=0;}
1.  Center=Mul1+Mul2;
Center=Center+Mul3;
Center=Center+Mul4;


Addition=0;
if (Decisions[1]!=NotADegree){Addition=Decisions[1];}
if (Decisions[3]!=NotADegree){Addition=Addition+Decisions[3];}
if (Decisions[5]!=NotADegree){Addition=Addition+Decisions[5];}
```

```
if (Decisions[7]!=NotADegree){Addition=Addition+Decisions[7];}

Crisp=Center/Addition;

return Crisp;
}
/*==================================================================
Fonksiyon Adi: FindCenterOfMembershipFunction
Açiklama:      Bulanik degerleri kesin degerlere dönüştürür.
                  Agirlikli Ortalama Methodu ile dönüsüm yapilmistir.
==================================================================*/
signed char FindCenterOfMembershipFunction (unsigned char MembershipFunction){
unsigned char CenterOfMembership;

CenterOfMembership=0;
if (MembershipFunction==NB)
        {CenterOfMembership=-90;}
else if  (MembershipFunction==NK)
        {CenterOfMembership=-50;}
else if  (MembershipFunction==Zero)
        {CenterOfMembership=0;}
else if  (MembershipFunction==PK)
        {CenterOfMembership=50;}
else if  (MembershipFunction==PB)
        {CenterOfMembership=90;}
        return CenterOfMembership;
}
```

3.4.2 Aydınlatma

Bir ortamı ve içerisinde nesneleri, istenile ölçütlerde görsel algılamaya uygun kılacak şekilde tasarlanmış ışık uygulamaları ”aydınlatma” olarak tanımlanır. (Aydınlatma Tasarımı ve Proje Uygulalamarı A.Ünal 2004)

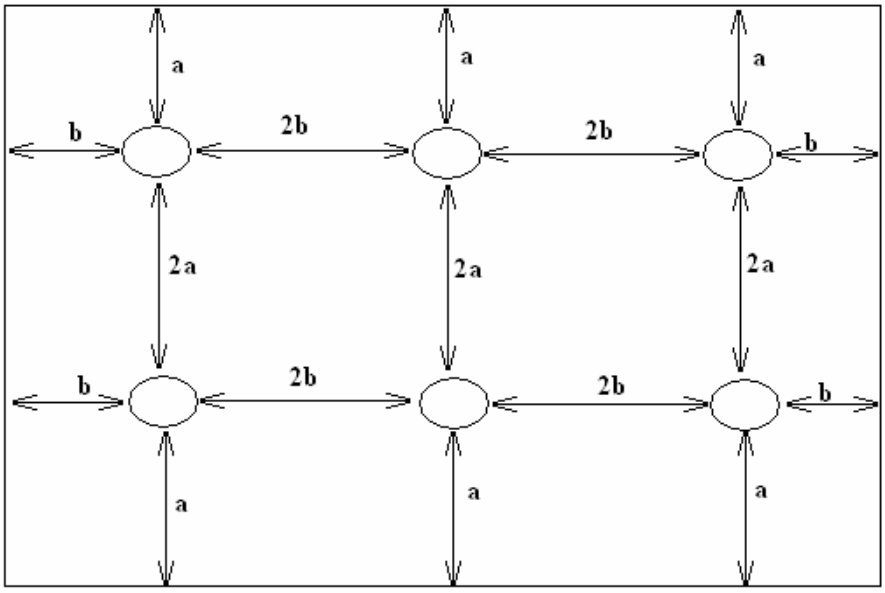
Aydınlatma tasarımlarındaki görsel konfor ölçütleri aydınlatmanın niteliği, aydınlatmanın niceliği, ışıklık ve yüzey özellikleridir. Aydınlatmanın niteliğinde aydınlığın niteliği, ışığını

rengini, renksel geri verimi, aydınlık düzeyinin dağılımını ve gölge konularını içeren geniş bir kavramdır. Aydınlık yüzeyi dağılımı için ışık kaynakları tasarlanan sistemde optimum olacak şekilde yerleştirilmiştir.

Aydınlatılan mekanın her noktasında Aydınlık seviyesinin elde edilmesi oldukça zordur. Bir oda içerisindeki maksimum, ortalama ve minimum aydınlık seviyeleri arasındaki büyük farklar ışık kaynaklarının yanlış konumlandırılmasından yada uygulama için uygun olmayan ışık dağılım özelliğine sahip aygıtların kullanılmasından kaynaklanır. Oda içerisindeki E-min/E-ortalama değeri aydınlık seviyesi dağılım faktörü olarak tanımlanır. Bu değer aydınlatmaya olan ihtiyacın düşük olduğu yerlerde 0.4 yüksek olduğu yerlerde ise 0.66 değerlerinden düşük olmamalıdır.

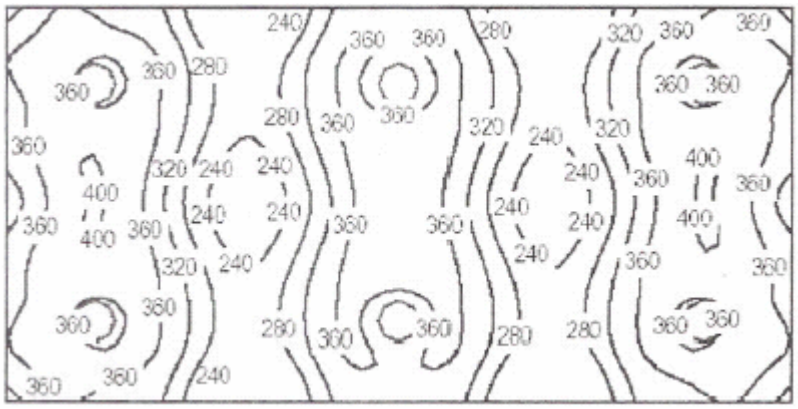
Işık kaynakları aydınlatacakları hacim içersinde belli kuralar doğrultusunda yerleştirilirler. Hacim içerisinde yanlış yerleştirilmiş ışık kaynakları kamaşma gölge oluşumu veya ışık dağılımının boşa harcanması gibi sorunlara yol açabilir.

Tasarlanan sistemde şekil 3.29 Deki gibi ışık kaynağı yerleşimi yapılmıştır.



Şekil 3.29 Işık kaynaklarının yerleşimi

Bu durumda aydınlık seviyesi dağılımı şekil 3.30 daki örnek gibi olur.



Şekil 3.30 Örnek aydınlık seviyesi dağılımı

3.4.2.1 P Kontrol Algoritması

K kazançlı kontrol işleminde, kontrol işareti kontrolör çıkışına sabit bir oran ile aktarıldığından, oransal kontrol olarak adlandırılır. Sezgisel olarak, oransal işleve ilave olarak, giriş işaretinin türevinden veya integralinden yararlanılabileceği düşünülebilir. Buna göre içinde toplayıcı (Toplama veya çıkartma) kuvvetlendirici, zayıflatıcı, türev ve integral alıcı elemanlar bulunan, daha genel sürekli bir kontrolör göz önünde bulundurulabilir. Tasarımcının görevi bu elemanlardan hangilerinin, hangi oranda ve şekilde bağlanarak kullanılması gerektiğini belirlemektir [15].

Oransal kontrol için P kontrolü (P=Proportional) integral kontrolü için I kontrolü (I=Integral), Türev kontrolü için D kontrolü (D=derivative) gerçekleştirilebilir.

Tasarlanan sistemde P kontrolü gerçekleştirilmiştir. Sistemde her bir ışık grubu için giriş olarak LDR bilgisi alınır. Kullanıcı tarafından girilen set değer ile LDR bilgisi karşılaştırılarak, ışık kaynağı grubunun çıkışı sürülür. Çıkış bilgisi LDR bilgisi ile set değeri arasındaki farkın belirli bir değer ile güçlendirilerek çıkışa verilmesi ile sağlanır.

Tasarlanan sistemde ışık yoğunluğunun her noktada aynı olması için odanın içerisinde üç farklı noktaya ışık sensörü olarak kullanılan LDRler yerleştirilmiştir. Bu LDRlerden gelen bilgi ışığında mikrodenetleyici o LDR ile ilgili olan ışık kaynağı olarak kullanılan led grubunu sürer. Bu şekilde odanın içerisinde istenilen aydınlık düzeyine yakın bir aydınlık yüzeyi odanın her noktasında sağlanmış olur.

3.4.3 Güvenlik

Sistemde yapılan güvenlik kontrolünde odanın penceresinde bulunan rit roleden gelen sinyal koptuğunda, yani pencere açıldığında eğer oda aktif edilmemiş ise alarmın bağlı olduğu role

çekilir. Mikrodenetleyicinin dış kesmesine bağlı olan rit role için yazılan C yazılımı aşağıdaki gibidir.

