T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MOTOR KONTROLLÜ GÜNEŞ İZLEME SİSTEMİ

Eyyüp ÖKSÜZTEPE

YÜKSEK LİSANS TEZİ

ELEKTRİK EĞİTİMİ ANABİLİM DALI

ELAZIĞ

1998

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MOTOR KONTROLLÜ GÜNEŞ İZLEME SİSTEMİ

Eyyüp ÖKSÜZTEPE

YÜKSEK LİSANS TEZİ
ELEKTRİK EĞİTİMİ ANABİLİM DALI

Bu tez, ... Tarihinde, Aşağıda Belirtilen Jüri Tarafından Oybirliği / Oyçokluğu ile başarılı / Başarısız Olarak Değerlendirilmiştir.

Prof. Dr. Ata SELÇUK	Doç. Dr. Mustafa POYRAZ	Yrd. Doç Fikret ATA
Danışman	Üye	Üye

Bu tezin kabulü, Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ...../ ...../ ..... tarih ve ..... sayılı kararıyla onaylanmıştır.

ÖZET

YÜKSEK LİSANS TEZİ

MOTOR KONTROLLÜ GÜNEŞ İZLEME SİSTEMİ

Eyyüp ÖKSÜZTEPE

Fırat Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik Eğitimi Anabilim Dalı

1998, Sayfa:83

Güneş enerjisinin depolanması güneş kollektörleriyle olmaktadır. Gün boyunca maksimum güneş enerjisi elde etmek için, güneşin kollektörler tarafından takip edilmesi gerekir. Bu sebepten, bilgisayar programı ile güneşi takip edebilecek mekanik bir sistem yapılmıştır. Sistem küresel hareket ederek, güneşi takip edebilmektedir.

Anahtar kelimeler: Güneş enerjisi, güneş izleme sistemi, fotovoltaik sistem

ABSTRACT

Master Thesis

MOTOR CONTROLLED SOLAR TRACKİNG SYSTEM

Eyyüp ÖKSÜZTEPE

Fırat Üniversity
Graduate School of Natural and Applied Sciences
Department of Electrical Education

1998, Page:83

Storing the solar energy is beening by solar collactors. To obtain the maximum solar energy all day long, it is necessary to follow the sun by the collactors. For this reason, a mechanical system have been constructed, which can follow the sun by a computer program. The system can be followed the sun by spherical motion.

KeyWords: Solar energy, solar tracking system, photovoltaic system.

TEŞEKKÜR

Bu tez çalışmasında her türlü yardımı esirgemeyen başta değerli danışmanım Prof. Dr. Ata SELÇUK' a, ve Yrd. Doç. Dr. Muktim ERDOĞMUŞ, Yrd. Doç. Fikret ATA, Doç. Dr. Mustafa POYRAZ'a teşekkür ederim.

İÇİNDEKİLER

ŞEKİL LİSTESİ

1. GİRİŞ

1.1 Güneş Pili Sistemlerinin Çeşitleri

Güneş pilleri, elektrik enerjisinin gerekli olduğu her uygulamada kullanılabilir. Güneş pili modülleri uygulamaya bağlı olarak, akümülatörler, inverterler, akü şarj kontrol cihazları ve çeşitli elektronik destek devreleri ile birlikte kullanılarak bir güneş pili sistemi (FV sistem) oluştururlar. Bu sistemler, özellikle yerleşim yerlerinden uzak, elektrik şebekesi olmayan yörelerde, jeneratöre yakıt taşımanın zor ve pahalı olduğu durumlarda kullanılırlar. Bunun dışında dizel jeneratörler ya da başka güç sistemleri ile birlikte karma olarak kullanılmaları da mümkündür.

Güneş pili sistemi uygulamaları iki ana gruba ayrılabilir:

- Şebeke bağlantılı sistemler
- Bağımsız sistemler

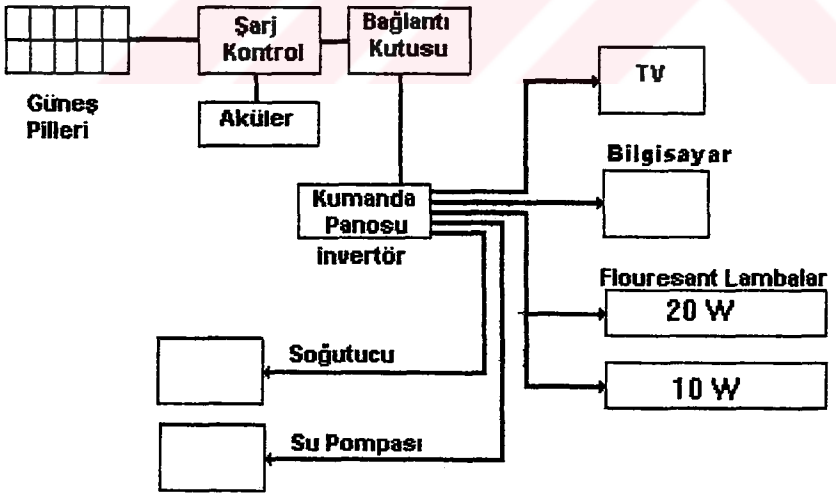
1.1.1 Şebeke Bağlantılı Güneş Pili Sistemleri

Şebeke bağlantılı güneş pili sistemlerinin gücü, birkaç kW 'dan birkaç MW 'a kadar değişebilmektedir. Bu tür sistemler, iki ana gruba ayrılır. İlk tür sistem, temelde bir yerleşim biriminin mesela, bir konutun elektrik ihtiyacını karşılar. Bu sistemlerde, üretilen fazla enerji elektrik şebekesine satılır, yeterli enerjinin üretilmediği durumlarda ise şebekeden enerji alınır. Böyle bir sistemde enerji depolaması yapmaya gerek yoktur, yalnızca üretilen d.a. elektriğin, a.a. elektriğe çevrilmesi ve şebeke uyumlu olması yeterlidir. İkinci tür şebekeye bağlı güneş pili sistemleri kendi başına elektrik üretip, bunu şebekeye satan büyük güç üretim merkezleri şeklindedir. Bunların büyüklüğü 600-700 kW 'dan MW 'lara kadar değişir.

1.1.2 Bağımsız Güneş Pili Sistemleri

FV sistemlerinin en tipik ve en yaygın kullanım şekli, yerleşim yerlerinden uzak yörelerde enerji gereksinimini karşılayan bağımsız (stand-alone) sistemlerdir. Bu sistemler birkaç Watt 'dan birkaç yüz kW 'a kadar değişebilen güçlerde ve çok çeşitli türlerde yüklerin enerji talebini karşılayabilir.

Bu tür sistemlerde yeterli sayıda güneş pili modülü, enerji kaynağı olarak kullanılır. Güneş 'in yetersiz olduğu zamanlarda ya da özellikle gece süresince kullanılmak üzere genellikle sistemde akümülatör bulundurulur. Güneş pili modülleri gün boyunca elektrik enerjisi üreterek bunu akümülatörde depolar, yüke gerekli olan enerji akümülatörden alınır. Akünün aşırı şarj ve deşarj olarak zarar görmesini engellemek için kullanılan kontrol birimi ise akünün durumuna göre, ya güneş pillerinden gelen akımı ya da yükün çektiği akımı keser. Şebeke uyumlu alternatif akım elektriğinin gerekli olduğu uygulamalarda, sisteme bir inverter eklenerek akümülatördeki d.a. gerilim, 220 V, 50 Hz'lik sinüs dalgasına dönüştürülür. Şekil 1.1 'de alternatif akımla çalışan yüklere enerji sağlayan tipik bir güneş pili sisteminin şeması verilmiştir.



Şekil 1.1 Bağımsız güneş pili sistemi.

Bağımsız güneş pili sistemlerinin kullanıldığı tipik uygulama alanları aşağıda sıralanmıştır.

- Radyolink istasyonları, kırsal radyo, telsiz ve telefon sistemleri
- Petrol boru hatlarının katodik koruması, metal yapıların (köprüler, kuleler vb.) korozyondan korunması
- Elektrik ve su dağıtım sistemlerinde yapılan telemetrik ölçümler, hava gözlem istasyonları
- Bina içi ya da dışı aydınlatma
- Dağ evleri ya da yerleşim yerlerinden uzaktaki evlerde TV, radyo, buzdolabı gibi elektrikli aygıtların çalıştırılması
- Tarımsal sulama ya da ev kullanımı amacıyla su pompajı
- Orman gözetleme kuleleri
- Deniz fenerleri
- İlkyardım, alarm ve güvenlik sistemleri
- İlaç ve aşı soğutma

Güneş pili sistemleri en çok iletişim alanında kullanılmaktadır. Radyolink istasyonlarının çoğunlukla elektriği bulunmayan yüksek ve ulaşım sorunu olan yörelerde kurulu olması nedeniyle, bu tesislerde güneş pili modülleri kullanmak uygun bir çözüm olmuştur. Bu alanı, su pompajı ve aşı-ilaç koruma izlemektedir.

Bağımsız ve şebeke bağlantılı sistemlerin dışında, güneş pilleri, uzay uygulamaları ve tüketim ürünlerinde kullanılmaktadır. Uzayda 1958 yılından bu yana, uyduların ve uzay araçlarının elektrik ihtiyacını karşılamak için güneş pilleri kullanılmaktadır. Tüketiciye yönelik ürünlerin ise en tipik örneği, yıllardır ticari ortama kabul edilmiş olan güneş pili ile çalışan hesap makineleridir.

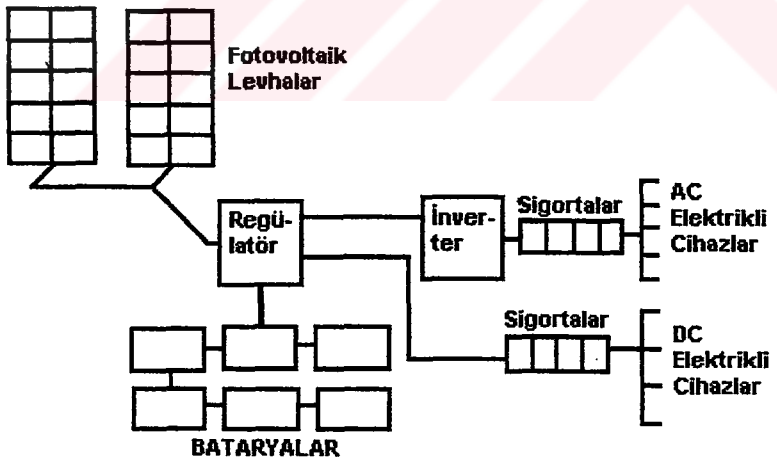
1.2 Sistemin tasarımı

Herhangi bir yere foto voltaik sistem kurulmadan önce, bazı incelemelerin yapılması gerekir. Bu incelemeler kurulacak sistemin üstlendiği görevi başka enerji kaynaklarıyla yapılması durumunda, foto voltaik sisteme göre daha pahalı olup olmadığı sonucunun

çıkarılması ile ilgilidir. Eğer sonuç gerek işletme bakımından, gerekse bakım ve onarımın daha kolay, daha ekonomik olması yönündeyse bu sistemin kurulmasına geçilir.