```
void ExtInt (void) interrupt 0 using 1 {
        if (Flag_RoomActive==1)
                { Relay_Out=0;
                  Flag_RoomActive=0;}
            else
                {Relay_Out=1;
                Flag_RoomActive=1;}
                }
```

3.4.4 Bilgisayar Ara Yüzü

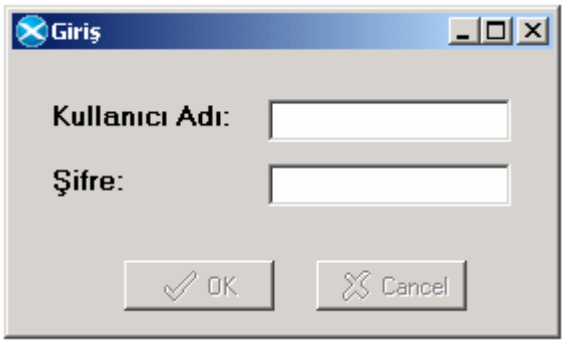
Tasarlanan ev otomasyonu sisteminde bilgisayar ara yüzü ile kullanıcıya bilgisayar üzerinden erişimi sağlamak hedeflenmiştir. Bilgisayar ara yüzü daha önce de bahsettiğimiz gibi RS232 protokolü üzerinde odada bulunan elektronik sistemde ile haberleşmektedir.

Malikane.exe olarak adlandırılan bilgisayar yazılımı Delphi görsel dilinde yazılmıştır. Malikane programı sayesinde kullanıcılar bağlı oldukları odaların sıcaklık değerleri, ışık değerleri, odanın aktif olup olmadığı, ısıtıcı veya soğutucunun hangisinin aktif olduğu gibi parametreleri görebilirler. Ayrıca bağlı oldukları odanın sıcaklık-zaman değişim grafiğini de görebilirler.

Şekil 3.31 Bilgisayar yazılımı ikonu

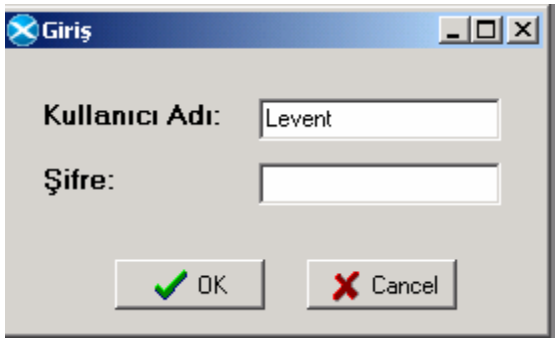
Bilgisayar yazılımına giriş için her kullanıcının bir şifreye ihtiyacı vardır. Her kullanıcıya verilen şifre ile kullanıcılar programa ulaşabilmektedirler.

Kullanıcı malikane yazılımını çalıştırdığı zaman karşısına "Giriş" başlığında bir pencere açılmaktadır. Bu pencerede herhangi bir şifre veya kullanıcı adı girilmediği sürece hiçbir buton aktif olmaz ve kullanıcının bilgileri girmesi veya programı kapatması beklenir.



Şekil 3.32 Bilgisayar yazılımı giriş penceresi pasif

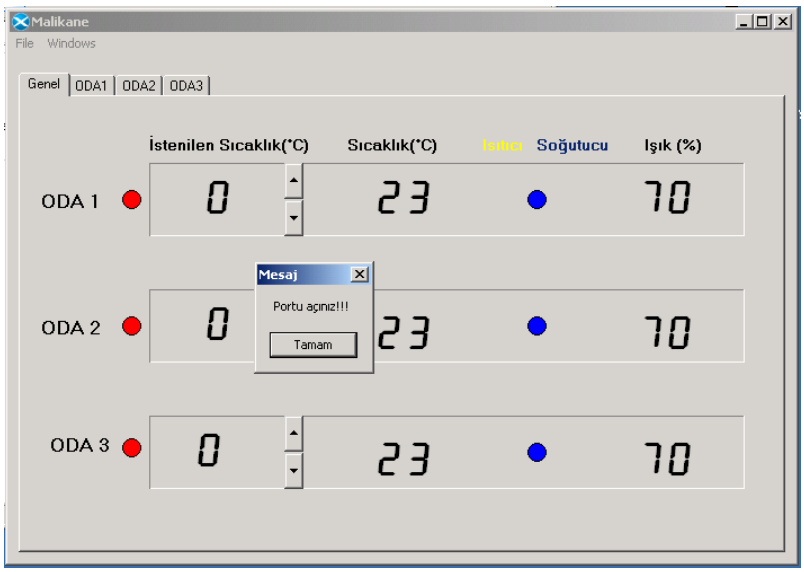
Kullanıcı şifre ve kullanıcı adı bilgilerinin girmeye başladığı zaman "OK" ve "Cancel" butonları aktif hale gelir.



Şekil 3.33 Bilgisayar yazılımı giriş penceresi aktif

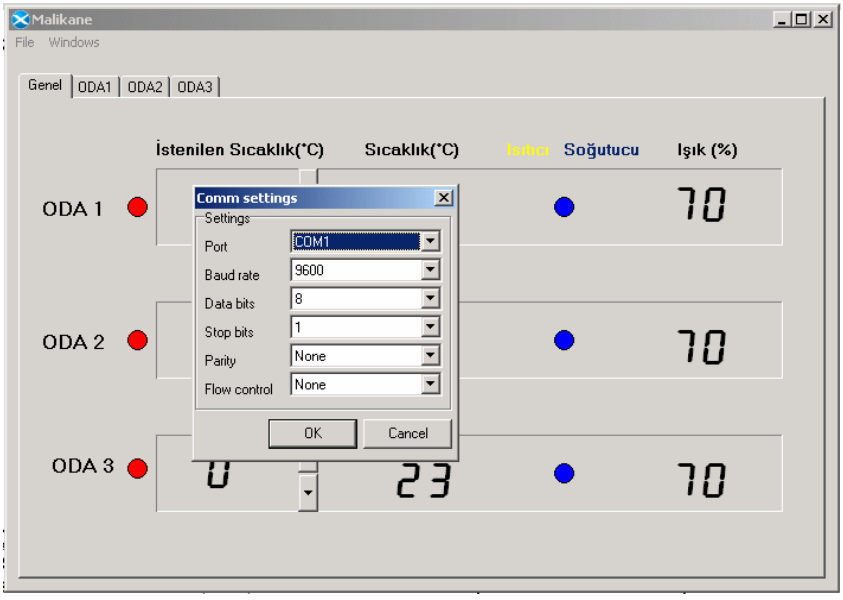
Kullanıcı şifresinin ve kullanıcı adı doğru girdikten sonra "OK" tuşuna basarak programı çalıştır.

Program açıldığında kullanıcının karşısına "Malikane" olarak adlandırılmış bir pencere ile karşılaşır. Program açıldıktan bir saniye sonra bilgisayar yazılımı kullanıcıya bilgisayar ile kullanılan modül arasındaki iletişim açılması için "Portu Açınız!!!" şeklinde bir mesaj çıkarır.



Şekil 3.34 Bilgisayar yazılımı Com Port bağlantı uyarı penceresi

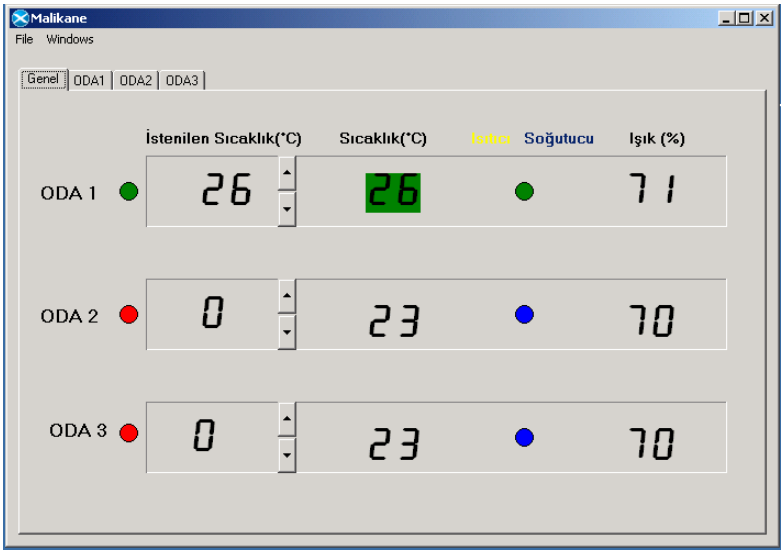
Çıkan bu mesaj için "Tamam" butonuna tıklanması veya kapatılmak istenmesinde Malikane yazılımı kullanıcıya "Comm Setting" başlığında elektronik kart ile iletişim için kullanılacak olan bağlantı noktasını ve bağlantı ayarlarının yapıldığı pencereyi açar.



Şekil 3.35 Bilgisayar yazılımı Com Port bağlantı ayar penceresi

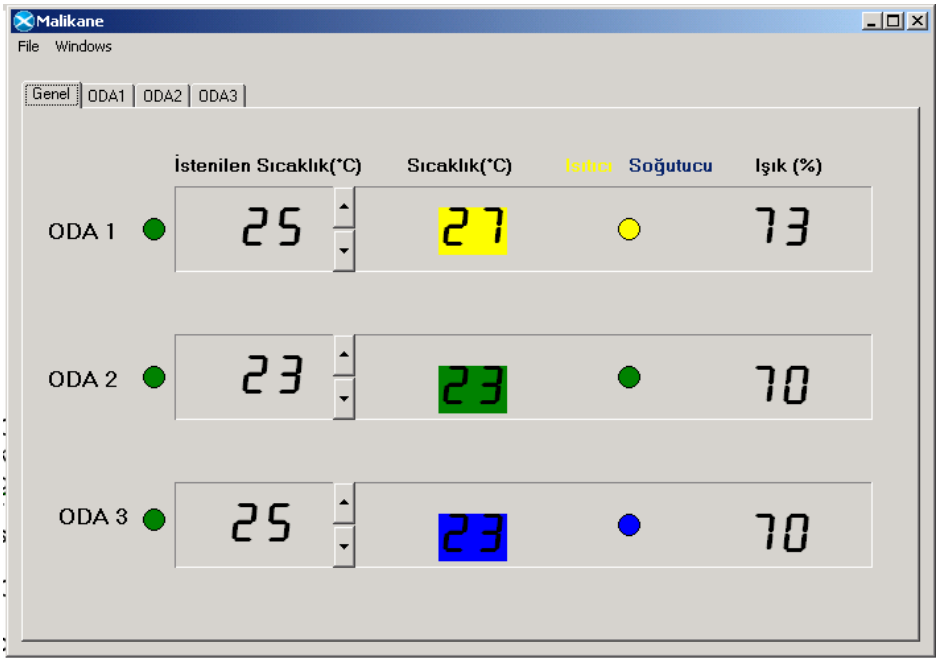
Comm Settings penceresinden kullanıcı tasarlanmış olan elektronik kart ile iletişim için kullanılan Portu şetçikten sonra, haberleşme ayarlarını 9600 Baud rate, 8 data bits, 1 Stop Bit, None Parity ve None Flow Control olarak ayarlaması gerekir. İletişim ayarlarını kullanıcının değiştirmesine gerek yoktur. Yazılım açıldığında iletişim ayarları istenilen şekilde gelmektedir. Kullanıcı sadece ilgili portu seçmesi yeterli olacaktır.

Comm Settings penceresinden kullanıcı ilgili ayarları yapıp, “OK” butonu ile çıktığında malikane yazılımı sistem için tasarlanmış olan elektronik karta sorguda bulunup ilgili parametreleri alarak ilgili gösterge panellerini yeniler.



Şekil 3.36 Bilgisayar yazılımı genel penceresinin tek oda ile görünümü

Genel penceresinden Şekil 3.36 da görüldüğü gibi ODA 1 den alınan bilgiler kullanıcıya sunulmaktadır. Genel penceresinde ODA 1, ODA2 ve ODA 3 gözükmektedir. ODA 1ile iletişim gerçekleştiği için ODA 1 önündeki simge yeşil renkte, ODA 2 ve ODA 3 ile haberleşilemediği için ODA 2 ve ODA 3 önündeki simge kırmızı renktedir.

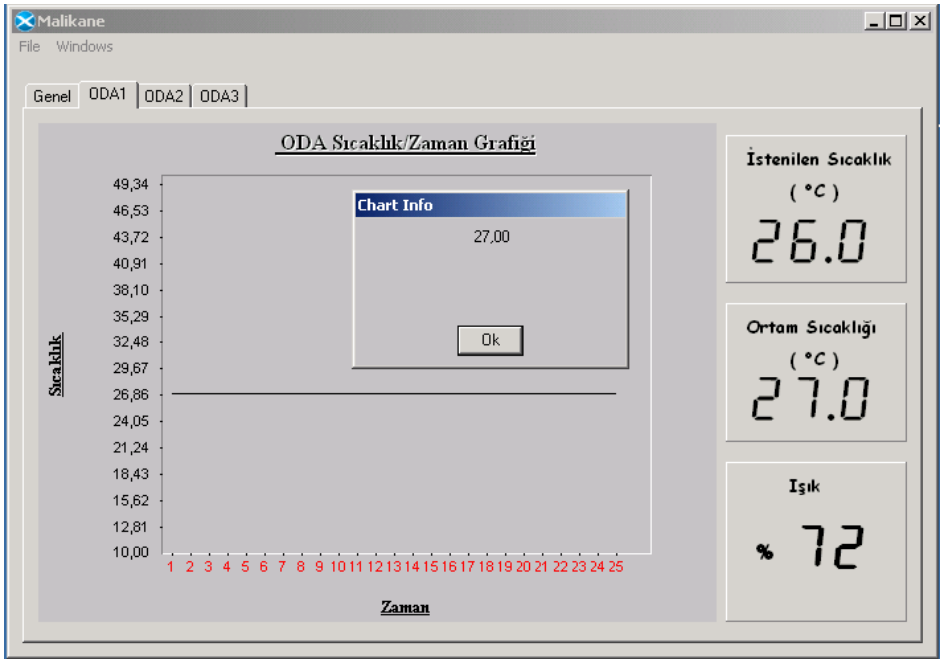


Şekil 3.37 Bilgisayar yazılımı genel penceresinin üç oda ile görünümü

Genel penceresinden üç farklı odanın istenilen sıcaklığı, oda sıcaklığı, o an ısıtıcının veya soğutucunun çalıştığını ve ışıl değeri gözükmektedir. Sıcaklık başlığı altında gösterilen odanın anlık sıcaklık değeri ve ısıtıcı/soğutucu başlığı altındaki simge eğer o an ısıtıcı veya soğutucu çalışmıyorsa yeşil renkte, ısıtıcı çalışıyorsa sarı, soğutucu çalışıyorsa mavi renkte kullanıcıya gösterilir.

Kullanıcı Oda 1 pencere içerisinde seçerse, karşısına Şekil 3.38 deki gibi bir pencere açılacaktır. Bu pencerede Oda 1'in sıcaklık zaman grafiğini, set edilen sıcaklık değerini, anlık oda sıcaklığını ve ışık değeri rahatlıkla görecektir.

Oda Sıcaklık Zaman Grafiğinde kullanıcı değerini öğrenmek istediği bir grafik noktasının üzerine tıkladığı zaman "Chart Info" başlığı altında seçilen noktanın değeri gösterilir.



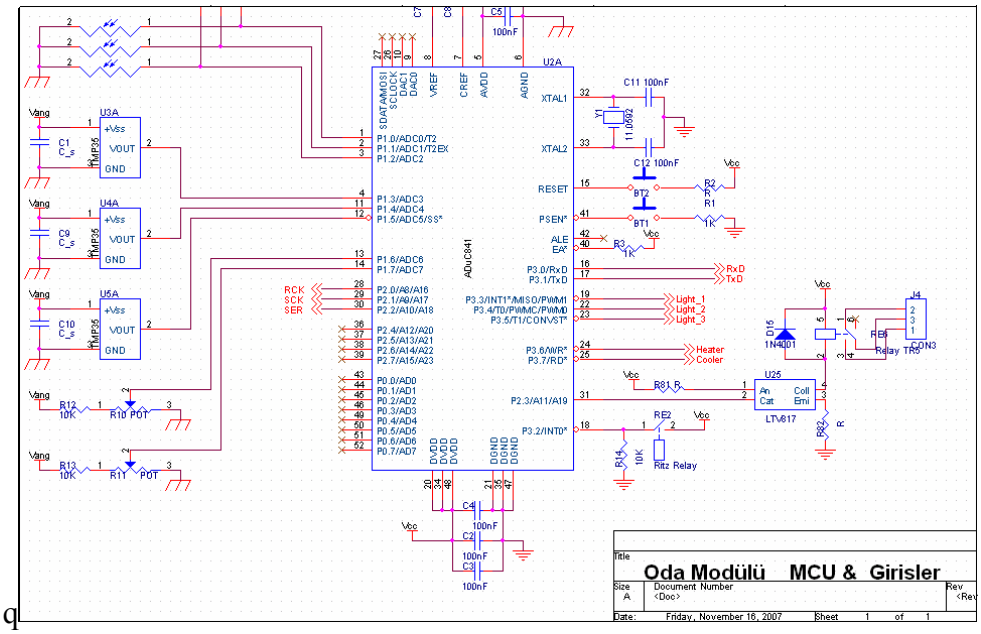
Şekil 3.38 Bilgisayar yazılımı genel penceresinin oda penceresi

Malikane yazılımı elektronik kart ile her saniyede bir haberleşmektedir. Bu şekilde odada kontrol edilen parametrelerdeki değişim bilgisayarda kullanıcının kontrolüne sunulmuş olmaktadır.

4. UYGULAMASI GERÇEKLEŞTİRİLEN EV OTOMASYON SİSTEMİ DONANIMI

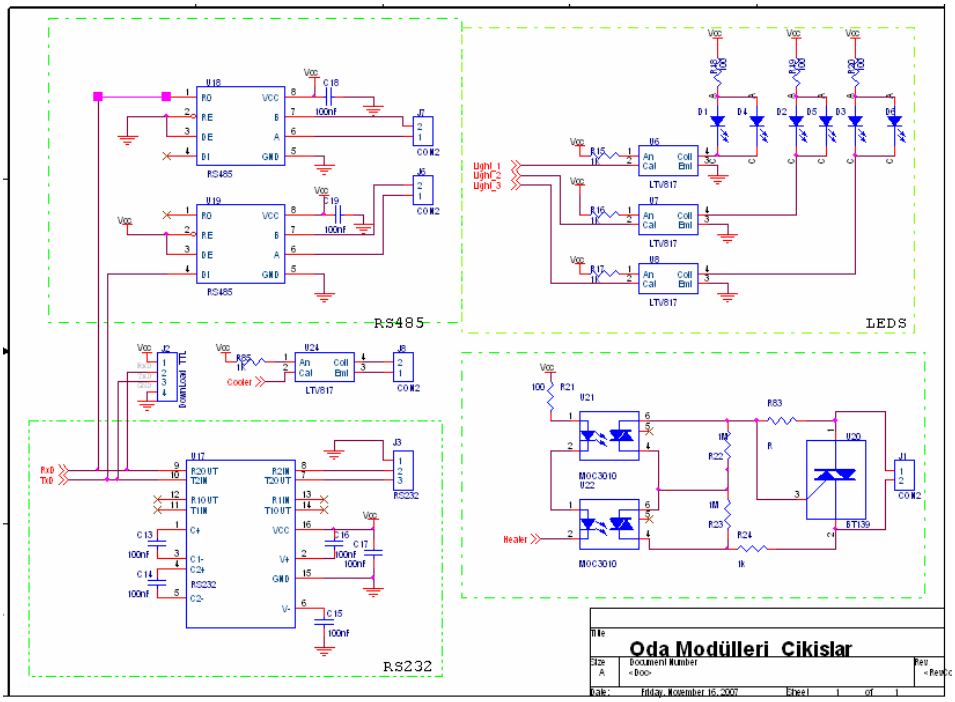
4.1 Şematik

4.1.1 Oda Modülü PCBsinin Mikrokontrolör ve Giriş Kısmının Şematiği



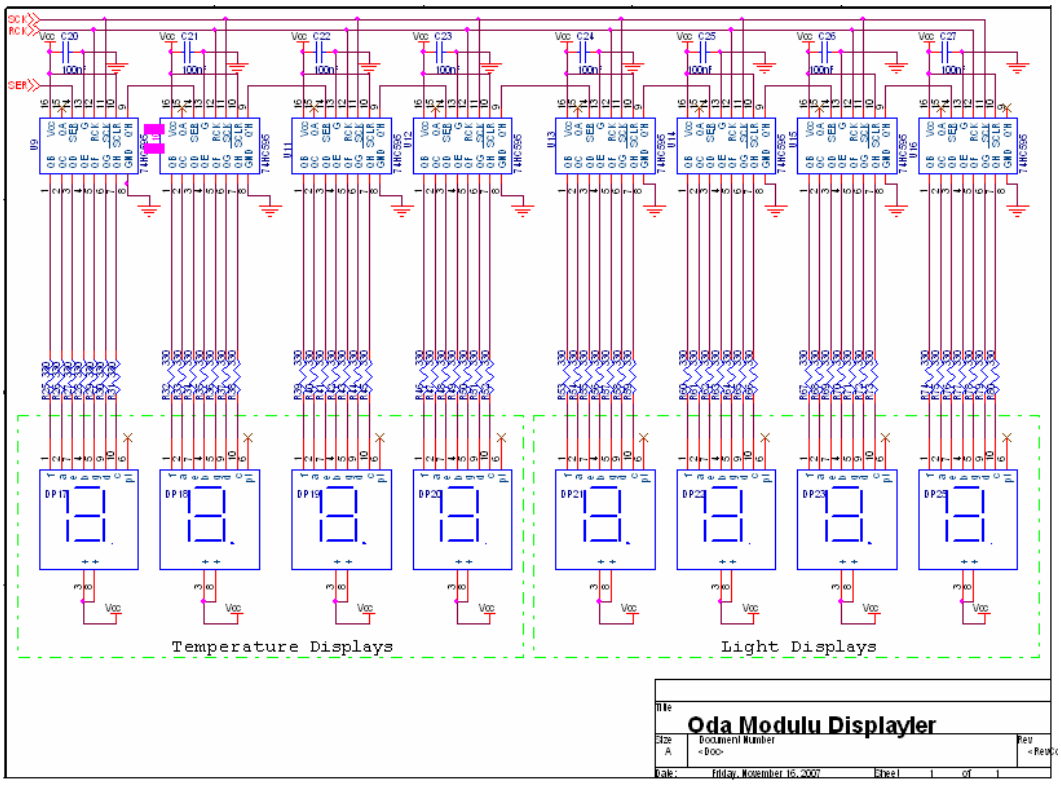
Şekil 4.1 Oda modülü PCBsinin mikrokontrollör ve giriş kısmının şematiği

4.1.2 Oda Modülü PCBsinin Çıkış Kısmının Şematiği



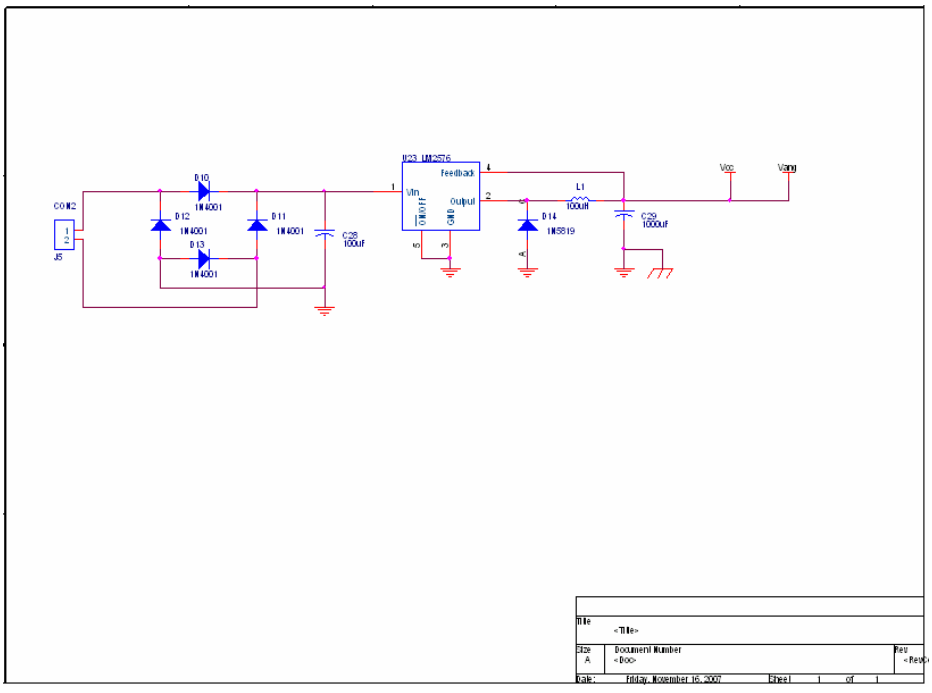
Şekil 4.2 Oda modülü PCBsinin çıkış kısmının şematiği

4.1.3 Oda Modülü PCBsinin Display Kısmının Şematiği



Şekil 4.3 Oda modülü PCBsinin display kısmının şematiği

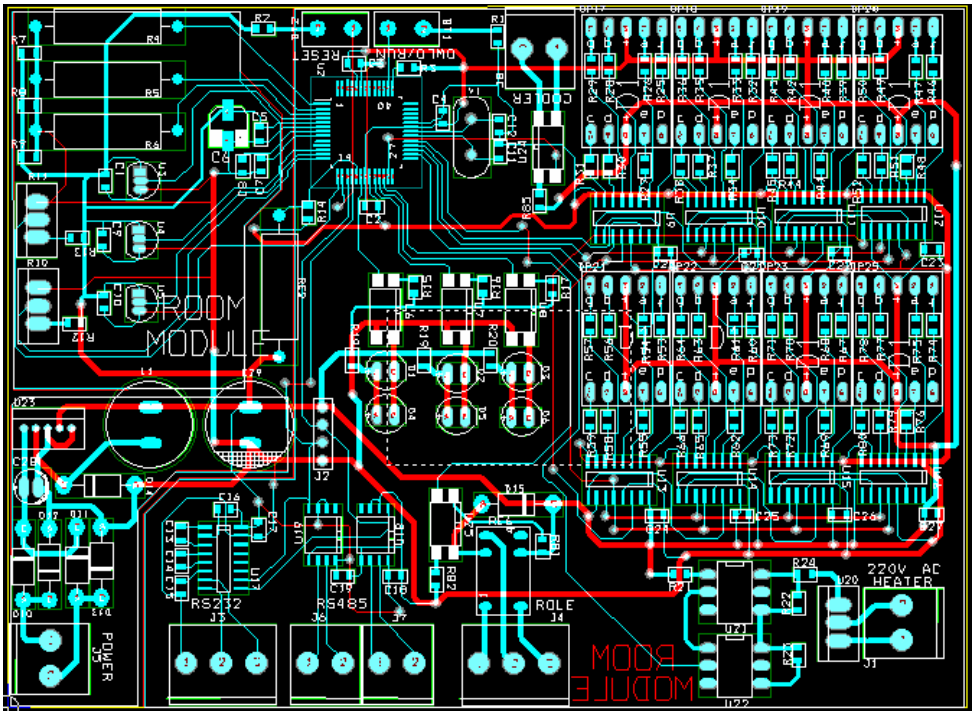
4.1.4 Oda Modülü PCBsinin Power Kısmının Şematiği



Şekil 4.4 Oda modülü PCBsinin power kısmının şematiği

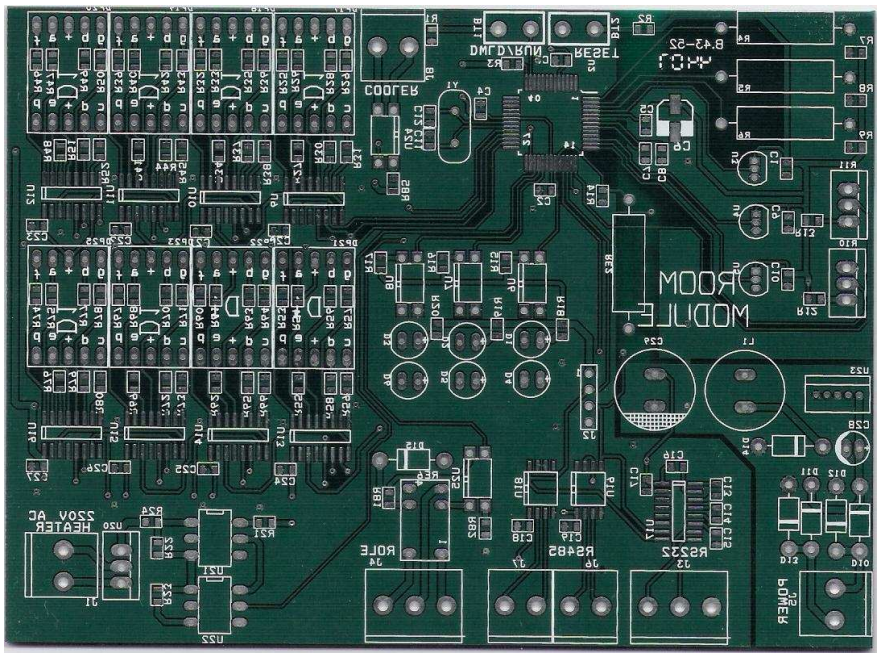
4.2 PCB

4.2.1 Oda Modülü PCBsinin OrCAD Çizim Görüntüsü



Şekil 4.5 Oda modülü OrCAD çizim görüntüsü

4.2.2 Oda Modülü PCBsinin Gerçek Görüntüsü



Şekil 4.6 Oda modülü PCBsinin gerçek görüntüsü

5. SONUÇ

Yapılan çalışmada mikrodenetleyici ve bilgisayar kullanılarak ev otomasyonu yapılmıştır. Ev otomasyonu da diğer otomasyon sistemleri gibi insana daha konforlu ve sistematik bir yaşam alanı oluşturmak için tasarlamıştır. Otomasyon sistemleri insan odaklıdır. Günümüzün değişen koşullarıyla teknoloji her alanda karşımıza çıkar olmuştur. Günlük yaşamımızda da eskiden lüks olarak görülen ve sadece büyük plazalarda uygulanan bina otomasyonu, ev otomasyonu olarak yerini almıştır. Artık, villalar başta olmak üzere büyük sitelerde ev otomasyonu bir standart haline gelmiştir.

Günlük oturduğumuz evlerde de ev otomasyon sistemleri sonradan eve entegre edilebilmektedir. Herhangi bir eve istenilen her türlü ev otomasyon seçeneği eklenebilir, Proxy kart ile giriş, kamera ile kontrol, otopark otomasyonu, elektronik eşyaların kontrolü (TV, çamaşır makinesi, kombi, aydınlatma vb.), güvenlik gibi.

Otomasyon sistemleri, kullanıcıların üzerindeki sorumlulukları birazda olsa azaltmakta, günlük stres hayatından uzak tutmaktadır. Ev otomasyonu ile güvenlik otomasyonu yapılan bir evin sahibi, evinde oluşabilecek herhangi bir hırsızlık olayı girişimi için kaygı duymamaktadır. Çünkü eğer evine yanlış yollardan girilmek istenildiğinde kendisine ve güvenlik görevlerine haber veren bir sistemin devrede olduğu bilerek günlük hayatını rahatlıkla sürdürebilir.

Bina otomasyonu ile kontrol edilen bir plaza veya apartmanın işletmeciliği (yöneticiliği) arasında günümüzde çok büyük farklar bulunmaktadır. Örnek olarak yirmi katlı bir plaza örnek alınırsa, otomasyon sistemi ile kontrol edilen bir plazada plaza yönetiminin en fazla 4–5 kişilik bir kontrol ekibine ihtiyaç duyulmaktadır. Oysa bina otomasyonu ile kontrol edilmeyen 20 katlı bir bina kontrolü için her katın büyüklüğüne göre 1–2 belki 3 kişiye ihtiyaç vardır. Bu da toplamda 60–70 kişilik bir teknik ekibe tekabül etmektedir. Oysa başlangıçta göze önüne alınabilecek bir otomasyon sistemi ile bina kontrolü kolaylaşır, enerji tasarrufu sağlanır, kullanım maliyetleri azalır ve modern bir yapı oluşturulmuş olur.

Tasarlanan ev otomasyon sisteminde öncelikle kullanıcının güvenliği ve konforu göz önüne alınmıştır. Sistemde evin içindeki sıcaklık, ışık ve güvenlik kontrolü gerçekleştirilmiştir.

Sistemde yapılan sıcaklık kontrolünde her oda ayrı ayrı ele alınmıştır. Yani her odanın ısısı kullanıcının o oda için arzu ettiği sıcaklık değerine ayarlanabilir. Bu sıcaklık değeri için mikrodenetleyici kendi içerisinde bulanık mantık algoritmasını uygulayarak, çıkış için gerekli olan ısıtma veya soğutma işlemini yapmaktadır. Bulanık mantık algoritmasının

kullanılmasındaki amaç insan diline en uygun algoritma olmasının yanında sistemin bir transfer fonksiyonun olmamasıdır.

Bulanık mantık ile yapılan sıcaklık kontrolünde ısıtmada ekonomik kazanç sağlanmış olur.

Bulanık mantık algoritması diğer klasik P,PI veya PID gibi kontrol algortimalarınan göre kesin değerler üzerinden çalışmamakta, bulanık ifadeler yani dilsel ifadeler ile işlemleri gerçekleştirmektedir. Bu nedenle kontrol edilen sistemi transfer fonksiyonuna çok fazla ihtiyaç duyulmaz.

Bulanık mantık kontrol algoritması sistemde her iki saniyede bir çalışmakta ve çıkış değerlerinin yenilemektedir. Bunun nedeni, model olarak alınan odanın sıcaklığının hemen değişmemesidir. Bu süre kullanılan ısıtıcı, soğutucunun gücüne, sıcaklığı kontrol edilen odanın büyüklüğüne göre değişiklik göstermektedir. Bu değişimler için ısıtılacak olan odanın termodinamik denklemlerine başvurulabilir.

Kapalı çevrim kontrol algoritması ile kontrol edilen bir yapıda bu kontrol sayesinde enerji tasarrufu da sağlanmış olur. Sürekli olarak ısıtıcıyı, soğutucuyu çalıştırmak yerine istenilen sıcaklık değerinin sabit tutulmasını sağlamak daha ekonomiktir.

Sistem de yapılan ışık kontrolünde kapalı çevrim kontrol esas alınmıştır. Odada bulunan ışık sensörleri ile odanın farklı noktalarındaki ışık seviyeleri ölçülmüş ve bu seviyelere göre ışık grupları sürülmüştür. Bu sayede ışık grupları aynı seviye yerine gerekli olan seviyede ışık vererek oda içerisinde her noktada aynı ışık şiddetine minimum enerji harcayarak ulaşılmış olur. Cama yakın olan yani dışarıdan gün ışığı alan bölgeyi aydınlatan ışık grupları daha az, odanın karanlık bölümlerindeki ışık grupları ise o bölgeyi diğer bölgeler ile aynı aydınlık seviyesine getirmek için diğerlerinden daha fazla ışık verecek şekilde ayarlanır. Bu şekilde enerji tasarrufu sağlanmış olur. Ayrıca odanın her yerinde aynı aydınlık seviyesinin olması insan konforu açısından da önemlidir. Bu seviye çalışma performansını etkileyerek, çalışma süresinin optimum kullanımını sağlar.

İnsan konforu için en önemli unsurlardan biri de güvenliktir. İnsanoğlu var olduğu zamandan beri kendisinin ve etrafındakilerin güvenliğini düşünür. Günümüzde de insan sevdiklerine ve evine gelebilecek dış tehditlere karşı kendini savunma ihtiyacı duyar. Ev otomasyonunda da bu ihtiyaç göz önünde bulundurularak güvenlik kontrolü yapılmıştır. Eve karşı herhangi bir izinsiz girişe karşı bir alarm sistemi kurulmuştur.

Bilgisayar yazılımının desteği ile ev otomasyonunda kullanılan sistemlerin performansları izlenebilir, oda kontrol parametrelerinin zamana göre değişimi grafik ortamında gösterilebilir ve algoritmanın geliştirilmesi için saklanarak daha sonraki değerlerle karşılaştırma yapılabilir.

Bilgisayar yazılımı desteği ile ev otomasyon sistemine dışarıdan internet, Ethernet, GPRS gibi sistemler ile ulaşımı sağlanabilir. Bu sayede kullanıcı kontrol edilen odanın veya evin kontrol parametrelerini dışarıdan izleyebilir ve müdahalede bulunabilir.

Tasarlanan sistemdeki bilgisayar yazılımı ile kullanıcı ev içerisinde kontrol edilen sıcaklık, ışık ve güvenlik parametrelerine ulaşabilmekte ve sıcaklık-zaman grafiğini izleyebilmektedir.

Tasarlanan bilgisayar destekli ev sisteminde kullanıcı konforunu sağlamak, sağlığını korumak ve güvenliğini sağlamak hedeflentir.

KAYNAKLAR

Elmas Çetin (2003) "Bulanık Mantık Denetleyiciler",Seçkin Yayınevi , Ankara.

Jantzen Jan , "Tutorial On Fuzzy Logic"

Mendel Jerry, "Fuzzy Logic Systems For Engineering"

Zhang Jianwei, Wille Frank, Knoll Alois, "Fuzzy Logic Rules for Mapping Sensor Data to Robot Control"

Pacini Peter, Thorson Andrew, "Fuzzzy Logic Premier", Kasım 1990,Vanderbilt

L.A. Zadeh, "Fuzzy Sets, Information and Control,1965

L.A. Zadeh, "Outline of a New Approach to the Analysis of Complex Systems and Decision Process,1973

L.A. Zadeh , "Fuzzy Algorithms", 1968

Balkan Ezel, 2006, "Borland Delphi 7.0", Seçkin Yayınevi,Ankara

Taşbaşı G. Murat,2003, "İleri C Programlama", Atlaş Yayınevi, İstanbul

L.A. Zadeh, "Making computers think like people," IEEE. Spectrum, 8/1984,

Hellman M., "Fuzzy Logic Introduction" Fransa

Passino Kelvin, YurkovichStephen, "Fuzzy Control" 1998 , Addison Wesley Longman,

Takagi T., Sugeno M., (1985) "Fuzzy Identification of Systems and Its Application to Modeling and Control", IEEE Transactions on Systems Man and Cybernetics, Vol. 15. No. 1.

Kuo Bejamin, "Otomatil Kontrol Sistemleri" Literatür Yayınevi, Eylül 2005, İstanbul

Gümüşkaya Haluk, "Mikroişlemciler ve 8051 Ailesi" Alfa Yayınevi,Eylül 2002, İstanbul

www.analog.com

www.8052.com

www.delphitürkiye.com

EKLER

EK1 Oda Panosunun Yazılımı