Sistemin nereye kurulacağı, güneş enerjisi yönünden Meteoroloji Müdürlüklerinden alınan bilgiler son derece önemlidir. Bunun yanı sıra Sistemin besleyeceği alıcıların gücü ve günlük ne kadar süreyle çalıştırılacağı, sistemde kullanılan foto voltaik levha adedinin tayininde önemli rol oynamaktadır. Güneşin aylara göre enerjisiyle, kullanılan levhalarda ne kadar enerji üretildiği hesaplanır. Üretilen enerji, özellikle yazın güneşin bol olduğu günlerde ihtiyaçtan fazladır. Bunun yanı sıra güneşin olmadığı günlerde de üretilen enerji ihtiyacı karşılanamaz. Bol güneşli günlerde üretilen fazla enerji, akümülatörlerle depo edilip,ihtiyacın karşılanamadığı zamanlarda tüketilir. Akümülatör adedi üretilen toplam net enerji miktarını depolayabilecek sayıda olmalıdır. Bataryaların şarj durumunu kontrol etmek için regülatör kullanılır. Regülatörün akım kapasitesi, alıcıların akümülatörlerden çektikleri akıma göre bulunur. Foto voltaik levhalar doğru akım ürettiklerinden, alternatif akım ihtiyacı inverter tarafında oluşturulur. Genellikle inverter giriş gerilimi 12 volt,çıkış gerilimi 225 volt alternatif akımdır. Alıcıların gücünü karşılayabilecek inverter kullanılır.

Küçük bir FV sistemi Şekil 1.2 'de gösterilen bölümlere ayırabiliriz.



Şekil 1.2 Küçük bir FV sistem tasarımı

FV levhalar güneş enerjisini direkt olarak enerjisine dönüştürürler. Tipik bir levha güneşli açık havada 12 Volt, 10 Amper kadar, yani 120 Watt elektrik üretebilir. Elde edilen gerilimi artırmak için levhalar seri olarak, akımı artırmak için ise paralel olarak bağlanırlar. Güneşten maksimum enerjiyi toplayabilmek için FV levhaların gün boyunca en çok güneş gören güney yönüne bakmaları ve bulunan eyleme göre zamana bağlı olarak yatay ile belirli bir eğilimde olmaları gerekir. Genel olarak kış aylarında levha yaz aylarına nisbeten daha dikey olmalıdır.

Güneş enerjisi değişen ve her zaman olmayan bir enerji türüdür. Mesela, güneş doğmadan önce, güneş battıktan sonra veya kapalı ve bulutlu havalarda güneş enerjisi olmadığından toplanan fazla enerjinin depolanıp bu zamanlarda kullanılması gerekir. Bu amaçla yüksek kapasiteli (mesela 100 Ah) batarya kullanılır. Genel olarak bir bataryanın ömrünü artırmak için kapasitesinin %80 'den fazla deşarj olmaması gerekir.

FV sistemlerde güneş olduğu zamanlarda bataryaların tamamıyla dolduktan sonra akım almalarını (overcharge) önlemek gerekir. Fazla şarj bataryanın ısınmasına, sıvı kaybına ve batarya ömrünün kısalmasına yol açar. Regülatör, FV levhalar ile bataryalar arasına konur ve bataryaları fazla şarj almalarını önler.

İnverter, 12 veya 24 voltluk düşük doğru akımı 240 Volt alternatif akıma dönüştürür. Çok küçük uygulamalarda inverter yerine düşük gerilim ve doğru akımla çalışan elektrikli cihazlar kullanmak mümkündür.

Bir FV sistem tasarımı yapmazdan önce ilk olarak sistemin kurulacağı bölgedeki aylara göre dağılmış, metre kareye düşen ortalama günlük güneş enerjisini bilmek gerekir. Gerekli olan bu bilgi normal olarak bölgeye en yakın olan meteoroloji istasyonundan elde edilebilir.

FV sistem tasarımı yapılırken ilk olarak sistemin kurulacağı yerdeki elektrik tüketimini hesaplamak gerekir. Bunun içinde kullanılacak olan elektrikli cihazların enerji harcamalarını (güç harcamaları ve kullanım zamanları) bilmeliyiz. FV sistem tasarımları genel olarak enerji eşitliği esas alınarak yapılır. Bu çalışmada sunulan tasarımda aylık enerji eşitliği esas alınmıştır.

1.3 Güneş Pili Sistemlerinin Ekonomisi

Güneş pili sistemlerinin enerji maliyetini üç önemli etken belirler. Bunlar; pil verimi, sistemin ilk yatırım maliyeti ve ömrüdür.

1.3.1 Verim

Pil veriminin maliyet üzerinde doğrudan bir etkisi vardır. Bu verimin artırılmasıyla güneş pili sistemlerinin maliyeti azalacaktır. Daha gelişmiş teknolojiler kullanılarak gelecekte pil verimlerinin %20 'ler mertebesine çıkarılacağı umulmaktadır.

1.3.2 Yatırım maliyeti

Güneş pili sistemlerinin işletme ve bakım maliyetleri çok az olduğu için, toplam sistem maliyetinin büyük bir kısmını ilk yatırım maliyeti oluşturur. Üretim teknolojisinin geliştirilmesi, yüksek verimli pillerin yapılması, modül tasarım ve yapım tekniklerinin geliştirilmesi ile ilk yatırım maliyeti azalacaktır. Güneş pili sistemlerinin ilk yatırım maliyetleri arasında arazi, tesisat, montaj, inverter, ve diğer güç cihazları gibi destek elemanların maliyeti de yer alır. Destek sistemlerin maliyeti, bir güneş pili sisteminin maliyetinin yaklaşık yarısını oluşturduğu için, bu tür maliyetleri azaltmak, en az modül maliyetini azaltmak kadar önem taşır.

1.3.3 Modül ömrü

Silisyum kristal piller için bu etken fazla önem taşımaz, çünkü bu pillerde, hedeflenmiş olan 30 yıllık ömre ulaşılmıştır. Amorf silisyum ve diğer güneş pili türlerinde zamanla güç çıkışı bozularak azaldığı için, ömür daha önemlidir. Modül ömrünün artmasının, enerji maliyetleri üzerinde önemli etkisi olacaktır.

Bir güneş pili sisteminin ürettiği enerjinin maliyeti, depolama yapılmadığı zaman 0.3-0.4 $/kWh arasındadır. Bu maliyetle güneş pili sistemleri, enterkonnekte şebekenin olmadığı veya ulaşımın zor ve pahalı olduğu bölgelerde diğer alternatif enerji kaynakları ile

yarışabilir düzeydedir. Bu gibi yerlerde bir kaç kW 'a kadar küçük güçteki uygulamalar (iletişim, ilaç-aşı soğutma, su pompajı ve aydınlatma gibi), teknolojik açıdan olduğu kadar ekonomik açıdan da kendini kanıtlamıştır. Ancak istenen hedef, fiyatları enterkonnekte elektrik düzeyinde olmasıdır. 2000 'li yıllara kadar maliyetin şebeke elektriği ile yarışabilecek düzeye geleceği umulmaktadır.

1.4 Güneş Pili Sistemlerinin Üstünlükleri

Güneş pilleri dayanıklı, güvenilir ve uzun ömürlüdür. Çalışmaları sırasında hiçbir elektriksel sorun çıkarmazlar ve bozulmazlar. Güneş pili modüllerinin karşı karşıya kalabilecekleri en büyük tehditler, yıldırım düşmesi ve uzun dönemde (20 yıl) hava koşullarından dolayı aşınmadır.

Güneş pilleri modüler yapıdadır, uygun şekilde düzenlenerek 1 V 'tan, birkaç kV 'a kadar çıkış verebilirler, çok küçük güç ihtiyaçlarını karşılayabildikleri gibi, kendi başlarına bir güç santralı olarak da çalışabilirler.

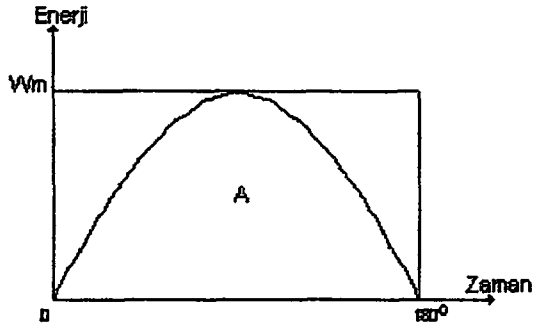
İlk yatırım maliyetlerinin fazla olması güneş pili sistemlerinin en büyük dezavantajıdır. Elektrik şebekesinin olduğu yerlerde güneş pilinin kullanılması uygun değildir. Ancak elektrik hattı bulunmayan veya elektrik götürülmesinin pahalı olduğu kırsal yörelerde güneş pillerinin kullanımı daha ekonomik olabilmektedir.

Güneş pili sistemlerinin en fazla üstünlük gösterdiği alanlardan biri de, tıpkı bütün diğer yenilenebilir enerji kaynakları (rüzgar, biyomas, hidrolik, termal güneş, jeotermal) gibi çevre açısından olumsuz etkilere sahip olmamasıdır.

1.5 Güneş Pilinden Maksimum Enerji Alımı

Güneş pillerinden maksimum enerji, güneş fotonlarının güneş pili yüzeyine dik gelmesi sonucu maksimum sayıda elektron boşluk çifti oluşturması ile sağlanabilir. Sabit olarak monte edilen güneş pili, sadece öğle vakileri dik güneş ışınlarına maruz kalır. Diğer vakitler ise güneş ışınlarının ancak dik bileşenini elektrik enerjisine dönüştürebilir. Güneş ışınlarının yatay bileşenlerinden faydalanılamaz.

Güneş ışınlarının tamamından faydalanmak için güneş pilinin güneşe sürekli dik tutulması gereklidir. Bu olay güneş izleme sistemleri ile gerçekleştirilir.



Şekil 1.3 Güneş enerji grafiği

Şekil 1.2' de güneşin enerji grafiği yaklaşık olarak sinüs eğrisine benzemektedir. A bölgesi olarak gösterilen alan, güneşi takip etmeyen fotopil' in maruz kaldığı enerji alanını göstermektedir. Dış dikdörtgen alanda fotopil'in güneşe sürekli yönlendirilmesi sonucu maruz kalınan enerji miktarını gösterir. Buna göre A alanı,

$dW = Wm*dl \qquad dl = \text{Sin}\,\alpha * d\,\alpha$

$W = \int_0^{180} Vm*\text{Sin}\,\alpha * d\,\alpha = -Wm*\text{Cos}\,\alpha \Big|_0^{180} = 2Wm$ olur.

Güneş pilinin güneşi takip ettiği durumda bir gün boyunca maruz kaldığı enerji ise,

$W = Wm*\pi \qquad W = 3.14\ Wm$ olur.

Enerji oranı = (3.14Wm) / (2Vm) = 1.57

Görüldüğü üzere güneş pilinin güneşi takip etmesiyle elde edilen enerji, sabit zemine monte edilen fotopile göre yaklaşık 1.57 kat fazla olmaktadır.

1.6 Ülkemizde Güneş Pilleri İle İlgili Çalışmalar

Özellikle gelişmiş ülkelerce, güneş pilleri konusunda sürdürülen Ar-Ge ve uygulama çalışmaları paralelinde, ülkemizde de bu konularda başta Elektrik İşleri Etüt İdaresi Genel Müdürlüğü (EİE) olmak üzere kamu kurum ve kuruluşları ile bazı üniversitelerde çalışmalar sürdürülmektedir.

EİE bünyesinde yürütülen, bu konudaki uygulama, Ar-Ge ve demonstrasyon çalışmaları aşağıda özetlenmiştir:

- Güneş pilleri konusunda, mevcut teknolojiyi takip edebilmek amacıyla, 1983 yılında Türkiye Atom Enerjisi Kurumu (TAEK) ile EİE ortak bir proje yapmıştır. Bu proje sonucunda 'Metal-Yalıtkan-Yarıiletken (MIS)' türü 2 Wp gücünde güneş pili paneli yerli imkanlarla laboratuvar şartlarında imal edilmiştir.
- 1985 yılında ise yaklaşık 1,2 kWp gücünde bir güneş pili sistemi projelendirilerek, EİE tesislerinde kurulmuştur. İşletme çalışmaları sürdürülmekte olan bu sistemin üretim sonuçlarının analizleri 1986 yılı içerisinde tamamlanmıştır.
- Güneş pillerinin, ülkemizde küçük ölçekli zirai sulamada kullanım imkanlarının araştırılması amacı ile 1988 yılında yurtdışından güneş pili su pompaj sistemi getirtilerek Atatürk Orman Çiftliği arazisine tesis edilmiştir. Bu sistem, 14 adet güneş pili modülü (her biri 44 W), dalgıç pompa ve elektronik kontrol devresinden oluşmakta olup 616 Wp gücündedir. Sistemin performansının belirlenebilmesi için, pompalanan su miktarı ve modül düzlemindeki toplam güneş enerjisi bir yıl boyunca ölçülmüş ve sistemin bilgisayar modeli hazırlanmıştır.
- Elektrik enerjisinin götürülemediği küçük yerleşim birimlerinde çevre aydınlatılmasında kullanılmak üzere güneş enerjisi ile çalışan bağımsız bir birim geliştirilmiştir. Bu birim, iki adet güneş pili modülü (her biri 48 W), 18 W gücünde alçak basınçlı sodyum buharlı lamba, inverter, regülatör, 2 V - 65 Ah kuru akü ve direkten oluşmaktadır. Bu proje kapsamında 5 aydınlatma birimi geliştirilerek, 2 adedi AOÇ 'ndeki Atatürk evi önüne, 1 adedi AOÇ su pompaj sistemi yanına ve 2 adedi de EİE tesislerine monte edilmiştir.

Araştırma kuruluşları (TÜBİTAK-MARMARA) ve bazı üniversitelerimizde (Ege Üniversitesi, ODTÜ, İTÜ, Hacettepe Üniversitesi, Çukurova Üniversitesi ve Gazi Üniversitesi) güneş pilleri ile ilgili olarak

- Su pompajı
- Elektrik üretimi
- Laboratuvar koşullarında imalat

gibi konularında araştırma, geliştirme ve demonstrasyon çalışmaları sürdürülmektedir.

1.7 Dünyadaki Çalışmalar

Dünyadaki güneş enerjisi ile ilgili son çalışmalar ve örnekler aşağıda sıralanan biçimde özetlenebilir.

- 9 Ülkeden oluşan Uluslararası Enerji Örgütü (IEA) Güney İspanya'da Almeria yakınında Small Solar Power System (SSPS) adını taşıyan bir girişimde bulunmuştur. Bu proje her biri 0,5 MW gücünde farklı teknoloji ile kurulan iki güneş santralını kapsamaktadır. Santralların yapımı 30,4 milyon dolar'a malolmuştur.
- ABD'de 10 MW gücündeki 'Solar One' adlı güneş santralının yapımı Barstow, California'da sürmektedir. Gelecek yıl Haziran ayında işletmeye alınacak olan Solar One, başarılı olduğu takdirde 100 MW 'lık 'Solar Two' projesine başlanacaktır.
- Japonya 'da güneş pilleriyle elektrik üretecek olan 1 MW gücündeki ilk büyük fotovoltaik (FV) jeneratörün yapımına başlanmıştır. Sistemin yapım maliyeti, alışılmış termik santralların 30 katı, nükleer santralların 15 katı, hidrolik sistemlerin ise 9 katına ulaşmaktadır. Ancak bu aşamada maliyet önemli bir etken olmamakta FV sistemlerin çalışabilirliğinin denenerek tespit edilmesi ilk hedef olarak görülmektedir.
- Alman ve Yunan hükümetlerinin anlaşmasıyla Atina yakınında kurulacak olan 'güneş köyü' 435 binadan oluşacak ve binaların ısıtma, sıcak su ve elektrik ihtiyacı güneş enerjisinden sağlanacaktır.

- ABD 'de güneş enerjisiyle çalışan buzdolabı yapılmıştır ve fiyatı 1000 dolar'dır. SSCB ve Çin Halk Cumhuriyeti 'nde ise güneşle çalışan mutfak fırınlarının üretimine geçilmiştir.
- İsrail, güneş endüstrisini yaklaşık 30 yıldır geliştirmektedir. Şimdiden tüm konutların %30 'u güneşle ısıtılmış sıcak su kullanmakta, yeni binaların ise güneşli ısıtıcı ve soğutucu teçhizat bulundurmaları gerekmektedir. Ayrıca deniz suyunun damıtılarak kullanılabilir suya dönüştürme tesislerinin en fazla olduğu ülke İsrail'dir. Güneş enerjisi sanayii giderek çok uluslu petrol şirketlerinin kontrolüne girmektedir. 1979 yılında petrol şirketlerinin güneşten enerji elde eden çalışmalar için ayırdıkları toplam miktar 80 milyon doları aşmıştır. Atlantik kıyısı petrol kaynaklarını işleten Atlantik Richfield (ARCO) şirketi bu yıl güneş enerjisine 25 milyon dolar yatırarak en büyük özel sektör kuruluşu niteliğini kazanmıştır. Güneş enerjisi sanayii'nde söz sahibi Daystar ve Solar Power şirketlerinin en büyük hissedarı çok uluslu EXXON petrol şirketidir. Güneş pili üretiminde dünyanın lideri olarak kabul edilen SOLAREX şirketinin % 20,9 hissesine ise Standard Oil of İndiana (AMOCO) sahip bulunuyor.

Hükümetlerin Güneş enerjisi için ayırdıkları ödeneklerden en büyük payı da dev şirketler almakta ve bu alanda faaliyet gösteren küçük şirketleri satın alarak, büyük yatırımlara girmektedirler. Petrol fiyatlarının hızla yükselmesi sonucu elde edilen karları, geleceğin muhtemel enerji alternatiflerinin araştırılması ve geliştirilmesine ayıran petrol şirketleri böylece hem enerji darlığının önüne geçmekte, hem de geleceğin enerji sektörünü kontrolleri altına almaktadırlar.

2 GÜNEŞİN HAREKETİ

2.1 Günlük Hareketler:

Genel olarak dünyanın kendi ekseninde dönmesi sonucu günlük hareketler oluşur. Küresel gökbilim ilk olarak gökyüzü yuvarlağını bir küre olarak kabul eder. Gözlemci bu kürenin merkezindedir. Kimi zaman da gözlemcinin yeri Yer merkezi olur. İlk bakışta bu doğru sayılmaz. Fakat sonsuz yarıçaplı gök küresi içinde yerin bir nokta sayılabileceği göz önüne alınırsa, yerine göre kullanılacak olan bu varsayımda yanlış sayılmaz.

Herkesin bildiği çevren (ufuk) genel olarak gökküresinin görülen ve görülmeyen parçasının ara cizgisidir. Büyük bir denizin ortasında bulunan bir gözlemci için bu ara çizgi bir çemberdir. Daha basit olarak Yer ile göğün birleştiği çizgiye "çevren" denir. Bu bizim gördüğümüz çevrendir. Gökbilimde çevren şöyle tanımlanır. Gözlemcinin bulunduğu noktadaki çekül doğrultusuna dik olan düzlemin gökküresi ile arakesitine gökbilim çevren denir. Buna çevren çemberi diyebiliriz. Kimi yerde, yer küresine gözlem noktasından çizilen teğet düzlemin arakesiti olarak da tanımlanmaktadır. Yerin tam küre olması nedeniyle bu tanımın özürlü olduğu söylenebilir. Yer ile göğün birleştiği çevren çizgisi ise gökbilim çevrenin birazcık altına düşer, gözlem noktasının yüksekliğine ve de çevresine bağlı olur. Onun için o da bizim için yarayışlı olmaz.

Çekül doğrultusu gök yuvarlağını iki noktada keser. Bunlardan biri tam tepemizdedir. Ona "zenit noktası" ya da kısaca "zenit" denir. Öbür nokta gökküresinin göremediğimiz alt yakasında ve zenitin tam karşısında bulunur.

Günlük hareket, gökyüzüne serpilmiş olan yıldızların topluca yaptığı dönme hareketidir. Her gün yıldızlar doğmakta, ortak bir eksene göre birer çember yayı çizmekte ve sonra batmaktadırlar. Güneş ay ve gezegenler biraz ayrıcalık göstermekle beraber onlarda yine bu toplu harekete katılmaktadırlar. Çağlar boyunca bu hareketin yer ile bir ilişkisi olmadığı, tersine Yer'in durağan olduğu, bütün gökcisimlerin böylece yer çevresinde döndüğü düşüncesi süre geldi. Ancak 16. Yüzyıl başlarında G.Galile bu

düşüncenin yanlış olduğunu ve dünyanın döndüğünü öne sürdü. Aynı yüzyılın ortalarında fizikçi L.Foucault uzun bir sarkaç deneyi ile yerin döndüğünü gözler önüne serdi.