```
/*====================================================================
Dosya Adi: Main.h
Açiklama:
====================================================================*/
extern unsigned char Temp;
extern signed int Fuzzy_CripValue;

extern unsigned char bdata ProcessFlags;
extern bit Flag_ChangeOnSetTemp;
extern bit Flag_DisplayFlash;
extern bit Flag_RoomActive;
extern bit Flag_FuzzyControl;
extern unsigned char FlashCounter;
extern unsigned char ExCounter;

extern unsigned char ByteCounter;
extern unsigned char *StringPointer;
extern unsigned char *ReceivePointer;
```

```
/*=====================================================================
Dosya Adi: Main.c
Açiklama:
=====================================================================*/
//#pragma SRC
//#pragma SYMBOLS CODE DEBUG
#include <ADuC841.h>
//#include <intrins.h>                  // rotate,nop islemleri için
#include <main.h>
#include <Interrupts.h>
#include <GeneralFunctions.h>
#include <fuzzy.h>
#include <7segment.h>
#include <Uart.h>


unsigned char Temp;
unsigned char FlashCounter;
signed int Fuzzy_CripValue;

unsigned char bdata ProcessFlags;
sbit Flag_ChangeOnSetTemp=ProcessFlags^0;
sbit Flag_DisplayFlash=ProcessFlags^1;
sbit Flag_RoomActive=ProcessFlags^2;
sbit Flag_FuzzyControl=ProcessFlags^3;
unsigned char FlashCounter;
unsigned char ExCounter;


unsigned char ByteCounter;
unsigned char *StringPointer;
unsigned char *ReceivePointer;


void main (void){
  MCU_Init();    // hiz  üç ikiye bölüyor bu seri portta sorun yaratabilir.
  ADC_Init();
  ADC_Calibration();
  Timer0Init();
  Timer1Init();
  ExtIntInit();
  InitUart();
  InitUartVariables();
  InitVariables();  //  tüm değişkenler atanarak ilk değerleri verilecek.


  while (1)
  {
   SCONV=1;
   ADCI=0;
          if (Flag_ChangeOnSetTemp==1)
                    {
                   DisplayDrive(SetTemperature,LightSet);
                    }
          else
                    {
                     DisplayDrive(RoomTemperature,LightSet);
                     }
          if (ADCI==1)
```

```
                {
                ADCI=0;
                Temp=TakeADCResult();
                TakeAverage(Temp);
                }

        if ((FuzzyTimeCounter==FuzzyTime)&&(Flag_FuzzyControl==1))
                {
                FuzzyTimeCounter=0x00;
                                                        //----- Fuzzy algortimasi
                CalculateRoomTemperature();
                CalculateError();
                CalculateDeltaError();
                FuzzificationOfError();
                FuzzificationOfDeltaError();
                DecisionCycle();
                Fuzzy_CripValue=Defuzzificaiton();

                if (Fuzzy_CripValue > 0)
                        {
                        Heater=1*((unsigned char)Fuzzy_CripValue);
                        Cooler=0;
                        Cooler_Out=1;
                        }
                else if (Fuzzy_CripValue < 0)
                        {
                        Fuzzy_CripValue=-Fuzzy_CripValue;
                        Cooler=1*((unsigned char)Fuzzy_CripValue);
                        Heater=0;
                        Heater_Out=1;
                        }
        }
        if (FlagPacketOK==1)
                {
                FlagPacketOK=0;
                 ReceivePointer=&ReceivedBytes;
                 PrepareUartBuffer();
                 SendingString(27,((unsigned char *)UartTransmitBuffer));
                 SendingChar(0x0D);
                 SendingChar(0x0A);
                }
        }
}
```

```
/*===================================================================
Dosya Adi: GeneralFunctions.h
Açiklama:  General Functions dosyasinda kullanilan fonsksiyonlar ve
                        degiskenlerin tanimlandigi alan.
===================================================================*/
//          Prototypes          (Funstions).....................................
extern void Timer0Init(void);
extern void Timer1Init(void);
extern void ExtIntInit(void);
extern void InitVariables(void);
extern void MCU_Init(void);
extern void ADC_Init(void);
extern void ADC_Calibration(void);
extern void ADC_Calibration(void);
extern void TakeAverage(unsigned char ADCResult);
extern unsigned char TakeADCResult(void);
extern void ChangeADCChannel (void);


//   Constantss.....................................................
#define PERIOD     -250              // 250 clock cycles interrupt period
#define AverageConstant 10
#define LimitTemp          2                                        //displayi set esnasinda flash
yapmak için kullanilir.
#define P_Control_Mul    40                                         //Sicakliktaki P control Katsayisi
#define P_Control_Div     100                                       //Sicakliktaki P control Katsayisi

extern unsigned char Timer;
extern unsigned char Average[10];
extern unsigned char AverageCounter;

extern signed char LDR_1;
extern signed char LDR_2;
extern signed char LDR_3;

extern unsigned char TMP35_1;
extern unsigned char TMP35_2;
extern unsigned char TMP35_3;

extern unsigned char Cooler;
extern unsigned char Heater;
extern unsigned char LightSet;
```

```
/*====================================================================
Dosya Adi: General Functions.c
Açiklama:  Genel amaçli fonksiyonların tanimlandigi dosya
====================================================================*/
#include <ADuC841.h>
#include <intrins.h>                    // rotate işlemleri için
#include <main.h>
#include <Interrupts.h>
#include <GeneralFunctions.h>
#include <fuzzy.h>
#include <7segment.h>
#include <Uart.h>

//cikis Fonksiyon adi (giris)
void Timer0Init(void);                  // giris ve cikis değeri yok.
void Timer1Init(void);
void ExtIntInit(void);
void InitVariables(void);
void MCU_Init(void);
void ADC_Init(void);
void ADC_Calibration(void);
void TakeAverage(unsigned char ADCResult); //  giris degeri istiyor.
unsigned char TakeADCResult(void);                      //  cikisda deger veriyor.
void ChangeADCChannel (void);
unsigned char LightControl (unsigned char LightSens,unsigned char LightValue);


unsigned char Timer;
unsigned char Average[10];
unsigned char AverageCounter;

signed char LDR_1;
signed char LDR_2;
signed char LDR_3;

unsigned char TMP35_1;
unsigned char TMP35_2;
unsigned char TMP35_3;

unsigned char Cooler;
unsigned char Heater;
unsigned char LightSet;

void Timer0Init(void)
{
 TH0   = 0xFF;       // set timer period
 TL0   = 0x21;
 TMOD  |= TMOD | 0x01;                // select mode 1
 TR0   = 1;                          // start timer 0
 ET0   = 1;                          // enable timer 0 interrupt
 EA    = 1;                         // global interrupt enable
 }
void Timer1Init(void)
{
 TH1   = 0xBB;       // set timer period
 TL1   = 0x21;
 TMOD  |= TMOD | 0x10;                // select mode 1
 TR1   = 1;                          // start timer 0
 ET1   = 1;                          // enable timer 0 interrupt
```

```
 EA    = 1;                        // global interrupt enable
}
void ExtIntInit(void)
          {
          IT0=1;              //Edge Sevsivity
          EX0=1;              // External Interrupt Enable;
          EA=1;
          }
void InitVariables(void)
{
 Timer0Counter=0x00;
 Timer1Counter=0x00;
 FuzzyTimeCounter=0;
 DisplayFlashTimeCounter=0;

 AverageCounter=0x00;
 Average[0]=0x00;
 Average[1]=0x00;
 Average[2]=0x00;
 Average[3]=0x00;
 Average[4]=0x00;
 Average[5]=0x00;
 Average[6]=0x00;
 Average[7]=0x00;
 Average[8]=0x00;
 Average[9]=0x00;

 LDR_1=0x00;
 LDR_2=0x00;
 LDR_3=0x00;
 TMP35_1=0x00;
 TMP35_2=0x00;
 TMP35_3=0x00;
 LightSet=0x00;
 SetTemperature=0x00;
 PreSetTemperature=0x00;
 Heater=0x00;
 Cooler=0x00;
 Fuzzy_CripValue=0;
 NewError=0;
 OldError=0;
 Flag_FuzzyControl=0;

 Heater_Out=1;
 Cooler_Out=1;
 Relay_Out=1;
 LedGroup_1=1;
 LedGroup_2=1;
 LedGroup_3=1;


 FlashTime=0;
 Flag_ChangeOnSetTemp=0;
 Flag_DisplayFlash=0;
 Flag_RoomActive=1;
 ExCounter=0x00;
}
void MCU_Init(void)
{
 //PLLCON |=PLLCON |0x03;        // Dis kristall i 8'e böler. 11.092Mhz/8=1.3824Mhz.
```

```
 //PLLCON |=PLLCON |0x00;        // Dis kristall i 1'e böler. 11.092Mhz/1=11.092Mhz
 CFG841=CFG841|0x01;
 EWAIT=0x07;
}
void ADC_Init(void)
{
          ADCCON1         = 0x8C; //10001100;
          /*                                        |||||||_EXC
                                        ||||||__T2C
                                                  |||||___AQ0
                                        |||||_____AQ1
                                        ||||______CK0
                                        |||_______CK1
                                        ||________EXT_REF
                                        |_________MDI
  */
          ADCCON2 = 0x00; //00000011;
  /*                              |||||||_CS0
                                                  ||||||__CS1
                                                  |||||___CS2
                                                  ||||____CS3
                                                  ||||_____SCONV
                                                  |||______CCONV
                                                  ||_______DMA
                                                  |________ADCI
  */
          ADCDATAH=0x00;
          ADCDATAL=0x00;
}
void ADC_Calibration(void)
{
          ADCI=0;
          ADCCON3 = 00000001;
          while (ADCI)
          {;;}
          ADCI =0;
          ADCCON3 = 00000011;
          while (ADCI)
          {;;}
          ADCI=0;
   SCONV=1;
}
void TakeAverage (unsigned char FilterResult)
{
unsigned int Sum;
unsigned char i;
unsigned char Duty;
          i=0;
          Sum=0;
          Duty=0;
          Average[AverageCounter]=FilterResult;
          AverageCounter=AverageCounter+1;
          if (AverageCounter>=AverageConstant)
          {
          AverageCounter=0;
          for (i=0;i<AverageConstant;i++)
                    {
                    Sum=Sum+Average[i];
                    }
          Duty=Sum/AverageConstant;
```

```
        if (ADCCON2==0x00)
                {
                Sum=Duty*100;  // 100 e oranlıyorum
                Duty=Sum/255;
                LDR_1=LightControl(Duty,LDR_1);
                }
        else if (ADCCON2==0x01)
                {
                Sum=Duty*100;  // 100 e oranlıyorum
                Duty=Sum/255;
                LDR_2=LightControl(Duty,LDR_2);
                }
   else if (ADCCON2==0x02)
                {
                Sum=Duty*100;  // 100 e oranlıyorum
                Duty=Sum/255;
                LDR_3=LightControl(Duty,LDR_3);
                }
        else if (ADCCON2==0x03)
                {
                TMP35_1=Duty;
                }
        else if (ADCCON2==0x04)
                {
                TMP35_2=Duty;
                }
        else if (ADCCON2==0x05)
                {
                TMP35_3=Duty;
                }
        else if (ADCCON2==0x06)
                {
                LightSet=Duty;
                if (LightSet>99)
                        {LightSet=99;}
                }
        else if (ADCCON2==0x07)
                {
                SetTemperature=Duty;
                if      (SetTemperature < 11)
                        {SetTemperature=10;}
                if      (SetTemperature >50)
                        {SetTemperature=50;}
                if (PreSetTemperature==0x00)
                        {PreSetTemperature=SetTemperature;}

                if ((PreSetTemperature<(SetTemperature-
LimitTemp))||(PreSetTemperature>(SetTemperature+LimitTemp)))
                        {
                         Flag_ChangeOnSetTemp=1;
                         Flag_DisplayFlash=1;
                         PreSetTemperature=SetTemperature;
                        }
                Flag_FuzzyControl=1;
                }
        ChangeADCChannel();
        }
}
unsigned char TakeADCResult(void)
```

```
{
unsigned char ADC_H;
unsigned char ADC_L;

        ADC_H  = ADCDATAH;
        ADC_H  = _crol_(ADC_H,4);
        ADC_H  = ADC_H & 0xF0;
        ADC_L  = ADCDATAL;
        ADC_L  = _cror_(ADC_L,4);
        ADC_L  = ADC_L & 0x0F;
        ADC_H  = ADC_H | ADC_L;

        //ADC_L  =0xFF;
        //ADC_H  = ADC_L-ADC_H;

        return ADC_H;
}
void ChangeADCChannel (void)
{
        if (ADCCON2 == 0x00)
                {ADCCON2 = 0x01;}
        else if (ADCCON2 == 0x01)
                {ADCCON2 = 0x02;}
        else if (ADCCON2 == 0x02)
                {ADCCON2 = 0x03;}
        else if (ADCCON2 == 0x03)
                {ADCCON2 = 0x04;}
        else if (ADCCON2 == 0x04)
                {ADCCON2 = 0x05;}
        else if (ADCCON2 == 0x05)
                {ADCCON2 = 0x06;}
        else if (ADCCON2 == 0x06)
                {ADCCON2 = 0x07;}
        else if (ADCCON2 == 0x07)
                {ADCCON2 = 0x00;}
}
unsigned char LightControl (unsigned char LightSens,unsigned char LightValue)
{
unsigned char LightError;
unsigned char LightOutput;
LightError=0;
LightOutput=LightValue;
LightSens=100-LightSens;
LightSens=(LightSens/2);

if (LightSet < LightSens)
        {
        LightError=LightSens-LightSet;
        if (LightError > LightOutput)
                {       LightOutput=2; }
        else
                {
                LightError=P_Control_Mul*LightError;
                LightError=LightError/P_Control_Div;
                LightOutput=LightOutput-LightError;
                }
        }
else if (LightSet > LightSens)
        {
        LightError=LightSet-LightSens;
```

```
		LightError=P_Control_Mul*LightError;
		LightError=LightError/P_Control_Div;
		if (LightOutput<99)
			{
			LightOutput=LightOutput+LightError;
			}
		if (LightOutput>98)
			{
			LightOutput=99;
			}
		}
 /*
if (ADCCON2==0x00)
		{
		SendingChar('G');
		SendingChar((LightSens/100)+0x30);
		SendingChar(((LightSens%100)/10)+0x30);
		SendingChar((LightSens%10)+0x30);

		SendingChar('S');
		SendingChar((LightSet/100)+0x30);
		SendingChar(((LightSet%100)/10)+0x30);
		SendingChar((LightSet%10)+0x30);

		SendingChar('H');
		if(LightError<0)
			{LightError=LightError*(-1);}
		SendingChar((LightError/100)+0x30);
		SendingChar(((LightError%100)/10)+0x30);
		SendingChar((LightError%10)+0x30);

		SendingChar('C');
		SendingChar((LightOutput/100)+0x30);
		SendingChar(((LightOutput%100)/10)+0x30);
		SendingChar((LightOutput%10)+0x30);
		SendingChar(0x0D);
		SendingChar(0x0A);
		}
		*/
return	LightOutput;
}
```

```
/*==================================================================
Dosya Adi: Int_header.h
Açiklama:  Interrupt ve Interruptta kullanilan fonksyionları içerir.
==================================================================*/
extern unsigned char Timer0Counter;              // variable to count interrupts
extern unsigned char Timer1Counter;              // variable to count interrupts
extern unsigned char FuzzyTimeCounter;
extern unsigned int DisplayFlashTimeCounter;


sbit LedGroup_1 = P3^3;
sbit LedGroup_2 = P3^4;
sbit LedGroup_3 = P3^5;
sbit Heater_Out   = P3^6;
sbit Cooler_Out   = P3^7;

sbit Relay_Out= P2^3;


#define Period     100                   //PWM'in periyodu.
#define FuzzyTime  100                   // fuzzy algoritmasinin kaç saniyede bir kosması için gerekli.
#define DisplayFlashTime 1000
```

```
/*====================================================================
Dosya Adi: Interrupt.c
Açiklama:  Interrupt ve Interruptta kullanilan fonksyionları içerir.
====================================================================*/
#include <ADuC841.h>
#include <main.h>
#include <Interrupts.h>
#include <GeneralFunctions.h>
#include <fuzzy.h>
#include <7segment.h>
#include <Uart.h>

 unsigned char Timer0Counter;              // variable to count interrupts
 unsigned char Timer1Counter;              // variable to count interrupts
 unsigned char FuzzyTimeCounter;
 unsigned int DisplayFlashTimeCounter;

void timer0 (void) interrupt 1 using 1{   // Int Vector at 000BH, Reg Bank 1
  TH0   = 0xFD;  //0xC9;
  TL0   = 0xAF;   //0xFF;

  Timer0Counter++;                        // increment interrupt counter
  if (Timer0Counter==Period)
            {
            Timer0Counter=0x00;
            LedGroup_1=0;
            LedGroup_2=0;
            LedGroup_3=0;

            }
  if (Timer0Counter==LDR_1)
            { LedGroup_1=1;                       }// LED_1 kapatildi.

  if (Timer0Counter==LDR_2)
            { LedGroup_2=1;                       }// LED_2 kapatildi.

  if (Timer0Counter==LDR_3)
            { LedGroup_3=1;                       }// LED_3 kapatildi.
}
void timer1 (void) interrupt 3 using 1{                   // Int Vector at 000BH, Reg Bank 1
  TH1   = 0xFA;       // set timer period
  TL1   = 0x00;
  Timer1Counter++;
  if (Timer1Counter==Period)
                        {
                        Timer1Counter=0;
                        if (Heater !=0x00)
                                  {Heater_Out=0;}
                        else
                                  {Heater_Out=1;}
                        if  (Cooler !=0x00)
                                  {Cooler_Out=0;}
                        else
                                  {Cooler_Out=1;}
                        }
  if ((Timer1Counter==Heater)&&(Heater!=0x00))
                        { Heater_Out=1;               }// Heater kapatildi.

  if ((Timer1Counter==Cooler)&&(Cooler!=0x00))
```

```
                { Cooler_Out=1;          }// Cooler kapatildi.