İster yer dönsün, ister gök dönsün, bu dönemin bir tek ekseni vardır. Bu eksenin yer Küresini deldiği noktalara "yerin uçlakları", gökküresini deldiği noktalara da "göğün uçlakları" denir. Gök uçlaklarından biri bizim ülkelerden görülür. Ona "kuzey uçlak", öbürüne de "güney uçlak" denir. Küçük ayı takım yıldızlarının en ucunda bulunan orta parlaklıkta bir yıldız var. Bu yıldız kuzey uçlağın çok yakınında, bu demektir ki neredeyse dönme ekseni üzerine rastlanmaktadır. Onun için bütün kuzey ülkelerinden bu yıldız durağan bir nokta olarak görülmekte, onun yöresindeki bütün yıldızlar o yıldızın çevresinde dönmektedir. Bundan dolayı eski denizciler ona "demir kazık" yıldızı demişlerdir.

Dönme eksenine dik olan büyük çembere "gök eşleği" denir. Onun belirttiği düzlem "eşlek düzlemi" dir. Bu düzlemin Yer küresiyle arakesiti "yer eşleği" olur.

Çevren çemberi ile gök eşleğinin kesim noktaları gözlemcinin doğu ve batı noktalarıdır. Günlük hareket yönünde (saat yönü) alttan üste geçerken rastlanan nokta doğudur. O halde tam eşlek üzerinde bulunan bir yıldızın doğduğu ve battığı noktalar o yerin doğu ve batı noktalarını verir. Güneşte 21 Mart ve 23 Eylül tarihlerinde eşlekte bulunur. Onun için çoğu yerde, "doğu ve batı noktaları güneşi bu tarihlerde doğduğu ve battığı noktalar " olarak tanımlanır.

Uçlaklardan ve zenitten geçen yarı-çembere "öğlen çemberi" ya da "öğlen" (Meridyen) denir. Güneş bu çembere gelince gün ortası yani öğle olmuştur. Onun için ona bu ad verilmiştir. Öğlen düzlemi zenit ve nadiren geçtiğine göre düşey doğrultuda bulunur. Onun çevren düzlemi ile arakesitine "öğlen çizgisi" denir. Bu çizgi çevren çemberini iki noktada keser. Onlardan biri "güney" öbürüde "kuzey" noktası olur. Kuzey noktası uçlak tarafına düşer. Basit bir kural olarak şöyle söylenir: Ayakta duran bir kimsenin sol elini doğu, sağ elini batı yönüne uzatsa güney önüne ve kuzey de arkasına gelir. Bu tanım güney yarım kürede bulunan bir gözlemci için de geçerli olacaktır. Göğün dönme ekseni ve eşleği değişmeden durur. Güney enlemlerdeki bir M noktasının yalnız çevren düzlemi yer değiştirmiştir. Göğün kuzey uçlağı altta, güney uçlağı da güney doğrultusunda ve üstte bulunur. Demek ki bizim gök uçlağımız kuzey doğrultuyu gösterdiği halde güney ülkelerinde güney uçlak onların güney doğrultusunu gösterir. Bizde gün yayı güneye yıkık,

onlarda kuzeye yıkıktır. Yön bakımından bir değişiklik yoktur, yukardaki kural aynen geçerlidir.

Burada şunu yeniden belirtmek gerekir ki gök uçlakları ve eşlek çemberi gök küresi üzerinde çakılı durmaktadır. Zenit, çevren ve öğlen çemberi, doğu-batı, kuzey-güney noktaları gözlem yerine bağ olarak gök küresi üzerinde başka başka yerlerde bulunurlar.

Günlük hareketin sonucu olarak yıldızların, güneş ve ayın doğup battıklarını söyledik. Şimdi bu olayları daha yakından inceleyelim. Önce günlük hareket tanımını daha yakından açıklığa kavuşturalım. Merkezinde bulunduğumuz bir gökküresi var. Milyonlarca yıldızlar bu küre üzerinde serpilmiştir. Öyle ki bir yıldızın bu küre üzerindeki yeri yüzyıllar boyunca durağandır. Onun için onları gökküresi üzerinde çakılı sayabiliriz. Güneş, ay ve gezegenler belirli çizgiler üzerinde gezinirler. Fakat onlar da yine bu küre üzerine bulunurlar. Bütün bu cisimleri üzerinde taşıyan bu koca küre bir eksen çevresinde, eksi yönde (saat yönü) dönmektedir. Dönme süresi bir gündür. Her gün gördüğümüz bu toplu dönmeye "günlük hareket" denir. Bir gözlemci bu harekete katılan yıldızların kimini görür, kimini görmez. Kimini birkaç saat görür, kimini her zaman her zaman görür. Bu durumlar yıldızın gök küresi üzerindeki yerine ve gözlemcinin bulunduğu enleme bağlıdır.

Önce yeryüzünün bir φ enleminde duran bir gözlemciyi göz önüne alalım ve orada yıldızların doğma ve batmalarını inceleyelim. İlk sorun φ enlemi belli iken çevren çemberini gök küresi üzerinde nasıl yerleştireceğiz? Bu da güç değildir. Yer küresi üzerinde, enlemi φ olan bir nokta göz önüne alalım. Enlem, yer eşleğinden olan açısal uzaklıktır. Yer yuvarlağını şimdilik bir küre olarak düşünelim. Seçtiğimiz noktadan çizilen teğet düzlem ile AP doğrultusu arasındaki açı φ enlemine eşittir.

Sonuç olarak, gök uçlağının çevrenden açısal uzaklığı gözlem yerinin enlemine eşittir. Çevren çemberini gök küresine yerleştirirken bu özellikten yararlanacağız. Şöyle ki gök küresinde uçlaklar doğrusu yani dönme ekseni belirtildikten sonra, merkezden ona dik olarak çizilecek çember "eşlek" olacak. Kuzey uçlaktan φ açısal uzaklığında çizilecek çember de o yerin "çevreni" olacaktır. Güney yarım kürede duran bir gözlemci için elbette güney uçlak kullanılacaktır. Bu çizimlerde yer küresi, gök küresi merkezinde bir nokta olarak temsil edilir. Günlük harekette yıldızlar genellikle doğarlar, çevren üstünde bir süre yükselirler. Öğlen çemberine gelince en yüksek noktaya erişmiş olurlar. Bu anda yıldız "üst

geçişe" (üst külminasyon) gelmiştir. Sonra gittikçe alçalırlar ve sonra da batarlar. Çevrenin altında aynı hareketi böylece sürdürür, yine öğlen çemberinde alçak noktaya erişmiş olurlar. Bu anda da yıldız "alt geçişe" (alt küminasyon) geçmiştir. Sonra yükselmeye başlar ve yeniden doğarlar. Her bir yıldız böylece kendine özgü bir çember üzerinde dönüp dolaşır. Ona yıldızın "günlük hareket çemberi" diyebiliriz. Bu çember yıldızın gök küresi üzerindeki yerine bağlıdır. Eşlekte bulunan bir yıldız en büyük çemberi çizer. Uçlaklara yaklaştıkça çemberlerde küçülür. Genellikle bu çemberlerin bir parçası çevren üstünde, öbür parçası da altında kalır. Üstteki parçaya "gün yayı" alttakine de "gece yayı" denir. Bu deyim güneşe göre söylenmiştir. Güneş gün yayını çizerken ortalık aydınlık, gece yayını çizerken de ortalık karanlık olur. Yıldızlarda durum tersine olur. Onlar gün yayını gece, gece yayını da gündüz çizerler. Gün yayı görülme süresine karşılıktır. Onun için gün yayı deyimi yerine, karışıklığı önlemek için çoğu zaman "görülme süresi" deyimini kullanacağız. Bilindiği üzere;

180° lik gün yayı= 12 saatlik görülme süresidir.

Şimdi yıldızların gök küresi üzerinde bulundukları yerlere göre gün ve gece yaylarını, dolayısıyla görülme sürelerini irdeliyelim. Önce bir tanım verelim: Bir yıldızın, eşleğe dik olan saat çemberi boyunca, eşlekten olan açısal uzaklığına "dikaçıklık" denir. Bu da genellikle δ harfi ile gösterilir. Kuzeye doğru 0° ile -90° arasında ölçülür. Yer küresi üzerinde enlem ne ise gök küresi üzerinde de dikaçıklık odur.

Tam eşlekte bulunan bir yılz için (δ=0°) gün ve gece yayları eşit, görülme süresi 12 saattir. Çünkü eşlek çemberi ile çevren çemberi ikiside merkezden geçmektedir ve birbirlerini yarıya bölmektedirler. Buna göre, eşlekte bulunan bir gök cismi tam doğu noktasından doğar ve batı noktasından batar.

Bizim enlemlerde bulunan bir yerde, eşleğin kuzeyinde bulunan bir yıldız (δ>0) için gün yayı >180° ve dolayısıyla görülme süresi > 12 saat. Böyle bir yıldızın doğma ve batma noktaları çevrende kuzey yana doğru kaymıştır.

Kuzey uçlaktan açısal uzaklığı gözlem yerinin φ enlemine eşit olan bir Y1 yıldızı ele alalım. Onun dikaçıklığı δ= +90° -φ dır. Böyle bir yıldızın günlük çemberi tamamen çevren

çemberin üstünde kalır. K kuzey noktasını yalayarak geçer. Hiç batmaz, yani görülme süresi 24 saattir.

Dikaçıklığı $\delta > (-90° - \varphi)$ olan yıldızlar için günlük çemberlerin tümü çevren üstünde kalır. Dikaçıklık büyüdükçe bu çemberler küçülür ve P kuzey uçlağına doğru yaklaşır. Onların görülme süreleri 24 saatten büyüktür. Bundan dolayı bu durumda bulunan yıldızlara "batmayan yıldızlar" denir.

Göğün gney yakasına serpilmiş yıldızlar için ($\delta < 0$), durum tamamen tersine olur. Gün yayı $<180°$, görülme süresi < 24 saattir, doğma ve batma noktaları da çevrenin güney noktasına doğru kaymış olur.

Göğün P' güney uçlağından açısal uzaklığı $-\varphi$ olan Y2 yıldızı ($\delta = -90° + \varphi$) göz önüne alınsın. Onun günlük çemberinin tümü çevren altında kalır ve çevren düzlemini G noktasında yalayarak iner. Bu yıldızı A düzlemcisi hiç göremez. Bunun gibi, daha güneydeki yıldızlarda ($\delta < -90° + \varphi$) hiç görülmezler. Bundan böyle bu yıldızlar φ enlemi için "doğmayan yıldızlardır".

2.1.1 Günlük hareketin enlemlere göre değişimi

Buraya dek yaptığımız incelemede A gözlem noktasını, dolayısıyla φ enlemini durağan olarak almış, ona göre gün ve gece yaylarını irdelemiştik. Acaba yeryüzünde eşleklerden uçlaklara doğru bir gezi yapsak durum nasıl olur? Buda ikinci bir irdelemedir. Burada göğün dönme ekseni ve yıldızların günlük çemberleri olduğu gibi duracak, yalnız çevren düzleminin eksene göre eğikliği değişecektir. Buna göre durum şöyle olacaktır.

1. Yer eşleğinde bulunan bir gözlemci için ($\varphi = 0°$) çevren düzlemi PP' ekseni ile çakışık olur. Bundan böyle bütün yıldızların günlük çemberleri çevren düzlemince yarıya bölünür. Bunun sonucu, bütün yıldızların gün ve gece yayları eşit, görülme süreleri 12 saat olur. Orası için doğmayan ve batmayan yıldız yoktur.

2. Yer eşleğinden kuzey uçlağa doğru gittikçe φ açısı artı yönde büyüyecek. Kuzeydeki yıldızların ($\delta < 0°$) görülme süreleri 12 saatten büyük, güneydeki yıldızların

($\delta<0°$) görülme süreleri 12 saatten küçük olacaktır. Doğmayan ve batmayan yıldızlar vardır. Onların sayısı φ ile orantılı olarak artacaktır. Enlemin büyümesi gün ve gece yaylarını da büyütecektir.

3.Yer küresini tam uçlak noktasına ($\varphi = +90°$) gelindiğinde çevren düzlemi gök eşleği ile çakışmış olacaktır. Gök küresinin kuzey yakasında bulunan bütün yıldızlar çevrene paralel çemberler çizerek günlük hareketlerini sürdürürler. Onların hepside batmayan yıldızlardır. Güney yakadaki yıldızlara gelince onlar hiçbir zaman görülmezler. Onların günlük çemberler. Çevren altındadır. Bu yıldızların hepside doğmayan yıldızlardır.

4.Yer eşleğinin güneyine doğru gidilseydi aynı özellikler karşı yönde olurdu. Burada φ enlemi eksi yönde büyüdükçe göğün P' güney uçlağı tepeye doğru dikilir, güney yakanın yıldızları batmayan yıldızlar olarak gittikçe çoğalırdı. Tam güney uçlağa ($\varphi' = -90°$) gelindiğinde, P' noktası tepemizde bulunur, göğün güney yakasındaki yıldızların hepsi de batmayan yıldızlar olarak görülür.

2.1.2 Günlük hareketin merdiyenlere göre değişimi:

Merdiyenler güneşin hareketini etkilemezler ancak her merdiyen arasında 4 dk saat farkı olduğuna göre güneşin doğuş ve batış saatleri değişir.güneşin hareketi birinci derecede enleme bağlıdır.

2.2 Güneş'in Görünürdeki Hareketi:

Güneş de günlük harekete katılır. O da her gün doğar, gün yayını çizer ve sonra batar. Bu hareket ilk görünüşte her hangi bir yıldızınkinden farksızdır. Fakat yıl boyunca sürekli yapılacak bir gözlem sonucunda iki önemli fark ortaya çıkar. Bunlardan ilki onun her gün yaklaşık olarak 4 dakika gerilemesi, ikincisi de gök eşleğinin iki yanında, belirli iki çizgi arasında, yıl boyunca gezinti yapmasıdır. Bu ikinci olayı gün ve gece yaylarının yıl boyunca değişmesi olarak da söyleyebiliriz.

Eğer üzerinde yaşadığımız yer yuvarlağı güneş çevresinde dolanmasaydı ve güneş de yıldızlar gibi sonsuz uzaklıkta olsaydı böyle bir değişiklik olmazdı. Değişikliğin ne olduğunu şöylece gözleyebiliriz. Kolaylık olması için güneşin eşlekte bulunduğu tarihte (21 Mart ve 23 Eylül) işe başlayalım. Yine eşlekte bulunan ve sabaha karşı bir yıldız seçelim. Yıldızların ve güneşin doğma saatlerini saptayalım. Diyelim ki güneş yıldızdan 3 saat 16 dk. Sonra doğmuştur. Birkaç gün aynı iş yapılırsa görülecektir ki ikinci gün güneş yaklaşık 3sa.20dk. , üçüncü gün 3sa.24dk. sonra doğacaktır. O halde güneş gök küresi üzerinde, günlük hareketin ters yönünde günde yaklaşık 4dk., açı olarak günde yaklaşık 1° geri kalmaktadır. Bu gerilemenin daha yakın değeri 3dk.56sn dir.

Gün ve gece yaylarının yıl boyunca sürekli olarak değiştiği herkesçe bilinir. Türkiye gibi kuzey enlemlerde bulunan ülkelerde, yazın günler uzun ve geceler kısadır. Kışın ise geceler uzun ve günler kısadır. Demek ki bizim yaz mevsimlerimizde güneş gök eşleğinin kuzeyinde ve kışın da güneyinde bulunur. 21 Mart 23 Eylül de gece gündüz eşitliği olduğuna göre, bu tarihlerde güneş tam eşlek üzerine gelmiş olur. 21 Marttan sonra güneş her gün birazcık, yaklaşık 15'.6 kuzeye doğru kayar, böylece günlerde uzamaya başlar. Bu yöndeki kayma 22 Hazirana değin sürer. Bu tarihte güneş eşlekten kuzeye en büyük açısal uzaklığa erişmiştir. Bu açının değeri $\delta\Theta = +23° 27'$ dır. Bugün günlerin en uzun olduğu tarihtir. Bundan sonra güneş bir süre durakladıktan sonra güneye yönelir, her gün birazcık eşleğe doğru yaklaşır. Bunun sonucu günler kısalmaya başlar. Burada güneşin kuzeye erişebildiği 23° 27' uzaklıktaki paralel çembere "yaz dönencesi" denir. Olay gün uzamasından gün kısalmasına geçiştir. Onun için bu geçişin olduğu güne "gün dönümü" denir. Bu deyim olayın kendisi için de kullanılır.

Yaz dönencesinden sonra güneş 23 eylülde yeniden eşleğe gelmiş, $\delta\Theta = 0°$ olmuştur. Gece ile gündüz yine eşittir. Güneye kayma devam etmektedir. 22 aralıkta eşlekten güneye en büyük açısal uzaklığa ($\delta\Theta = -23° 27'$) erişmiştir. Güneş şimdi "kış dönencesi" ne gelmiştir. Bu da ikinci bir "gün dönümüdür". Çünkü bugün kuzey yarım küredekiler için gece en uzundur, bundan sonra da kısalmaya başlayacaktır. Gerçekten bu tarihten sonra güneş kuzeye yönelir, her gün birazcık eşleğe doğru yaklaşır. 21 Martta yeniden eşleğe gelmiş olur. Görülüyor ki güneşin yaptığı bu gezinti sonucu, bir yıl dört parçaya ayrılmış

oluyor. Bunlar sırayla "ilkbahar, yaz, sonbahar ve kış"tır. Bunlardan her biri bir mevsim olur.

2.2.1 Güneşin gün boyunca hareketi

İncelediğimiz iki olayın gerçek yönünü hiç olmazsa kısaca gözden geçirmek gerekir. Yerin eşlek düzlemi yörünge düzlemine göre $\varepsilon = 23° 27'$lık bir açı yapar. Yer de artı yönde belirli bir eksen çevresinde döner. Dolanma boyunca Yerin dönme ekseninin doğrultusu değişmez. Bu özellikleri bildikten sonra görünürdeki özellikleri kolayca anlayabiliriz. Birinci özellik 4dk lık gecikme idi. Gerçekten yer yörünge hareketini 365 günde tamamlar. Buna göre günde 1° artı yönde ilerler. Herhangi bir anda güneş bir A yerindeki gözlemcinin öğlen çemberinde bulunsun. Aynı anda ve aynı doğrultuda bir y yıldızı var olsun. Diyelim ki tam 30 gün sonra durum nedir? yer artı yönde yaklaşık $\alpha=29° 6'$ ilerlemiştir. O halde yıldız A noktasının öğlen çemberine geldiği an güneş henüs 29.6 derece doğuda bulunmaktır. Güneşinde öğlen çemberine gelmesi için yerin daha 29.6 derece artı yönde dönmesi, yani 1saat 58 dakika daha beklemesi gerekir.

2.2.2 Güneşin yıllık hareketi

İkinci özellik eşleğe göre yıllık gezinti idi.21 mart ta güneş tam eşlek üzerine rastlar. Dönme eksen doğrultusu sabit kaldığına göre yerin kuzey yakası gittikçe güneşe yatmış olur. Böylece güneş ışınları 21 martta eşleğe dik geldiği halde 22 haziranda 23° 27' lik enlemlere dik olarak düşer. Demek ki 21 marttan sonra kuzey enlemlerde gün yayları uzuyor, ışınlar gittikçe dikleşiyor.

Güneşin gök küresi üzerinde gezintisi incelenirken, hergün güneşin öğlen çemberine geldiği anda onun gök küresi üzerindeki yerinin saptanması en kolay iştir. Buna göre:

1 güneşin yıl boyunca gezindiği çizgi bir çemberdir. Ona tutulum çemberi denir.

2. güneş bu çember üzerinde ve doğu yönünde (artı yön) günde ortalama 59'.14 kayma yapar. Bunun sonucu hergün ortalama 3 dk. 56sn bir gecikme yapar.

3. Bu kaymanın dönemi, yani aynı noktaya gelme süresi 365.2564 gün dür.

4. Tutulum çemberi eşlek çemberini iki noktada keser. (21 mart ve 23 eylül)

5. Güneş yörünge hareketinin ortalama açısal hızı tutulum çemberi üzerinde bir ayda yaklaşık 30derecedir.

Gün ve gece yaylarının enlemlere göre değişimi:

Eşlekte Q=0 her mevsimde gece ve gündüz süreleri eşittir. Enlem büyüdükçe gün ve gece süreleri farkı artar. Q=90 yani kutuplarda yazın altı ay gündüz, kışın 6ay gece olur. 23 ve – 23 derece arasındaki bölgede gün ve gece süreleri farkı çok azdır. Güneş ışınları dik ve dike yakın eğikliklerde düşer.