 FuzzyTimeCounter++;
 if (Flag_ChangeOnSetTemp==1)
        {
        DisplayFlashTimeCounter++;
        if (DisplayFlashTimeCounter==DisplayFlashTime)
                {
                DisplayFlashTimeCounter=0;
                if (Flag_DisplayFlash==1)
                        {Flag_DisplayFlash=0;}
                else
                        {Flag_DisplayFlash=1;}
                FlashTime++;
                if (FlashTime==100)
                        {
                        FlashTime=0;
                        Flag_ChangeOnSetTemp=0;
                        Flag_DisplayFlash=0;
                        }
                }
        }
}

void ExtInt (void) interrupt 0 using 1 {
 ExCounter++;
if (ExCounter==0x01)
        {
        ExCounter=0;
        if (Flag_RoomActive==1)
                        {
                        Relay_Out=0;
                        Flag_RoomActive=0;
                        }
                else
                        {
                        Relay_Out=1;
                        Flag_RoomActive=1;
                        }
        }

}
```

```
/*===================================================================
Dosya Adi: fuzzy.h
Açiklama:  fuzzy.c dosyasında kullanilan degiskenlerin belirtildigi,
                        Fonksiyonlarin prototypelerinin yazildigi alan.
===================================================================*/
//            Constants .....................................
//Fuzzy Uyelik Fonksiyonlari
#define ResolutionOfDegree                  100
#define NotADegree                              0xAA
// Error'un üyelik fonksiyonlarinin başlangiç ve kesisim noktalari
#define ErrorTopOfNB                        10
#define ErrorEndOfNK                        12
#define ErrorStartOfNB                      14
#define ErrorTopOfNK                        15
#define ErrorStartOfZero            16
#define ErrorStartOfNK                      18
#define ErrorTopOfZero                      20
#define ErrorStartOfPK                      22
#define ErrorEndOfZero                      24
#define ErrorTopOfPK                        25
#define ErrorStartOfPB                      26
#define ErrorEndOfPK                        28
#define ErrorTopOfPB                        30

// DeltaError'un üyelik fonksiyonlarinin başlangiç ve kesisim noktalari
#define DeltaErrorTopOfNB                   10
#define DeltaErrorEndOfNK                   12
#define DeltaErrorStartOfNB                 14
#define DeltaErrorTopOfNK                   15
#define DeltaErrorStartOfZero       16
#define DeltaErrorStartOfNK                 18
#define DeltaErrorTopOfZero                 20
#define DeltaErrorStartOfPK                 22
#define DeltaErrorEndOfZero                 24
#define DeltaErrorTopOfPK                   25
#define DeltaErrorStartOfPB                 26
#define DeltaErrorEndOfPK                   28
#define DeltaErrorTopOfPB                   30

//üyelik fonksiyonlari
#define NB              0
#define NK              1
#define Zero    2
#define PK              3
#define PB              4


//    SubRoutines.....................................................
extern void CalculateRoomTemperature (void);
extern void CalculateError(void);
extern void CalculateDeltaError(void);
extern void FuzzificationOfError (void);
extern void FuzzificationOfDeltaError (void);
extern void DecisionCycle (void);
extern signed char Defuzzificaiton (void);

//    Ports and Port Pins.....................................

//    Variables  (Bytes, Bits) .....................................
extern signed char xdata NewError;
```

```
extern signed char xdata OldError;
extern signed char xdata DeltaError;
extern unsigned char RoomTemperature;
extern unsigned char PreSetTemperature;
extern unsigned char SetTemperature;
extern unsigned char xdata DegreeOfError[4];                    // DegreeOfError[0]=ilk u degeri,
DegreeOfError[1]=Üyelik fonksiyonu DegreeOfError[2]=diger u degeri DegreeOfError[3]=diger u degerinin
üyelik fonksiyonu.
extern unsigned char xdata DegreeOfDeltaError[4];    // DegreeOfDeltaError[0]=ilk u degeri,
DegreeOfDeltaError[1]=Üyelik fonksiyonu .....
extern unsigned char xdata Decisions[8];
```

```
/*====================================================================
Dosya Adi: Fuzzy.c
Açiklama:
====================================================================*/
#include <ADuC841.h>
//#include <intrins.h>                  // rotate,nop islemleri için
#include <main.h>
#include <Interrupts.h>
#include <GeneralFunctions.h>
#include <fuzzy.h>
#include <7segment.h>
#include <Uart.h>


//    Subroutines .....................................................
void CalculateRoomTemperature (void);
void CalculateError(void);
void CalculateDeltaError(void);
void FuzzificationOfError (void);
void FuzzificationOfDeltaError (void);
unsigned char MaxDotFunction (unsigned char Error,unsigned char DeltaError);
void DecisionCycle (void);
unsigned char MakeDecisionFromTable (unsigned char Error,unsigned char DeltaError);
signed char Defuzzificaiton (void);
signed char FindCenterOfMembershipFunction (unsigned char MembershipFunction);

//    Variables  (Bytes, Bits) .....................................
unsigned char RoomTemperature;
unsigned char PreSetTemperature;
unsigned char SetTemperature;

signed char xdata NewError;
signed char xdata OldError;
signed char xdata DeltaError;
unsigned  char xdata DegreeOfError[4];                          // DegreeOfError[0]=ilk u degeri,
DegreeOfError[1]=Üyelik fonksiyonu DegreeOfError[2]=diger u degeri DegreeOfError[3]=diger u degerinin
üyelik fonksiyonu.
unsigned  char xdata DegreeOfDeltaError[4];    // DegreeOfDeltaError[0]=ilk u degeri,
DegreeOfDeltaError[1]=Üyelik fonksiyonu .....
unsigned  char xdata Decisions[8];                              // Kural tablosu ve karar kuralinin seçtiği u
degerleri bulunur.


//Kural Tablosu
unsigned  const code RuleLine1[5]={Zero, PK , PB , PB , PB };
unsigned  const code RuleLine2[5]={ NK ,Zero, PK , PB , PB };
unsigned  const code RuleLine3[5]={ NB , NK ,Zero, PK , PB };
unsigned  const code RuleLine4[5]={ NB , NB , NK ,Zero, PK };
unsigned  const code RuleLine5[5]={ NB , NB , NB , NK , Zero};



//    FUNCTIONS              .....................................
/*====================================================================
Fonksiyon Adi: CalculateError
Açiklama:   Kontrol algoritmasi için gerekli hatayi bulur.
                              -Sicaklik degerleri düzeltilip          LastTemperature, PreviousTemperature
                              içerisine koyulmuş olmalidir.
```

```
                        -Hata=Set-Ölçülen
Cikis:    NewError ve OldError  degerler milisantigrad
==========================================================================*/
void CalculateRoomTemperature (void){
unsigned char AvaregeOfTMPs;

AvaregeOfTMPs=TMP35_1;
AvaregeOfTMPs=AvaregeOfTMPs+TMP35_2;
AvaregeOfTMPs=AvaregeOfTMPs+TMP35_3;
AvaregeOfTMPs=AvaregeOfTMPs/3;
RoomTemperature=AvaregeOfTMPs;}
/*==========================================================================
Fonksiyon Adi: CalculateError
Açiklama:   Kontrol algoritmasi için gerekli hatayi bulur.
                        -Sicaklik degerleri düzeltilip          LastTemperature, PreviousTemperature
                        içerisine koyulmuş olmalidir.
                        -Hata=Set-Ölçülen
Cikis:    NewError ve OldError  degerler milisantigrad
==========================================================================*/
void CalculateError (void){
//OldError=NewError;
//NewError=SetTemperature-RoomTemperature;
NewError=SetTemperature;
}
/*==========================================================================
Fonksiyon Adi: CalculateDeltaError
Açiklama:   Kontrol algoritmasi için gerekli hatanin degisimini bulur.
                              degerler milisantigrad
==========================================================================*/
void CalculateDeltaError (void)
{// DeltaError=(NewError-OldError);
 DeltaError=RoomTemperature; }
/*==========================================================================
Fonksiyon Adi: FuzzificationOfError
Açiklama:         NewError degerlerini bulaniklastirir.
degerler milisantigrad
==========================================================================*/
void FuzzificationOfError (void){
signed int MemberShip;

MemberShip=0;
if (NewError < ErrorTopOfNB){
        DegreeOfError[0]=ResolutionOfDegree;
        DegreeOfError[1]=NB;

        DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
        DegreeOfError[3]=NotADegree; }
else if (NewError < ErrorEndOfNK) {
                MemberShip=0;
                MemberShip=(25*NewError)*(-1);                  // y=-25x+350
                MemberShip=MemberShip+350;
                DegreeOfError[0]=MemberShip;
                DegreeOfError[1]=NB;

                DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise
o DegreeOfErroryoktur.
          DegreeOfError[3]=NotADegree; }
else if (NewError < ErrorStartOfNB)  {
        MemberShip=0;
```

```
            MemberShip=(25*NewError)*(-1);                  // y=-25x+350
            MemberShip=MemberShip+350;
            DegreeOfError[0]=MemberShip;
            DegreeOfError[1]=NB;

            MemberShip=0;
            MemberShip=100*NewError;                // y=100x-1200
            MemberShip=MemberShip-1200;
            DegreeOfError[2]=MemberShip;
            DegreeOfError[3]=NK; }
else if (NewError < ErrorTopOfNK) {
            MemberShip=0;
            MemberShip=100*NewError;                // y=100x-1200
            MemberShip=MemberShip-1200;
            DegreeOfError[0]=MemberShip;
            DegreeOfError[1]=NK;

            DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
            DegreeOfError[3]=NotADegree;    }
else if (NewError < ErrorStartOfZero) {
            MemberShip=0;
            MemberShip=(100*NewError)*(-1);                 // y=-(100*x)/3+600
            MemberShip=MemberShip/3+600;
            DegreeOfError[0]=MemberShip;
            DegreeOfError[1]=NK;

            DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
            DegreeOfError[3]=NotADegree; }
else if (NewError < ErrorStartOfNK) {
            MemberShip=0;
            MemberShip=(100*NewError)*(-1);                 // y=-(100*x)/3+600
            MemberShip=MemberShip/3+600;
            DegreeOfError[0]=MemberShip;
            DegreeOfError[1]=NK;

            MemberShip=0;
            MemberShip=25*NewError;                 // y=25x-400
            MemberShip=MemberShip-400;
            DegreeOfError[2]=MemberShip;
            DegreeOfError[3]=Zero;   }
else if (NewError < ErrorTopOfZero) {
            MemberShip=0;
            MemberShip=25*NewError;                 // y=25x-400
            MemberShip=MemberShip-400;
            DegreeOfError[0]=MemberShip;
            DegreeOfError[1]=Zero;
            DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
            DegreeOfError[3]=NotADegree;  }
else if (NewError < ErrorStartOfPK)  {
            MemberShip=0;
            MemberShip=(25*NewError)*(-1);                  // y=-25*x+600
            MemberShip=MemberShip+600;
            DegreeOfError[0]=MemberShip;
            DegreeOfError[1]=Zero;
            DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
            DegreeOfError[3]=NotADegree;    }
```

```
else if (NewError < ErrorEndOfZero) {
        MemberShip=0;
        MemberShip=(25*NewError)*(-1);                  // y=-25*x+600
        MemberShip=MemberShip+600;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=Zero;

        MemberShip=0;
        MemberShip=100*NewError;                // y=100x/3-2200/3 =100x/3-733
        MemberShip=MemberShip/3-733;
        DegreeOfError[2]=MemberShip;
        DegreeOfError[3]=PK;            }

else if (NewError < ErrorTopOfPK)       {
        MemberShip=0;
        MemberShip=100*NewError;                // y=100x/3-2200/3 =100x/3-733
        MemberShip=MemberShip/3-733;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=PK;
        DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
        DegreeOfError[3]=NotADegree;    }
else if (NewError < ErrorStartOfPB)     {
        MemberShip=0;
        MemberShip=(100*NewError)*(-1);                 // y=-(100*x)/3+2800/3
        MemberShip=MemberShip/3+933;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=PK;
        DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
        DegreeOfError[3]=NotADegree;    }
else if (NewError < ErrorEndOfPK)       {
        MemberShip=0;
        MemberShip=(100*NewError)*(-1);                 // y=-(100*x)/3+2800/3
        MemberShip=MemberShip/3+933;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=PK;

        MemberShip=0;
        MemberShip=25*NewError;                 // y=25x-650
        MemberShip=MemberShip-650;
        DegreeOfError[2]=MemberShip;
        DegreeOfError[3]=PB;                    }
else if (NewError < ErrorTopOfPB)       {
        MemberShip=0;
        MemberShip=25*NewError;             // y=25x-650
        MemberShip=MemberShip-650;
        DegreeOfError[0]=MemberShip;
        DegreeOfError[1]=PB;
        DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
        DegreeOfError[3]=NotADegree;    }
else if (NewError >= ErrorTopOfPB)      {
        DegreeOfError[0]=ResolutionOfDegree;
        DegreeOfError[1]=PB;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
        DegreeOfError[2]=NotADegree;                    // deger 99 degerinden buyuk ise o
DegreeOfErroryoktur.
        DegreeOfError[3]=NotADegree;    }
}
```

```
/*====================================================================
Fonksiyon Adi: FuzzificationOfDeltaError
Açiklama:          DeltaError degerlerini bulaniklastirir.
degerler milisantigrad
====================================================================*/
void FuzzificationOfDeltaError (void)
{
signed int MemberShipDeltaError;
MemberShipDeltaError=0;

if (DeltaError < DeltaErrorTopOfNB){
          DegreeOfDeltaError[0]=ResolutionOfDegree;
          DegreeOfDeltaError[1]=NB;
          DegreeOfDeltaError[2]=NotADegree;                              // deger 99 degerinden buyuk ise
o DegreeOfErroryoktur.
          DegreeOfDeltaError[3]=NotADegree;}
else if (DeltaError < DeltaErrorEndOfNK){
                    MemberShipDeltaError=0;
                    MemberShipDeltaError=(25*NewError)*(-1);                    // y=-25x+350
                    MemberShipDeltaError=MemberShipDeltaError+350;
                    DegreeOfDeltaError[0]=MemberShipDeltaError;
                    DegreeOfDeltaError[1]=NB;
                    DegreeOfDeltaError[2]=NotADegree;
                    DegreeOfDeltaError[3]=NotADegree;}
else if (DeltaError < DeltaErrorStartOfNB){
          MemberShipDeltaError=0;
          MemberShipDeltaError=(25*NewError)*(-1);                    // y=-25x+350
          MemberShipDeltaError=MemberShipDeltaError+350;
          DegreeOfDeltaError[0]=MemberShipDeltaError;
          DegreeOfDeltaError[1]=NB;