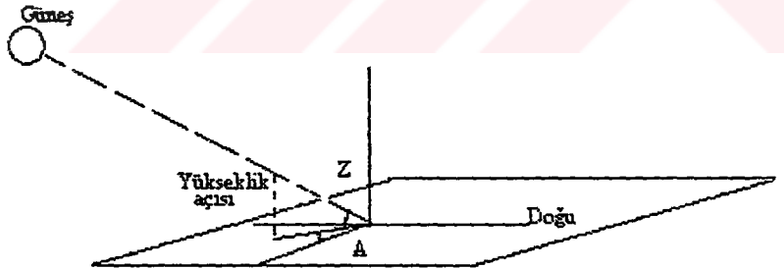
2.2.3 Güneşin zenit ve azimut açılarının değişimi

Zenit açısı (Z) =90-yükseklik açısı, Azimut açısı (A)=doğu açısı – 90,

$$Z = ArcCos(Sin\varphi.Sin\delta + Cos\varphi.CosH.Cos\delta$$ (Smart,1957).

$$A = ArcCos\frac{Sin\delta - Sin\varphi.CosZ}{Cos\varphi.SinZ}$$ (Smart,1957).

H:Saat açısıdır.24 saat 360 dereceye eşittir.



Şekil 2.1 Azimut ve zenit açıları

3 GÜNEŞ İZLEME SİSTEMİ

3.1 Sistemin kurulması

Sistem çalıştırılmadan önce bazı referanslar belirlenmelidir. Bu referanslar panelin hangi referanslara göre tespit edilmesidir. Güneşin konumunu hesaplayan denklemlerin referansları güneşin doğuş noktası ve düz zeminin yüzeyidir. Doğu yönü panelin referans noktasıdır. Birinci referans panelin yüzey normali düz zeminle sıfır derecelik açı yapmasıdır. İkinci referans panelin yüzey normali doğuyla sıfır derecelik açı yapmasıdır ki bu işlem bir pusulayla yapılır. Sistem yukarda anlatılan şekilde monte edildikten sonra bilgisayar programında panelin konumu belirtilir.

İkinci olarak sistemin kurulacağı yerin enlem ve boylamının haritadan bulunması ve programa girilmesidir.

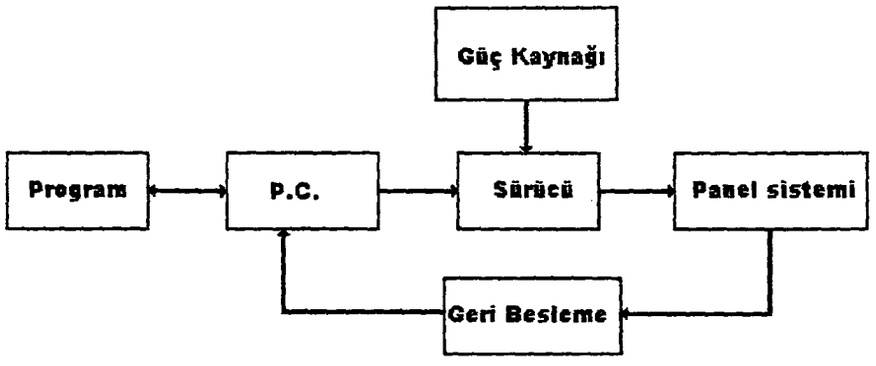
3.2 Sistemin çalışması

Güneş ışınlarının doğuyla yaptığı açı ve su yüzeyiyle yaptığı açı programda bulunan denklemler ile iki dakikada bir hesaplanır. Hesaplamada bilgisayar sistemindeki tarih ve saat alınır. Hesap sonucu panel bu konumda değilse bu konuma, doğru akım motoru ile hareket ettirilir. Panelin ne kadar hareket ettirildiği shaft encoderlerle belirlenip, bilgisayara girilir. Shaft encoderlerden alınan palsler sayılıp, açıya çevrilir. Panelin konum açıları hesap edilen açılara eşit yapılıncaya kadar panelin hareketi devam eder. Panel güneşe yönlendirildikten sonra panelin son konumu bilgisayara kaydedilir. İstenilen dakika sonra güneşin konum açıları yeniden hesaplanır. Panelin son konumu kayıtlı olduğundan, panelin kaç derece daha hareket ettirilmesi gerekiyorsa shaft encoderlerden o açıya eşit pals sayısı alıncaya kadar panel hareket ettirilir. Panelin bulunduğu pozisyon tekrar kaydedilir. Bu işlem güneşin doğuş saati ile batış saatleri arasında devam eder. Güneşin doğuş saati ile batış saatleri tarihe göre bilgisayar tarafından hesaplanır.

Program herhangi bir sebeple durdurulsa dahi panelin bulunduğu konum hafizada kayıtlı olduğundan, tekrar sistem çalıştırıldığında panel güneşe yönlendirilmesi hatasız yapılabilir.

3.3 Sistemin Blok Diyagramı

Yapılan sistemin blok diyagramı aşağıda verilmiştir.



Şekil 3.1 Sistemin blok diyagramı

3.3.1 Güç kaynağı

Yapılan güneş izleme sistemi şebeke bağlantılı fotovoltaik sistem ile veya bağımsız fotovoltaik sistem ile kullanılabilir. Sistem için gerekli beslemeler şunlardır:

Bilgisayar beslemesi: 220V, AC

Sürücü devre beslemesi: 12V, DC

Motorların beslemesi: 12V, DC

Shaft encoderlerin beslenmesi: 5V, DC

Bilgisayar beslemesi şebeke bağlantılı fotovoltaik sistemlerde direk şebekeden sağlanır. Fotovoltaik sistemlerden alınan enerji, inverter ile şebeke frekansına eşit AC'a dönüştürülür. Bağımsız güneş pillerinden üretilen doğru akım, ev aletlerinin çalışabilmesi için gerekli 220V, 50 Hz alternatif akıma, fotovoltaik sistemlerde var olan inverter ile dönüştürülür. Bilgisayar ınverter çıkışından beslenir.

Sürücü devre beslemesi ve paneli hareket ettiren dc motorların beslemesi fotopilde üretilen enerjinin depolandığı 12V akümülatörlerden yapılır.

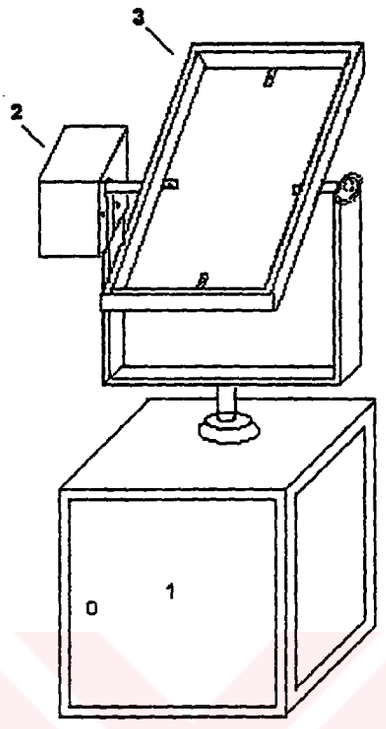
Geri besleme elemanı olarak kullanılan shaft encoderin ve kullanılan TTL entegrelerin beslenmesi için 5 Voltluk kaynağa ihtiyaç vardır. 5V, sürücü devresine konulan7805 regülatör entegresi ile elde edilir.

3.3.2 Panel sistemi

Güneş izleme sisteminde iki yörüngesel hareket vardır. Birinci hareket sistemin doğu-batı hareketi, diğeri güney-kuzey hareketidir. Doğu-batı hareketi güneşin doğuşundan batışına kadar devam eder. Güney kuzey hareketinin sınırları mevsimlere göre değişip, gün boyunca devam eder.

Şekil(3.2)'de yapılan güneş izleme sistemi görülmektedir. Sistemde hareketleri sağlayan iki adet 12V'luk doğru akım motoru kullanılmıştır. Kullanılan doğru akım motorlarının özelliği düşük güçlü, redüktörlü olması ve frenleme sistemine ihtiyaç göstermemesidir. Devir sayıları 20 dev/dk'dır. Stabilize açısından devir sayıları 0.5 dev/dak' ya indirilmiştir. Böylece az salınlı bir sistem yapılmış olup, aynı zamanda sistem çalıştırıldığında maksimum 1 dakika sürede güneşin pozisyonunu yakalar.

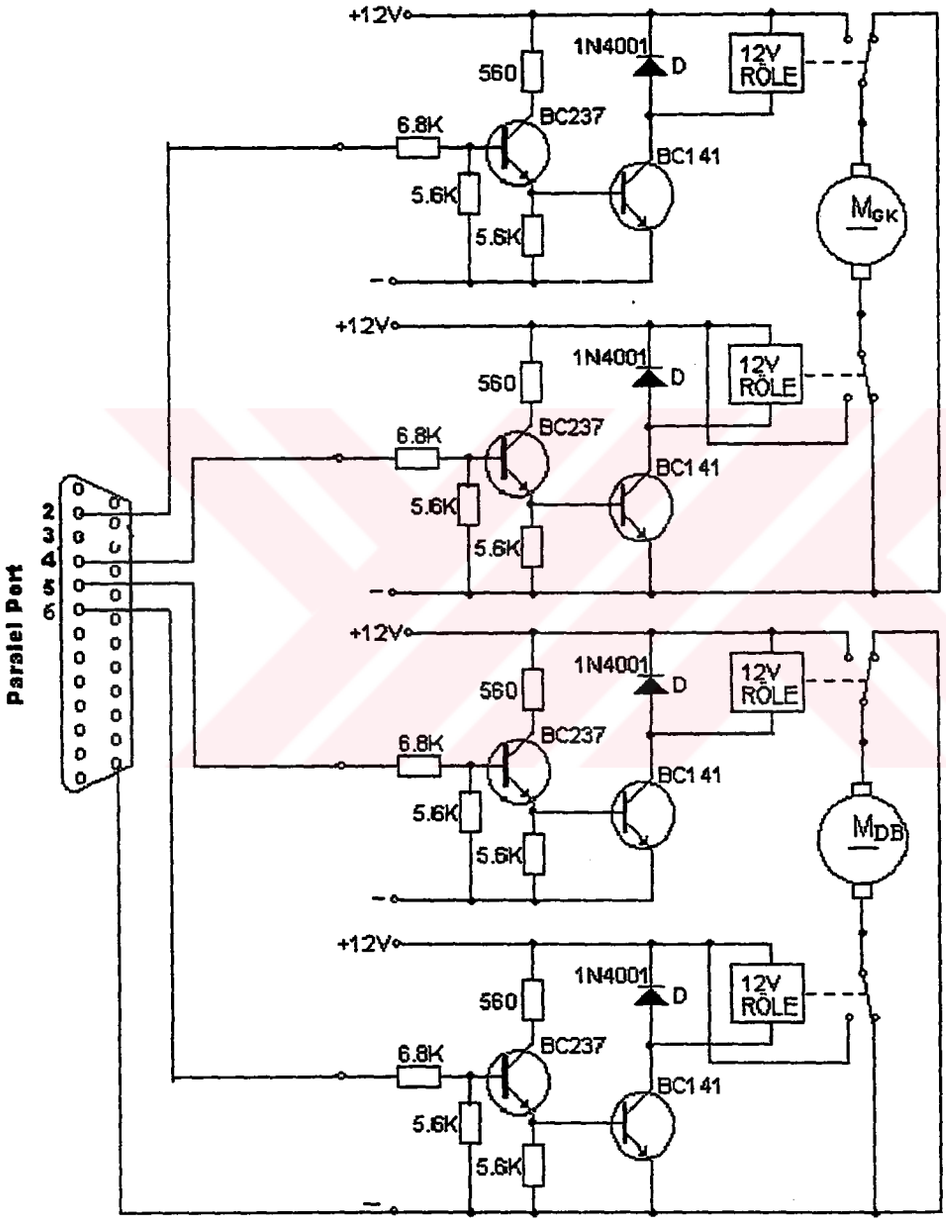
Şekil (3.2)' de 1 numara olarak gösterilen kutu sistemin diğer elemanlarını taşıyan parçadır. İçerisinde sürücü devre, bu devrenin besleme gerilimini sağlayan ve fotopil tarafından şarj edilen akümülatör, doğu-batı hareketini sağlayan doğru akım motoru, sonsuz dişli takımı ve doğu-batı shaft encoderi vardır. Şekil de 2 numara ile gösterilen parça güney-kuzey hareketini yaptıran motoru, güney kuzey saft encoderi ve sonsuz dişli takımını dış etkenlerden korumak amacıyla kullanılmıştır. İnce galvanizli sacdan yapılmıştır. 3 numara ile gösterilen kısım fotopilin monte edildiği paneldir. Özelliği hafif olmasıdır.



Şekil 3.2 Güneş izleme sistemi

3.3.3 Sürücü Devre

Motorları süren devredir. Bilgisayarın paralel portundan lojik bilgiler bir yükselteç devresiyle motoru çalıştıran 12Voltluk röleleri enerjilendirir. Bilgisayar, motorları 4 bit üzerinden sürer. Bu dört bit paneli doğuya, batıya, güneye ve kuzeye hareket ettiren bilgileri taşır.

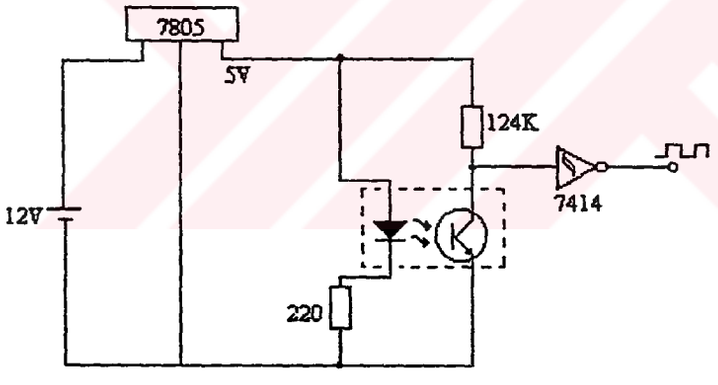


Şekil 3.3 Sürücü devre

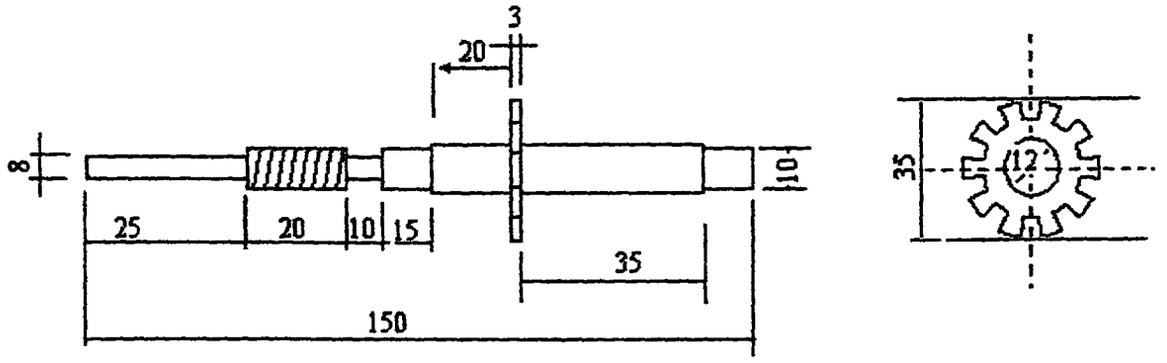
3.3.4 Geri Besleme

Yapılan güneş izleme sisteminde shaft encoder geri besleme elemanı olarak kullanılmıştır. Shaft encoder dairesel hareketlerde hız ve pozisyon sensörü olarak kullanılır. Sistemde pozisyon sensörü olarak kullanılmıştır. Sensörün bir devir dönmesiyle normalde 9 pals çıkış alınır. Ancak palsların yükselen ve düşen kenarları program tarafından değerlendirilerek 18 pals algılanabilmektedir. Sistemde kullanılan dc motorların devir sayıları 20 dev/dk olup sonsuz dişli kullanılarak panelin devir sayısı dakikada 0.5' e indirilmiştir. İmal edilen shaft encoder motorların miline bağlanmıştır. Yani motorun 20 devir dönmesi sonucu, panel180 derece açı taramakta ve shaft encoderden 360 pals alınmaktadır. Böylece yapılan sistem 0.5 derece hassasiyete sahip olmuştur.

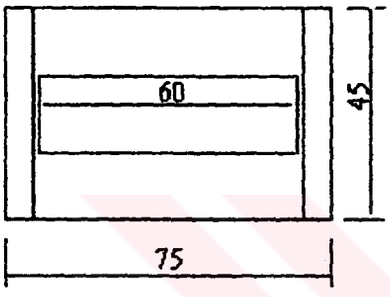
Kullanılan sensör tasarım aşamasında imal edilmiştir. Elektronik devresi aşağıda verilmiştir.



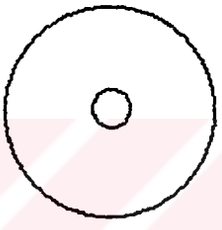
Şekil 3.4 Optik shaft encoder devresi



Dönen kısım



Dış muhafaza



Kapaklar

Şekil 3.5 Optik shaft encoder ölçüleri

3.3.5 PC

Sistemde, alınan bilgilerin değerlendirilmesi ve yazılan SPC (Sun Panel Control) kontrol programının çalıştırılması amacı ile pentium tabanlı bir PC kullanılmıştır.

Bilgisayar ile dış dünya arasındaki bilgi alışverişi bilgisayarın üzerinde bulunan portlarla gerçekleştirilir. Bilgisayarda iki çeşit port bulunur.

Sistem ünitesinin arka tarafında 25 uçlu D tipi konnektöre paralel port denir. Bu port kullanılarak sistemden dışarıya bilgi gönderilebilir (output) yada dışardan gelen bir sinyal (ınput) bilgisayar tarafından işlenebilir.

Bu porta bağlanan cihazlara gönderilen veya alınan sinyallerin seviyesi TTL seviyesindedir yani 5V' tur. Paralel portta veri alış verişi 8 bit üzerinden yapılır. Sistem ünitesine bağlı birden fazla paralel port bulunabilir. Bu paralel portlara LPT1, LPT2,LPT3... gibi isimler verilir. IBM uyumlu bilgisayarlarda bulunan paralel portun uç bağlantı numaraları aşağıda verilmiştir.

Bir paralel porta bağlanan cihazdan veri alışverişi yapılabilmesi için verilerin bir yere kaydedilmesi gerekir. İşte veri kaydının yapıldığı bu geçici belleğe register denilir. Her registerin hexadesimal olarak ifade edilen bir adresi vardır. Bu register adresleri 378_H, $37A_H$, 379_H dir. Port registerlerin kullanımı aşağıda verilmiştir :

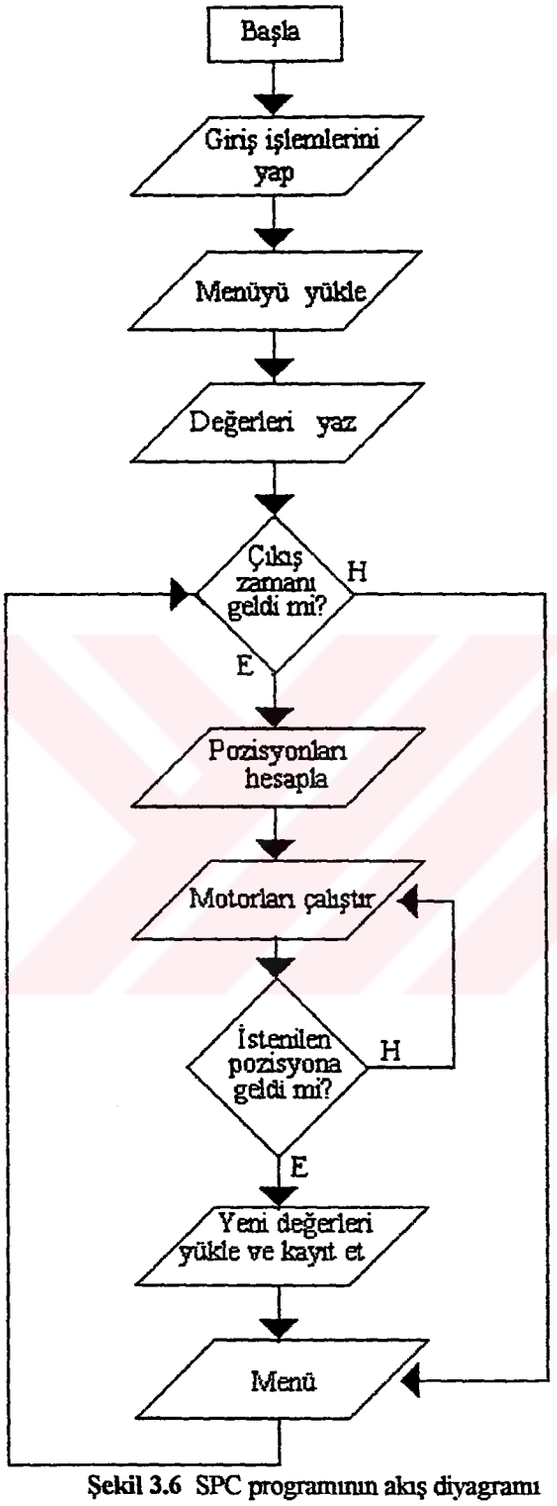
Data latch, 378_H: Bu adrese yazılan data, bir tampon belleğine gönderilir ve burada saklanır. Yeni bir data gönderilene kadar buradaki data sabit kalır.