          MemberShipDeltaError=0;
          MemberShipDeltaError=100*NewError;                    // y=100x-1200
          MemberShipDeltaError=MemberShipDeltaError-1200;
          DegreeOfDeltaError[2]=MemberShipDeltaError;
          DegreeOfDeltaError[3]=NK;         }
else if (DeltaError < DeltaErrorTopOfNK){
          MemberShipDeltaError=0;
          MemberShipDeltaError=100*NewError;                    // y=100x-1200
          MemberShipDeltaError=MemberShipDeltaError-1200;
          DegreeOfDeltaError[0]=MemberShipDeltaError;
          DegreeOfDeltaError[1]=NK;
          DegreeOfDeltaError[2]=NotADegree;
          DegreeOfDeltaError[3]=NotADegree;         }
else if (DeltaError < DeltaErrorStartOfZero){
          MemberShipDeltaError=0;
          MemberShipDeltaError=(100*NewError)*(-1);                    // y=-100x/3+600
          MemberShipDeltaError=MemberShipDeltaError/3+600;
          DegreeOfDeltaError[0]=MemberShipDeltaError;
          DegreeOfDeltaError[1]=NK;
          DegreeOfDeltaError[2]=NotADegree;
          DegreeOfDeltaError[3]=NotADegree;         }
else if (DeltaError < DeltaErrorStartOfNK)  {
          MemberShipDeltaError=0;
          MemberShipDeltaError=(100*NewError)*(-1);                    // y=-100x/3+600
          MemberShipDeltaError=MemberShipDeltaError/3+600;
          DegreeOfDeltaError[0]=MemberShipDeltaError;
          DegreeOfDeltaError[1]=NK;

          MemberShipDeltaError=0;
```

```
            MemberShipDeltaError=25*NewError;                       // y=25x-400
            MemberShipDeltaError=MemberShipDeltaError-400;
            DegreeOfDeltaError[2]=MemberShipDeltaError;
            DegreeOfDeltaError[3]=Zero;                 }
else if (DeltaError < DeltaErrorTopOfZero) {
            MemberShipDeltaError=0;
            MemberShipDeltaError=25*NewError;                       // y=25x-400
            MemberShipDeltaError=MemberShipDeltaError-400;
            DegreeOfDeltaError[0]=MemberShipDeltaError;
            DegreeOfDeltaError[1]=Zero;
            DegreeOfDeltaError[2]=NotADegree;
            DegreeOfDeltaError[3]=NotADegree;       }
else if (DeltaError < DeltaErrorStartOfPK){
            MemberShipDeltaError=0;
            MemberShipDeltaError=(25*NewError)*(-1);                        // y=-25x+600
            MemberShipDeltaError=MemberShipDeltaError+600;
            DegreeOfDeltaError[0]=MemberShipDeltaError;
            DegreeOfDeltaError[1]=Zero;
            DegreeOfDeltaError[2]=NotADegree;
            DegreeOfDeltaError[3]=NotADegree;       }
else if (DeltaError < DeltaErrorEndOfZero){
            MemberShipDeltaError=0;
            MemberShipDeltaError=(25*NewError)*(-1);                        // y=-25x+600
            MemberShipDeltaError=MemberShipDeltaError+600;
            DegreeOfDeltaError[0]=MemberShipDeltaError;
            DegreeOfDeltaError[1]=Zero;

            MemberShipDeltaError=0;
            MemberShipDeltaError=100*NewError;                      // y=(100*x)/3-633
            MemberShipDeltaError=MemberShipDeltaError/3-633;
            DegreeOfDeltaError[2]=MemberShipDeltaError;
            DegreeOfDeltaError[3]=PK;       }

else if (DeltaError < DeltaErrorTopOfPK){
            MemberShipDeltaError=0;
            MemberShipDeltaError=100*NewError;                      // y=(100*x)/3-633
            MemberShipDeltaError=MemberShipDeltaError/3-633;
            DegreeOfDeltaError[0]=MemberShipDeltaError;
            DegreeOfDeltaError[1]=PK;
            DegreeOfDeltaError[2]=NotADegree;
            DegreeOfDeltaError[3]=NotADegree;       }
else if (DeltaError < DeltaErrorStartOfPB){
            MemberShipDeltaError=0;
            MemberShipDeltaError=(100*NewError)*(-1);                       // y=-100x/3+933
            MemberShipDeltaError=MemberShipDeltaError/3+933;
            DegreeOfDeltaError[0]=MemberShipDeltaError;
            DegreeOfDeltaError[1]=PK;
            DegreeOfDeltaError[2]=NotADegree;
            DegreeOfDeltaError[3]=NotADegree;}
else if (DeltaError < DeltaErrorEndOfPK){
            MemberShipDeltaError=0;
            MemberShipDeltaError=(100*NewError)*(-1);                       // y=-100x/3+933
            MemberShipDeltaError=MemberShipDeltaError/3+933;
            DegreeOfDeltaError[0]=MemberShipDeltaError;
            DegreeOfDeltaError[1]=PK;

            MemberShipDeltaError=0;
            MemberShipDeltaError=25*NewError;                       // y=25x-650
            MemberShipDeltaError=MemberShipDeltaError-650;
            DegreeOfDeltaError[2]=MemberShipDeltaError;
```

```
        DegreeOfDeltaError[3]=PB;       }
else if (DeltaError < DeltaErrorTopOfPB){
        MemberShipDeltaError=0;
        MemberShipDeltaError=25*NewError;                       // y=25x-650
        MemberShipDeltaError=MemberShipDeltaError-650;
        DegreeOfDeltaError[0]=MemberShipDeltaError;
        DegreeOfDeltaError[1]=PB;
        DegreeOfDeltaError[2]=NotADegree;
        DegreeOfDeltaError[3]=NotADegree;}
else if (DeltaError >= DeltaErrorTopOfPB){
        DegreeOfDeltaError[0]=ResolutionOfDegree;
        DegreeOfDeltaError[1]=PB;
        DegreeOfDeltaError[2]=NotADegree;                       // deger 99 degerinden buyuk ise
o DegreeOfErroryoktur.
        DegreeOfDeltaError[3]=NotADegree;}                      // deger 99 degerinden buyuk ise
o DegreeOfErroryoktur.
}
/*====================================================================
Fonksiyon Adi:  DecisionCycle
Açiklama:       BUlanklastirilmis degerleri tabloya göre bulanık sonuç verir.
degerler Bulaniktir.
====================================================================*/
void DecisionCycle (void){
unsigned char Error;
unsigned char DeltaError;
Error=DegreeOfError[1];
DeltaError=DegreeOfDeltaError[1];
Decisions[0]=MakeDecisionFromTable(Error,DeltaError);

Error=DegreeOfError[0];
DeltaError=DegreeOfDeltaError[0];
Decisions[1]=MaxDotFunction(Error,DeltaError);

Error=DegreeOfError[1];
DeltaError=DegreeOfDeltaError[3];
Decisions[2]=MakeDecisionFromTable(Error,DeltaError);
Error=DegreeOfError[0];
DeltaError=DegreeOfDeltaError[2];
Decisions[3]=MaxDotFunction(Error,DeltaError);

Error=DegreeOfError[3];
DeltaError=DegreeOfDeltaError[1];
Decisions[4]=MakeDecisionFromTable(Error,DeltaError);
Error=DegreeOfError[2];
DeltaError=DegreeOfDeltaError[0];
Decisions[5]=MaxDotFunction(Error,DeltaError);

Error=DegreeOfError[3];
DeltaError=DegreeOfDeltaError[3];
Decisions[6]=MakeDecisionFromTable(Error,DeltaError);

Error=DegreeOfError[2];
DeltaError=DegreeOfDeltaError[2];
Decisions[7]=MaxDotFunction(Error,DeltaError);    }
 /*====================================================================
Fonksiyon Adi:  MakeDecisionFromTable
Açiklama:       Girilen Error ve Delta Error'e göre tabloda karar çikartir.
degerler Bulaniktir.                    s
====================================================================*/
unsigned char MakeDecisionFromTable (unsigned char Error,unsigned char DeltaError){
```

```
unsigned char Decision;

Decision=0;


if ((DeltaError==NotADegree)||(Error==NotADegree)){
        Decision=NotADegree;}
else{
        if ((DeltaError)==NB){
                Decision=RuleLine1[Error];}
        else if ((DeltaError)==NK){
                Decision=RuleLine2[Error];      }
        else if ((DeltaError)==Zero){
                Decision=RuleLine3[Error];}
        else if ((DeltaError)==PK){
                Decision=RuleLine4[Error];}
        else if ((DeltaError)==PB){
                Decision=RuleLine5[Error];}
        }
return Decision;
}
 /*====================================================================
Fonksiyon Adi:  MaxDotFunction
Açiklama:       u degerleri içinden büyük olani alir.
degerler Bulaniktir.
====================================================================*/
unsigned char MaxDotFunction (unsigned char Error,unsigned char DeltaError){
unsigned char MaxDot;

if ((DeltaError==NotADegree)||(Error==NotADegree)){
        MaxDot=NotADegree;}
else{
         if (DeltaError>Error)
                {MaxDot=DeltaError;}
        else
                {MaxDot=Error;}
        }
return MaxDot;
}
 /*====================================================================
Fonksiyon Adi:  Defuzzification
Açiklama:       Bulanik degerleri kesin degerlere dönüştürür.
                                Agırlıklı Ortalama Methodu ile dönüsüm yapilmistir.
====================================================================*/
signed char Defuzzificaiton (void){
signed int Mul1;
signed int Mul2;
signed int Mul3;
signed int Mul4;
signed int Crisp;
signed int Addition;

signed int Center;
Center=0;
Crisp=0;
Mul1=0;
Mul2=0;
Mul3=0;
Mul4=0;
```

```
if          (Decisions[0]!=NotADegree){
            Center=FindCenterOfMembershipFunction(Decisions[0]);
            Mul1=Center * Decisions[1];}
else
            {Mul1=0;}

if          (Decisions[2]!=NotADegree)
            { Center=FindCenterOfMembershipFunction(Decisions[2]);
              Mul2=Center * Decisions[3]; }
else{Mul2=0;}

if          (Decisions[4]!=NotADegree){
            Center=FindCenterOfMembershipFunction(Decisions[4]);
            Mul3=Center * Decisions[5]; }
else{Mul3=0;}

if          (Decisions[7]!=NotADegree){
            Center=FindCenterOfMembershipFunction(Decisions[6]);
            Mul4=Center * Decisions[7];}
else {Mul4=0;}
Center=Mul1+Mul2;
Center=Center+Mul3;
Center=Center+Mul4;


Addition=0;
if          (Decisions[1]!=NotADegree){Addition=Decisions[1];}
if (Decisions[3]!=NotADegree){Addition=Addition+Decisions[3];}
if (Decisions[5]!=NotADegree){Addition=Addition+Decisions[5];}
if (Decisions[7]!=NotADegree){Addition=Addition+Decisions[7];}

Crisp=Center/Addition;

return Crisp;
}
/*==================================================================
Fonksiyon Adi:  FindCenterOfMembershipFunction
Açiklama:           Bulanik degerleri kesin degerlere dönüştürür.
                                    Agirlikli Ortalama Methodu ile dönüsüm yapilmistir.
==================================================================*/
signed char FindCenterOfMembershipFunction (unsigned char MembershipFunction){
unsigned char CenterOfMembership;

CenterOfMembership=0;
if (MembershipFunction==NB)
            {CenterOfMembership=-90;}
else if  (MembershipFunction==NK)
            {CenterOfMembership=-50;}
else if  (MembershipFunction==Zero)
            {CenterOfMembership=0;}
else if  (MembershipFunction==PK)
            {CenterOfMembership=50;}
else if  (MembershipFunction==PB)
            {CenterOfMembership=90;}
            return CenterOfMembership;
}
```

```
/*===================================================================
Dosya Adi: GeneralFunctions.h
Açiklama:  General Functions dosyasinda kullanilan fonsksiyonlar ve
                        degiskenlerin tanimlandigi alan.
===================================================================*/
//          Prototypes          (Funstions).....................................
extern void Timer0Init(void);
extern void Timer1Init(void);
extern void ExtIntInit(void);
extern void InitVariables(void);
extern void MCU_Init(void);
extern void ADC_Init(void);
extern void ADC_Calibration(void);
extern void ADC_Calibration(void);
extern void TakeAverage(unsigned char ADCResult);
extern unsigned char TakeADCResult(void);
extern void ChangeADCChannel (void);


//   Constantss.....................................................
#define PERIOD     -250              // 250 clock cycles interrupt period
#define AverageConstant 10
#define LimitTemp          2                                        //displayi set esnasinda flash
yapmak için kullanilir.
#define P_Control_Mul    40                                         //Sicakliktaki P control Katsayisi
#define P_Control_Div     100                                       //Sicakliktaki P control Katsayisi

extern unsigned char Timer;
extern unsigned char Average[10];
extern unsigned char AverageCounter;

extern signed char LDR_1;
extern signed char LDR_2;
extern signed char LDR_3;

extern unsigned char TMP35_1;
extern unsigned char TMP35_2;
extern unsigned char TMP35_3;

extern unsigned char Cooler;
extern unsigned char Heater;
extern unsigned char LightSet;
```

```
/*=====================================================================
Dosya Adi: General Functions.c
Açiklama:  Genel amaçli fonksiyonların tanimlandigi dosya
=====================================================================*/
#include <ADuC841.h>
#include <intrins.h>                    // rotate işlemleri için
#include <main.h>
#include <Interrupts.h>
#include <GeneralFunctions.h>
#include <fuzzy.h>
#include <7segment.h>
#include <Uart.h>



//fonksiyon Protype lari
void SendSCKClock (void);
void SendRCKClock (void);
void SendBitBit(unsigned char Byte);
void Prepare7SegmentData(unsigned char DisplayValue);
void DisplayDrive (unsigned char Temperature, unsigned char Light);

// degiskenler
unsigned char code
SevenSegmentTab[14]={0x84,0xAF,0x98,0x89,0xA3,0xC1,0xC0,0x8F,0x80,0x81,0x93,0xD4,0xE8,0xFF};
//                                                                                  '0'  '1' '2' '3' '4' '5' '6' '7' '8'
'9'  '°'   'C'        '%' 'Blank'
unsigned char FlashTime;

void SendSCKClock (void){
        SCK=0;
        _nop_();
        _nop_();
        _nop_();
        SCK=1;
        _nop_();
        _nop_();
        _nop_();
        SCK=0;
        _nop_();
        _nop_();
        _nop_();
}
void SendRCKClock (void){
        RCK=0;
        _nop_();
        _nop_();
        _nop_();
        RCK=1;
        _nop_();
        _nop_();
        _nop_();
        RCK=0;
        _nop_();
        _nop_();
        _nop_();
}
void SendBitBit(unsigned char Byte){
```

```
unsigned char Temp_Byte;
unsigned char BitCounter;
        BitCounter=8;
        while (BitCounter > 0)
                {
                Temp_Byte=Byte & 0x01;
                if (Temp_Byte==0x01)
                        { SER=1; }
                else
                        { SER=0; }
                SendSCKClock();
                Byte >>=1;
                BitCounter--;
                }
}
void Prepare7SegmentData(unsigned char DisplayValue){
unsigned char DisplayData;
        DisplayData=SevenSegmentTab[DisplayValue];
        SendBitBit(DisplayData);
}
void DisplayDrive (unsigned char Temperature, unsigned char Light){
unsigned Temp_Onlar;
unsigned Temp_Birler;
unsigned Light_Onlar;
unsigned Light_Birler;
  Temp_Onlar= Temperature/10;
  Temp_Birler=Temperature-(10*Temp_Onlar);

  Light_Onlar= Light/10;
  Light_Birler=Light-(10*Light_Onlar);

if (Flag_DisplayFlash==1)
        {
        Prepare7SegmentData(Blank);
        Prepare7SegmentData(Blank);
        Prepare7SegmentData(Blank);
        Prepare7SegmentData(Blank);
        Prepare7SegmentData(Blank);
        Prepare7SegmentData(Blank);
        Prepare7SegmentData(Blank);
        Prepare7SegmentData(Blank);
        SendRCKClock();
        }
else
        {
        Prepare7SegmentData(Light_Birler);
        Prepare7SegmentData(Light_Onlar);
        Prepare7SegmentData(Yuzde);
        Prepare7SegmentData(Derece);
        Prepare7SegmentData(C);
        Prepare7SegmentData(Derece);
        Prepare7SegmentData(Temp_Birler);
        Prepare7SegmentData(Temp_Onlar);
        SendRCKClock();

        }
}
```

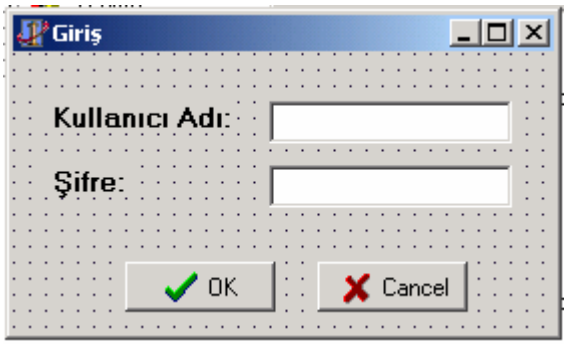
```
/*===================================================================
Dosya Adi: Uart.h
Açiklama:  Uart.c dosyasında kullanilan degiskenlerin belirtildigi,
                              Fonksiyonlarin prototypelerinin yazildigi alan.
===================================================================*/
//    SubRoutines.....................................
extern void InitUart(void);
extern void InitUartVariables(void);
extern void SendingChar(unsigned char Karakter);
extern void SendingString(unsigned Long,unsigned char *Pointer);
void PrepareUartBuffer(void);


//    Variables  (Bytes, Bits) .....................................
extern unsigned char xdata ReceivedBytes[10];
extern unsigned char xdata UartTransmitBuffer[32];
extern unsigned char bdata UartFlags;
extern bit FlagSending;
extern bit FlagPacketOK;

//              Constants .....................................................
#define Room      0x31                  // '1'
```

```
/*=====================================================================
Dosya Adi: Uart.c
Açiklama:  Seri Port kullanma Alt Programlari
=====================================================================*/
#include <ADuC841.h>
#include <intrins.h>                    // rotate işlemleri için
#include <main.h>
#include <Interrupts.h>
#include <GeneralFunctions.h>
#include <fuzzy.h>
#include <7segment.h>
#include <Uart.h>


//    Subroutines .....................................................
void InitUart(void);
void InitUartVariables(void);
void SendingChar(unsigned char Karakter);
void SendingString(unsigned Long,unsigned char *Pointer);
void PrepareUartBuffer(void);

//    Variables  (Bytes, Bits) .....................................
unsigned char xdata ReceivedBytes[10];
unsigned char xdata UartTransmitBuffer[32];
unsigned char bdata UartFlags;
sbit FlagSending=UartFlags^0;
sbit FlagPacketOK=UartFlags^1;
//_____________________________________________________________________

void Uart (void) interrupt 4 using 1{
if (TI==1)
          {
          TI=0;
          FlagSending=0;
          if (ByteCounter!=0x00)
                    {
                    ByteCounter--;
                    StringPointer++;
                    SBUF=*StringPointer;
                    FlagSending=1;
                    }
          }
else if (RI==1)
          {
          RI=0;
          *ReceivePointer=SBUF;
          if ((*ReceivePointer==0x0A)&&((*(ReceivePointer-1)==0x0D)))
                    {
                    if ( (*(ReceivePointer-6)=='<')&&(*(ReceivePointer-5)=='O')&&(*(ReceivePointer-
4)==':')&&(*(ReceivePointer-2)=='>'))
                              {
                              if ((*(ReceivePointer-3)==Room))
                                        {

                                        FlagPacketOK=1;
                                        ReceivePointer=&ReceivedBytes;
                                        }
                              }
```

```
                }
        else
                {
                ReceivePointer++;
                FlagPacketOK=0;
                }
        }

}
void InitUart(void){
        T3CON=0x86;
        T3FD=0X08;
        SCON=0x52;
        EA=1;
        ES=1;
}

void InitUartVariables(void){
        ByteCounter=0x00;
        StringPointer=&ByteCounter;
        FlagSending=0;
        ReceivePointer=&ReceivedBytes;
}

void SendingChar(unsigned char Karakter){
        while (FlagSending==1)
                {;;}
        SBUF=Karakter;
        ByteCounter=0x00;
        FlagSending=1;

}
void SendingString(unsigned Long, unsigned char *Pointer){
        while (FlagSending==1)
                {;;}
        ByteCounter=Long;
        StringPointer=Pointer;
        SBUF=*StringPointer;
        FlagSending=1;
}

void PrepareUartBuffer(void)
{
unsigned char UartChecksum;
unsigned char i;
unsigned char Data_Onlar;
unsigned char Data_Birler;

UartTransmitBuffer[0]='<';
UartTransmitBuffer[1]='O';
UartTransmitBuffer[2]=':';
UartTransmitBuffer[3]=Room;
UartTransmitBuffer[4]=' ';

UartTransmitBuffer[5]='S';
UartTransmitBuffer[6]=':';
Data_Onlar=SetTemperature/10;
Data_Onlar=Data_Onlar+0x30;
Data_Birler=(SetTemperature%10)+0x30;
UartTransmitBuffer[7]=Data_Onlar;
```

```
UartTransmitBuffer[8]=Data_Birler;
UartTransmitBuffer[9]=' ';


UartTransmitBuffer[10]='R';
UartTransmitBuffer[11]=':';
Data_Onlar=(RoomTemperature/10)+0x30;
Data_Birler=(RoomTemperature%10)+0x30;
UartTransmitBuffer[12]=Data_Onlar;
UartTransmitBuffer[13]=Data_Birler;
UartTransmitBuffer[14]=' ';

UartTransmitBuffer[15]='L';
UartTransmitBuffer[16]=':';
Data_Onlar=(LightSet/10)+0x30;
Data_Birler=(LightSet%10)+0x30;
UartTransmitBuffer[17]=Data_Onlar;
UartTransmitBuffer[18]=Data_Birler;
UartTransmitBuffer[19]=' ';

UartTransmitBuffer[20]='W';
UartTransmitBuffer[21]=':';
UartTransmitBuffer[22]=' '; // Pencerenin durumu
UartTransmitBuffer[23]=' ';

i=0;
UartChecksum=0;
while (i<24)
        {
        UartChecksum=UartChecksum+UartTransmitBuffer[i];
        i++;
        }
UartTransmitBuffer[24]='C';
UartTransmitBuffer[25]=':';
UartTransmitBuffer[26]=UartChecksum;
UartTransmitBuffer[27]='>';

}
```

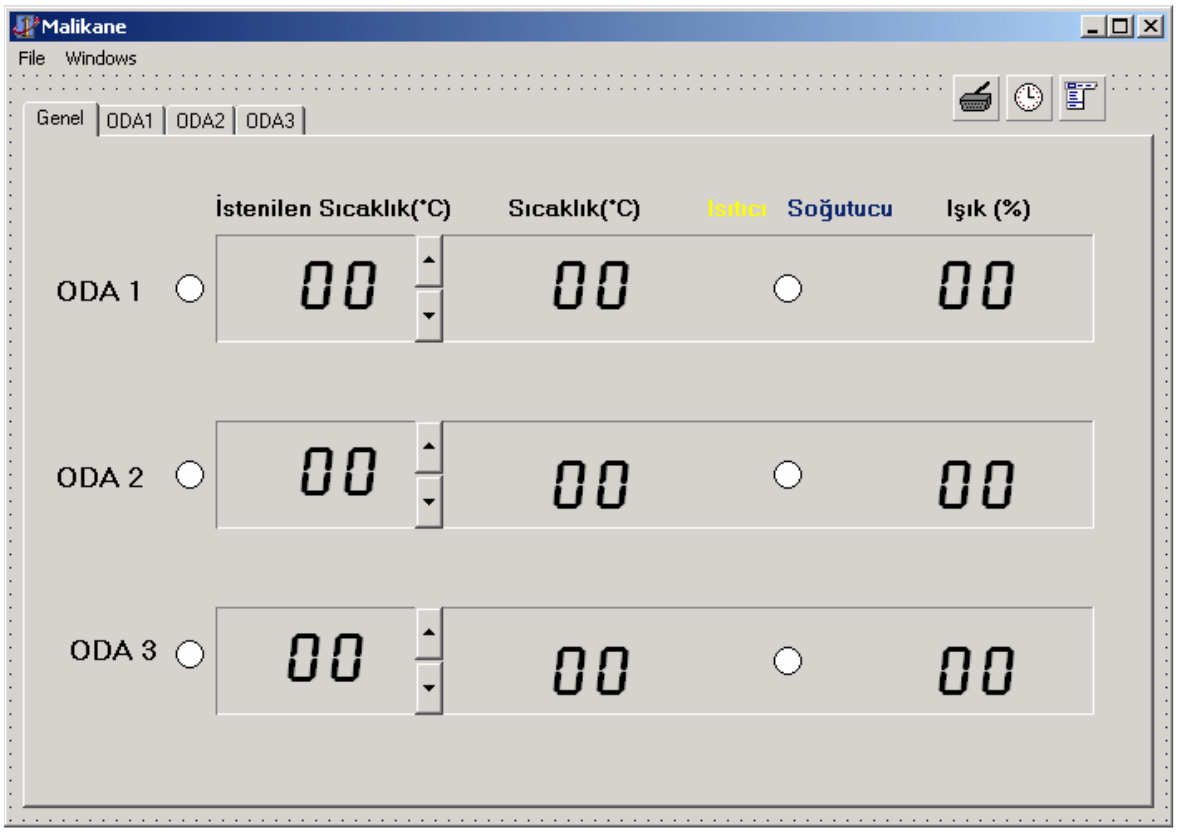
EK2 Bilgisayar Yazılımı




```
unit Giris;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Buttons;

type
  TForm1 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure Edit1Change(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Edit2Change(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

uses Odalar;

{$R *.DFM}
```

```
procedure TForm1.BitBtn1Click(Sender: TObject);
begin
if ((Form1.Edit1.Text='') or (Form1.Edit2.Text=''))then
      begin
      ShowMessage('Lutfen Kullanıcı Adınızı ve Şifrenizi giriniz!!!');
      end;

if Form1.Edit1.Text='Levent' then
      begin
      if Form1.Edit2.Text='JinX83' then
            begin
            Form2.Visible:=True;
            Form1.Visible:=False;
            Form2.Timer1.Enabled:=True;
            end
      else
            begin
            ShowMessage('Hatalı Şifre Girdiniz!!');
            Edit2.Text:='';
            end;
      end;
end;

procedure TForm1.BitBtn2Click(Sender: TObject);
begin
Form1.Edit1.Text:='Levent';
Form1.Edit2.Text:='JinX83';
end;

procedure TForm1.FormActivate(Sender: TObject);
begin
Form1.Edit1.Text:='';
Form1.Edit2.Text:='';
end;

procedure TForm1.Edit1Change(Sender: TObject);
begin
if Form1.Edit1.Text='' then
      begin
      Form1.BitBtn1.Enabled:=False;
      Form1.BitBtn2.Enabled:=False;
      end
else
      begin
      Form1.BitBtn1.Enabled:=True;
      Form1.BitBtn2.Enabled:=True;
      end;
end;
```

```
procedure TForm1.FormCreate(Sender: TObject);
begin
Form1.BitBtn1.Enabled:=False;
Form1.BitBtn2.Enabled:=False;
end;

procedure TForm1.Edit2Change(Sender: TObject);
begin
if Form1.Edit2.Text='' then
      begin
      Form1.BitBtn1.Enabled:=False;
      Form1.BitBtn2.Enabled:=False;
      end
else
      begin
      Form1.BitBtn1.Enabled:=True;
      Form1.BitBtn2.Enabled:=True;
      end;
end;

end.
```




```
unit Odalar;

interface
```

```
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, ComCtrls, StdCtrls, OleCtrls, chartfx3, CPort, Menus, CPortCtl;

type
  TForm2 = class(TForm)
    ODA1: TTabSheet;
    StaticText1: TStaticText;
    Bevel1: TBevel;
    StaticText2: TStaticText;
    StaticText3: TStaticText;
    StaticText4: TStaticText;
    StaticText5: TStaticText;
    Shape1: TShape;
    Shape2: TShape;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Genel: TTabSheet;
    StaticText6: TStaticText;
    Shape3: TShape;
    Bevel2: TBevel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Shape4: TShape;
    StaticText7: TStaticText;
    Bevel3: TBevel;
    Shape5: TShape;
    Shape6: TShape;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    ODA2: TTabSheet;
    ODA3: TTabSheet;
    Pages: TPageControl;
    UpDown1: TUpDown;
    UpDown2: TUpDown;
    UpDown3: TUpDown;
    StaticText10: TStaticText;
    Chartfx1: TChartfx;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    StaticText11: TStaticText;
    Label16: TLabel;
    StaticText12: TStaticText;
    StaticText13: TStaticText;
    StaticText14: TStaticText;
    Label17: TLabel;
```

```
StaticText15: TStaticText;
StaticText16: TStaticText;
Label18: TLabel;
Chartfx2: TChartfx;
Panel4: TPanel;
Panel5: TPanel;
Panel6: TPanel;
StaticText17: TStaticText;
StaticText18: TStaticText;
StaticText19: TStaticText;
StaticText20: TStaticText;
StaticText21: TStaticText;
StaticText22: TStaticText;
Label19: TLabel;
Label20: TLabel;
Label21: TLabel;
Chartfx3: TChartfx;
Panel7: TPanel;
Panel8: TPanel;
Panel9: TPanel;
MainMenu1: TMainMenu;
StaticText23: TStaticText;
StaticText24: TStaticText;
Label22: TLabel;
StaticText25: TStaticText;
StaticText26: TStaticText;
Label23: TLabel;
StaticText27: TStaticText;
Label24: TLabel;
StaticText28: TStaticText;
Windows1: TMenuItem;
Genel1: TMenuItem;
ODA11: TMenuItem;
ODA21: TMenuItem;
ODA31: TMenuItem;
Timer1: TTimer;
File1: TMenuItem;
PortSettings1: TMenuItem;
ComPort1: TComPort;
PortA1: TMenuItem;
PortKapat1: TMenuItem;
procedure FormCreate(Sender: TObject);
procedure UpDown1Click(Sender: TObject; Button: TUDBtnType);
procedure UpDown2Click(Sender: TObject; Button: TUDBtnType);
procedure UpDown3Click(Sender: TObject; Button: TUDBtnType);
procedure StaticText1Click(Sender: TObject);
procedure StaticText6Click(Sender: TObject);
procedure StaticText7Click(Sender: TObject);
procedure PagesChange(Sender: TObject);
procedure Button1Click(Sender: TObject);
```

```
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Comport1Click(Sender: TObject);
    procedure Genel1Click(Sender: TObject);
    procedure ODA11Click(Sender: TObject);
    procedure ODA21Click(Sender: TObject);
    procedure ODA31Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure PortSettings1Click(Sender: TObject);
    procedure PortA1Click(Sender: TObject);
    procedure PortKapat1Click(Sender: TObject);
    procedure ComPort1RxChar(Sender: TObject; Count: Integer);
    procedure PagesExit(Sender: TObject);
    procedure ComPort1Rx80Full(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

  TakenRoomTemperature:      string;
  ComData:              string;
  Room:                integer;
  Port:               boolean;
  Port_Uyari:           boolean;
  Packet_Sent:          boolean;
  Grafik_1_ElemanSayisi: integer;
  Grafik_2_ElemanSayisi: integer;
  Grafik_3_ElemanSayisi: integer;
  Grafik_1_Eleman:       integer;

const
  CR = Char(13);
  LF = Char(10);
  Space=Char(32);

implementation

uses Giris;

{$R *.DFM}

procedure TForm2.FormCreate(Sender: TObject);
begin
Form2.Shape1.Brush.Color:=clRed;
Form2.Shape3.Brush.Color:=clRed;
Form2.Shape5.Brush.Color:=clRed;
```

```
Form2.Shape2.Brush.Color:=clBlue;
Form2.Shape4.Brush.Color:=clBlue;
Form2.Shape6.Brush.Color:=clBlue;


//  Set Sıcaklığı
Form2.Label1.Caption:='0';
Form2.Label4.Caption:='0';
Form2.Label7.Caption:='0';


// Sıcaklık
Form2.Label2.Caption:='23';
Form2.Label5.Caption:='23';
Form2.Label8.Caption:='23';

//Işık
Form2.Label3.Caption:='70';
Form2.Label6.Caption:='70';
Form2.Label9.Caption:='70';

Room:=1;
Port:=False;
Port_Uyari:=False;
Packet_Sent:=True;
Form2.Timer1.Enabled:=False;

Grafik_1_ElemanSayisi:=0;
Grafik_1_Eleman:=0;
Grafik_1_ElemanSayisi:=0;
Grafik_1_ElemanSayisi:=0;
TakenRoomTemperature:='10';
end;

procedure TForm2.UpDown1Click(Sender: TObject; Button: TUDBtnType);
var
 Temperature:    integer;
 SetTemperature: integer;
begin
 Temperature:=StrToInt(Form2.Label2.Caption);
 SetTemperature:=StrToInt(Form2.Label1.Caption);

if Button=btNext then
   begin
   SetTemperature:=SetTemperature+1;
   end;
if Button=btPrev then
   begin
   SetTemperature:=SetTemperature-1;
   end;
```

```
 if SetTemperature > Temperature then
       begin
       Form2.Label2.Color:=clBlue;
       Form2.Shape2.Brush.Color:=clBlue;
       end
 else  if   SetTemperature < Temperature then
       begin
       Form2.Label2.Color:=clYellow;
       Form2.Shape2.Brush.Color:=clYellow;
       end
 else
       begin
       Form2.Label2.Color:=clGreen;
       Form2.Shape2.Brush.Color:=clGreen;
       end;

 Form2.Label2.Caption:=IntToStr(Temperature);
 Form2.Label1.Caption:=IntToStr(SetTemperature);
end;

procedure TForm2.UpDown2Click(Sender: TObject; Button: TUDBtnType);
var
 Temperature:    integer;
 SetTemperature: integer;
begin
 Temperature:=StrToInt(Form2.Label5.Caption);
 SetTemperature:=StrToInt(Form2.Label4.Caption);

if Button=btNext then
   begin
   SetTemperature:=SetTemperature+1;
   end;
if Button=btPrev then
   begin
   SetTemperature:=SetTemperature-1;
   end;

 if SetTemperature > Temperature then
       begin
       Form2.Label5.Color:=clBlue;
       Form2.Shape4.Brush.Color:=clBlue;
       end
 else  if   SetTemperature < Temperature then
       begin
       Form2.Label5.Color:=clYellow;
       Form2.Shape4.Brush.Color:=clYellow;
       end
 else
       begin
```

```
        Form2.Label5.Color:=clGreen;
        Form2.Shape4.Brush.Color:=clGreen;
        end;

 Form2.Label5.Caption:=IntToStr(Temperature);
 Form2.Label4.Caption:=IntToStr(SetTemperature);
end;

procedure TForm2.UpDown3Click(Sender: TObject; Button: TUDBtnType);
var
 Temperature:    integer;
 SetTemperature: integer;
begin
 Temperature:=StrToInt(Form2.Label8.Caption);
 SetTemperature:=StrToInt(Form2.Label7.Caption);

if Button=btNext then
   begin
   SetTemperature:=SetTemperature+1;
   end;
if Button=btPrev then
   begin
   SetTemperature:=SetTemperature-1;
   end;

 if SetTemperature > Temperature then
        begin
        Form2.Label8.Color:=clBlue;
        Form2.Shape6.Brush.Color:=clBlue;
        end
 else  if   SetTemperature < Temperature then
        begin
        Form2.Label8.Color:=clYellow;
        Form2.Shape6.Brush.Color:=clYellow;
        end
 else
        begin
        Form2.Label8.Color:=clGreen;
        Form2.Shape6.Brush.Color:=clGreen;
        end;

 Form2.Label8.Caption:=IntToStr(Temperature);
 Form2.Label7.Caption:=IntToStr(SetTemperature);
end;


procedure TForm2.StaticText1Click(Sender: TObject);
begin
 if Form2.Shape1.Brush.Color = clRed then
        begin
```

```
        Form2.Shape1.Brush.Color:=clGreen;
        end
 else
        begin
        Form2.Shape1.Brush.Color:=clRed;
        end;
end;

procedure TForm2.StaticText6Click(Sender: TObject);
begin
 if Form2.Shape3.Brush.Color = clRed then
        begin
        Form2.Shape3.Brush.Color:=clGreen;
        end
 else
        begin
        Form2.Shape3.Brush.Color:=clRed;
        end;
end;

procedure TForm2.StaticText7Click(Sender: TObject);
begin
 if Form2.Shape5.Brush.Color = clRed then
        begin
        Form2.Shape5.Brush.Color:=clGreen;
        end
 else
        begin
        Form2.Shape5.Brush.Color:=clRed;
        end;
end;




procedure TForm2.PagesChange(Sender: TObject);
begin
//Form.Chartfx2.OpenDataEx(1,1,3);
//Form2.Chartfx2.ThisSerie:=0;
//Form2.Chartfx2.Value[0]:=20;
//Form2.Chartfx2.Value[1]:=30;
//Form2.Chartfx2.Value[2]:=10;
//Form2.Chartfx2.CloseData(1);

end;

procedure TForm2.Button1Click(Sender: TObject);
var
temp: integer;
```

```
//length: integer;
begin
Form2.Chartfx1.OpenDataEx(1,1,Grafik_1_ElemanSayisi);
Form2.Chartfx1.ThisSerie:=0;
Form2.Chartfx1.Value[Grafik_1_ElemanSayisi]:=temp;
Form2.Chartfx1.CloseData(1);
Grafik_1_ElemanSayisi:=Grafik_1_ElemanSayisi+1;
end;

procedure TForm2.FormClose(Sender: TObject; var Action: TCloseAction);
begin
Form1.Close;
end;
procedure TForm2.Comport1Click(Sender: TObject);
begin
//Form2.ComPort2.ShowSetupDialog;
//Form2.Comport2.Open;
end;

procedure TForm2.Genel1Click(Sender: TObject);
begin
Form2.Pages.ActivePage:=Genel;
end;

procedure TForm2.ODA11Click(Sender: TObject);
begin
Form2.Pages.ActivePage:=ODA1;
end;

procedure TForm2.ODA21Click(Sender: TObject);
begin
Form2.Pages.ActivePage:=ODA2;
end;

procedure TForm2.ODA31Click(Sender: TObject);
begin
Form2.Pages.ActivePage:=ODA3;
end;

procedure TForm2.Timer1Timer(Sender: TObject);
var
RequestString:  string;
Tus         :   Integer;
begin
RequestString:='<'+'O'+':'+IntToStr(Room)+'>'+CR+LF;
if Port=True then
      begin
      if Packet_Sent=False then
            begin
            Form2.ComPort1.WriteStr(RequestString);
```

```
            Packet_Sent:=True;
            end;
      end
else
      begin
      if Port_Uyari=False then
            begin
            Port_Uyari:=True;
            Tus:=Application.MessageBox('Portu Açınız!!!','Mesaj',mb_YesNo);
            if Tus=6 then
                  begin
                  Form2.ComPort1.ShowSetupDialog;
                  Port:=True;
                  Packet_Sent:=False;
                  Form2.ComPort1.Open;
                  end;
            end;
      end;
end;

procedure TForm2.PortSettings1Click(Sender: TObject);
begin
Form2.ComPort1.ShowSetupDialog;
Form2.ComPort1.Open;
end;

procedure TForm2.PortA1Click(Sender: TObject);
begin
Form2.ComPort1.Open;
Port:=True;
Packet_Sent:=False;
end;

procedure TForm2.PortKapat1Click(Sender: TObject);
begin
Port:=False;
Form2.ComPort1.Close;
end;

procedure TForm2.ComPort1RxChar(Sender: TObject; Count: Integer);
var
Position:      integer;
RequestData:    string;
TakenData:     String;
StartOfData:    integer;
EndOfData:      integer;
Checksum:      byte;
TakenChecksum:  byte;
TakenChecksumChar:  Char;
```

```
  TakenSetTemperature:      string;
  TakenLight:               string;
  TakenWindowStatus:        string;


begin
Form2.ComPort1.ReadStr(TakenData,Count);
ComData:=ComData+TakenData;

RequestData:=CR+LF;
if (Pos(RequestData,ComData)<>0) then
      begin
      RequestData:='<'+'O'+':';
      StartOfData:=Pos(RequestData,ComData);
      RequestData:='C'+':';
      EndOfData:=Pos(RequestData,ComData);

      Checksum:=0;
      while (StartOfData < EndOfData) do
            begin
            Checksum:=Checksum+Ord(ComData[StartOfData]);
            StartOfData:=StartOfData+1;
            end;
       TakenChecksumChar:=ComData[EndOfData+2];
       TakenChecksum:=ord(TakenChecksumChar);
      if Checksum=TakenChecksum then
            begin
            TakenSetTemperature:=Copy(ComData,8,2);
            TakenRoomTemperature:=Copy(ComData,13,2);
            TakenLight:=Copy(ComData,18,2);
            TakenWindowStatus:=Copy(ComData,23,1);
            if ComData[4]='1' then
                  begin
                  Form2.Label1.Caption:=TakenSetTemperature;
                  Form2.Label16.Caption:=TakenSetTemperature+'.0';
                  Form2.Label2.Caption:=TakenRoomTemperature;
                  Form2.Label17.Caption:=TakenRoomTemperature+'.0';
                  Form2.Shape1.Brush.Color:=clGreen;

                  Grafik_1_Eleman:=StrToInt(TakenRoomTemperature);
                  Form2.Chartfx1.PointType:=0;
                  Form2.Chartfx1.OpenDataEx(1,1,Grafik_1_ElemanSayisi);
                  Form2.Chartfx1.ThisSerie:=0;
                  Form2.Chartfx1.Value[Grafik_1_ElemanSayisi]:=Grafik_1_Eleman;
                  Form2.Chartfx1.CloseData(1);
                  Grafik_1_ElemanSayisi:=Grafik_1_ElemanSayisi+1;


                  if (StartOfData > 10) then
                        begin
```

```
                    Form2.Label3.Caption:=TakenLight;
                    Form2.Label18.Caption:=TakenLight;
                    //Form2.Label3.Caption:=TakenWindowStatus;
                    //Form2.Label18.Caption:=TakenWindowStatus;
                    end;

                if TakenSetTemperature > TakenRoomTemperature then
                    begin
                    Form2.Label2.Color:=clBlue;
                    Form2.Shape2.Brush.Color:=clBlue;
                    end
                else  if TakenSetTemperature<TakenRoomTemperature then
                    begin
                    Form2.Label2.Color:=clYellow;
                    Form2.Shape2.Brush.Color:=clYellow;
                    end
                 else
                    begin
                    Form2.Label2.Color:=clGreen;
                    Form2.Shape2.Brush.Color:=clGreen;
                    end;
                end
            else if ComData[4]='2' then
                begin
                Form2.Label4.Caption:=TakenSetTemperature;
                Form2.Label19.Caption:=TakenSetTemperature+'.0';
                Form2.Label5.Caption:=TakenRoomTemperature;
                Form2.Label20.Caption:=TakenRoomTemperature+'.0';
                Form2.Label6.Caption:=TakenLight;
                Form2.Label21.Caption:=TakenLight;
                //Form2.Label3.Caption:=TakenWindowStatus;
                //Form2.Label18.Caption:=TakenWindowStatus;

                //Form2.Chartfx2.OpenDataEx(1,1,Grafik_2_ElemanSayisi);
                //Form2.Chartfx2.ThisSerie:=0;

//Form2.Chartfx2.Value[Grafik_2_ElemanSayisi]:=StrToInt(TakenRoomTemperature);
                //Form2.Chartfx2.CloseData(1);
                //Form2.Chartfx2.PointType:=0;
                //Grafik_2_ElemanSayisi:=Grafik_2_ElemanSayisi+1;
                 if TakenSetTemperature > TakenRoomTemperature then
                    begin
                    Form2.Label5.Color:=clBlue;
                    Form2.Shape4.Brush.Color:=clBlue;
                    end
                 else  if TakenSetTemperature<TakenRoomTemperature then
                    begin
                    Form2.Label5.Color:=clYellow;
                    Form2.Shape4.Brush.Color:=clYellow;
                    end
```

```
                    else
                        begin
                        Form2.Label5.Color:=clGreen;
                        Form2.Shape4.Brush.Color:=clGreen;
                        end;
                    end
                else if ComData[4]='3' then
                    begin
                    Form2.Label7.Caption:=TakenSetTemperature;
                    Form2.Label22.Caption:=TakenSetTemperature+'.0';
                    Form2.Label8.Caption:=TakenRoomTemperature;
                    Form2.Label23.Caption:=TakenRoomTemperature+'.0';
                    Form2.Label9.Caption:=TakenLight;
                    Form2.Label24.Caption:=TakenLight;
                    //Form2.Label3.Caption:=TakenWindowStatus;
                    //Form2.Label18.Caption:=TakenWindowStatus;


                    //Form2.Chartfx3.OpenDataEx(1,1,Grafik_3_ElemanSayisi);
                    //Form2.Chartfx3.ThisSerie:=0;

//Form2.Chartfx3.Value[Grafik_3_ElemanSayisi]:=StrToInt(TakenRoomTemperature);
                    //Form2.Chartfx3.CloseData(1);
                    //Form2.Chartfx3.PointType:=0;
                    //Grafik_3_ElemanSayisi:=Grafik_3_ElemanSayisi+1;
                     if TakenSetTemperature > TakenRoomTemperature then
                        begin
                        Form2.Label8.Color:=clBlue;
                        Form2.Shape6.Brush.Color:=clBlue;
                        end
                     else  if TakenSetTemperature<TakenRoomTemperature then
                        begin
                        Form2.Label8.Color:=clYellow;
                        Form2.Shape6.Brush.Color:=clYellow;
                        end
                     else
                        begin
                        Form2.Label8.Color:=clGreen;
                        Form2.Shape6.Brush.Color:=clGreen;
                        end;
                    end;
                end;

        Packet_Sent:=False;
        ComData:='';
        end;
end;
```

```
procedure TForm2.PagesExit(Sender: TObject);
begin
Form2.ComPort1.Close;
end;

procedure TForm2.ComPort1Rx80Full(Sender: TObject);
begin
Form2.ComPort1.Buffer.CleanupInstance();
end;

end.
```

ÖZGEÇMİŞ

AD : LEVENT
SOYAD: BİRGÜL

e-Mail : leventbirgul@hotmail.com
leventbirgul@yahoo.com
Adres : A. Nafız Gürman Mah. Muratlı Sok. Kayhan Apt.
No:27 Kat 7 Daire 15 34173
Merter/İSTANBUL
Telefon : (0212) 556 11 56
GSM : (0533) 560 43 82

Doğum Tarihi: 9 Ekim 1983
Doğum Yeri : Fatih/İSTANBUL
Medeni Hali : Bekar

Ehliyet : B sınıfı.

Eğitim:

1998-2001	Pertevniyal Anadolu Lisesi	- Aksaray/İstanbul
2001-2005	Yıldız Teknik Üniversitesi-Elektrik Mühendisliği Bölümü	- Beşiktaş/İstanbul
2005-	YTU-Elektrik Müh. Kontrol Anabilim D. Yük. Lis. Prog.	- Beşiktaş/istanbul

Eğitimde Alınan Dereceler:
Pertevniyal Anadolu Lisesi Mezuniyet Derecesi II.(İkincilik)

Yabancı Dil:
İngilizce (Advanced Level)

Bilgisayar Bilgisi:
C Programlama Dili
Dephi Programlama Dili
Assembler Programlama Dili

Staj Bilgisi:

2003	Park Plaza Teknik Servis - Otomasyon Bölümü	- Maslak/İstanbul
2004	Elektro Elektronik San. ve Tic. A.S.	- Altunizade/İstanbul

İş Bilgisi:

2003-2004	Elektrik Mühendisliği Bölümü-Kontrol Kumanda Anabilim dalında asistan öğrenci
2005	Tüm Elektronik Mühendislik San. ve Tic. Ltd. Şti. –Arge

Lisans Proje-1 **:** 8051 Microcontroller.
Lisans Proje-2 ve Bitirme Tezi **:** RF Uygulaması ve Q- Matik Sistem Uygulaması

Daha önce projesi gerçeklenmiş mikroişlemciler:
Motorola MC68HC908GP32, MC68HC908LK24, Analog Devices ADuC Ailesi, DallasDS500T,8051/52 Türevi microdenetleyiciler, Cypress CY212xx Ailesi, Renesas R8C, M16C, ARM7 STR711F