Controls, $37A_H$: Paralel port kontrol sinyalleri bu port aracılığıyla yapar. Ayrıca bu adresteki datalar mikro işlemci tarafından okunabilir.

Status, 379_H: Paralel porta bağlanan cihazların durumu hakkındaki dataların okunması için datalar bu adreste saklanır.

3.3.6 Program:

Programın akış diyagramı aşağıda verilmiştir.



Şekil 3.6 SPC programının akış diyagramı

SPC programının mantığı, güneşin pozisyonu saate göre belli olduğu veya hesaplandığına göre panelin bu pozisyona getirilmesidir. Bu işlemin ilk etabı güneş pozisyon bilgisinin hesaplanmasıdır. Daha sonra programda kayıtlı olan panelin pozisyonu ile hesaplanan güneşin pozisyonu arasında fark alınıp bu farka göre motorların hareket etmesi sağlanır. Panel hesaplanan konuma gelip gelmediği optik shaft encoder ile alınan pals sayısıyla belirlenir. Bu esnada panelin bulunduğu pozisyon kayıt edilir. Bir kayıt hatası olması durumunda panel bilgileri resetlenir ve panel başlangıç konumuna alınır. İşlem yeniden tekrarlanır. Bu işlem artımsal hataların büyük olması durumunda kullanıcı tarafından da yapılabilir. Bu işlem ana menüden kolaylıkla yapılabilmektedir. Güneşin pozisyon hesabı istenilen peryotlarda yapılabilir. Kullanılan shaft encoderin özelliğine göre açı pals oranı değiştirilmesi menüden mümkündür. Tüm program giriş bilgileri sayısal değerlerle yapılmalıdır. Aksi durumda hata mesajı verilmektedir.

SPC programı turbo pascalda yazılmıştır. Şimdi Turbo pascal programlama dili üzerinde kısaca duralım.

Pascal programlama dili 1968 yılında Niklaus Writh tarafından geliştirilmiştir (Akgöbek, 1995). Bu dilin Turbo Pascal versiyonu, günümüzde mühendislik, ticari ve bilimsel alanlarda yaygın olarak kullanılmaktadır.

Turbo Pascal' da yazılan programlar kısa, anlaşılması kolay ve çok hızlı olarak çalışırlar. En önemli özelliklerinden birisi, programcının kendisine gerekli olan program parçalarını (unit) bir defa hazırladıktan sonra, bu program parçalarını istediği yerde çok kolay bir şekilde kullanabilmektedir. İstenilen şekilde veri tipi tanımlanarak programın daha esnek ve daha kısa sürede tamamlanması sağlanmaktadır. Programın bir başka önemli özelliği ise program yazma işlemleri için hazırlanan editörün esnek olması, programların makine diline anında çevrilmesi, hata bulma ve hataları düzeltme işlemlerinin pratik ve hızlı olarak yapılmasına imkan vermesidir.

Turbo pascalda yazılan programlar bloklar halinde yazılıp, düzenli ve kolayca okunabilir bir yapıya sahiptir.

Program Yapısı

Turbo pascalda bir programın yapısı aşağıdaki gibidir:

```
Program [Program adı];
Uses [Kullanılacak unitler];
Type [Özel veri tipleri];
Cons [Sabitler];
Var [Değişkenler];
Label [Etiketler];
Procedüre [Precedüre adı];
Type ...
Cost ...
Label ...
Var ...
Begin
 :Procedüre
 :alt program kesimi
End;

Function [Fonksiyon adı];
Type ...
Cost ...
Label ...
Var ...

Begin
 :Function
 :alt program kesimi
End;
Begin
:Ana program
:End.
```

Veri Tipleri:

TURBO PASCAL'DA program içerisinde kullanılan bütün değişkenler tipleri ile beraber tanımlanmalıdırlar. Değişken, sabit ve tip tanımlamaları için VAR, CONST ve TYPE tanımlama blokları kullanılır. Kullanılan tipler:

1.Tamsayı Tipler: Tam sayı değişken ve sabitleri tanımlamak için kullanılan tiplerdir. Bu değişken ve sabitlere ondalıklı değer aktarılmaz. Ondalıklı değer aktarılması veya tamsayı değişkene atama sırasında işlem içerisinde / işaretinin olması veya real bir değişkenin olması durumunda 'type mismatch' hata mesajı verir. Tamsayı tipler aşağıda verilmiştir.

Tip Adı	Sınırları	Kapladığı alan
ShortInt	-128...127	1 Byte
Byte	0...255	1Byte
Integer	-32768 ...32767	2 Byte
Word	0 ... 65535	2 Byte
LongInt	-2147483648 ...2147483647	4Byte

Tablo 3.1. Tamsayı tipler

2.Gerçek (real) Sayı Tipler: Ondalık değişken ve sabitleri tanımlamak için kullanılır. Gerçek sayı tipleri aşağıda verilmiştir.

Tip Adı	Sınırları	Kapladığı alan
Real	$2.9*10^{-39}$... $1.7*10^{38}$	6 Byte
Single	$1.5*10^{-45}$... $3.4*10^{38}$	4 Byte
Double	$5.0*10^{-324}$... $1.7*10^{308}$	8 Byte
Extended	$3.4*10^{-4932}$... $1.1*10^{4932}$	10 Byte
Comp	$-2^{63}+1$ ... $2^{63}-1$	8 Byte

Tablo 3.2. Real sayı tipleri

3.String Tip: Alfasayısal değişken ve sabitleri tanımlamak için kullanılır. Program içerisinde sayısal anlamı olmayan değişkenlerin tanımlamasında kullanılır. String tipin sınırları 1 ile 255 karekterdir.

➢ Turbo Pascal' da Procedure'ler ve Functionlar :

Pascal' da procedure ve Function olmak üzere iki çeşit alt program çeşidi vardır. Herhangi bir alt program çağrıldığı zaman bu alt program devreye girer. Komutların işlenmesiyle alt programdan çıktıktan sonra işlem altprogramı çağıran komuttan bir sonraki komuta geçer. İçiçe birden fazla altprogram kullanılabilir.

Altprogramların bir diğer kullanım amacı ise programın yapısal olmasını sağlamak ve biribiriyle ilgili komutları veya programın bir bölümünü istenilen bir isim altında toplamaktır. Bu şekilde programın okunması kolaylaşmakta ve program yapısal bir görünüm kazanmaktadır.

Her alt program ; isim (ya da başlık), tanımlama ve icra bölümlerinden oluşur. Tanımlama bölümlerinde gerekli tip, sabit, değişken ve etiket tanımlamaları yapılabilir. İcra bölümü ise begin - end bloğu arasına yazılan komutların oluşturduğu bir bölümdür.
Procedure Altprogramları :Procedure altprogramları bir komut gibi çalışır. Genel yapısı aşağıdaki gibidir.

```
PROCEDURE  Procedure Adı [ ( Parametre Listesi ) ] ; [FORWARD]
    Tanımlamalar ;
BEGİN
  İcra görecek komutlar topluluğu ;
END ;
```

Genel yazılımda da görüldüğü gibi procedure kelimesinden sonra procedure adının kullanılması yeterlidir. İsteğe bağlı olarak parametreler tanımlanabilir. Genel olarak procedure' ler çağrıldıkları yerin üst tarafında olmak zorundadırlar. Ancak forward kelimesi yardımıyla procedure çağrıldığı yerin alt tarafında yazılabilir. Ayrıca procedure adı tanıtıcı isim özelliklerine uygun olmalıdır. Yani başka bir değişken ismiyle aynı olmamalıdır. Procedure içerisinde yapılan bütün tamınlamalar sadece bu procedure için geçerlidir. Bir procedure içerisinde başka procedure' ler hazırlanabilir. Bu procedure' lere

içinde bulundukları alt program içinde erişim sağlanabilir. Yani bir procedure içerisinde kullanılan başka bir altprogramı başka bir altprogramdan veye ana program kesiminden erişim sağlanamaz.

Procedure içerisine değer göndermek ve / veya procedure içerisinde hesaplanan bir değeri dışarı göndermek için parametre kullanılır. Parametre tanımlamasını aşağıdaki gibi gösterebiliriz.

PROCEDURE Procedure adı ([VAR] Değ1 [,Değ2....] : Tip [; ....]

Bu tanımlamada köşeli parantez içerisindeki ifadeler zorunlu değildir. Değişkenlerin değer alma ve değer gönderme işlemleri iki ayrı şekilde gerçekleşir. Sadece değer alan değişkenler tek taraflı , hem değer alan hem de değer gönderen değişkenler ise çift taraflı olarak çalışan değişkenlerdir. Değişken isminden önce VAR tanımlama komutun kullanılması durumunda belirtilen değişkenler çift taraflı olarak çalışırlar. Yani procedure içersine hem değer alır hem de procedure den dışarıya değer gönderir. Bir VAR komutunu etkisi kendisinden sonra gelen ilk (;) işaretine kadardır. Çift taraflı olarak kullanılan değişkenlere karşılık mutlaka değişken kullanılmalıdır. Çünkü procedure içerisinde değişkenin alacağı son değer dışarı gönderilecek bilgi ise ancak bir değişkene aktarılabilir.

Function Altprogramları :Pascalda kullanılan komutlara benzer komutlar yapmak için procedure ve yardımcı komut dediğimiz fonksiyonlar yapmak için ise function altprogramlarından yararlanılır. Function'lar tek başlarına kullanılamazlar. Bir yardımcı komut gibi başka komutlar yardımyla kullanılabilirler. Buna Basic'te kullanılan INT , MID$,VAL ... ile Pascal'da kullanılan TRUNC , ROUND , LENGTH ,... fonksiyaonları örnek olarak verilebilir. Programcı program içerisinde istediği gibi fonksiyon hazırlayabilir. Function'ların genel yazılımı aşağıdaki gibidir:

FUNCTION Function Adı [(Parametre Listesi)] :Tip;

Dikkat edilirse her function için 'Function Adı' ile belirtilen bir isim ve isteğe bağlı olarak , procedure 'lerde olduğu gibi parametre listesi verilebilir. Farklı olarak her function' nun bir tipi olmak zorundadır. Sonuç Function Adı' na aktarılarak function' nun değer alması sağlanabilir.

➢ Unit:

Bir unit' in yapısı bir forma bağlı olmasından bağımsız olarak temelde sabittir. Unit' in bölümlerini, şu şekilde tanımlayabiliriz :

```
Unit < tanımlayıcı >
inreface
uses < unit listesi >
{ public declarations }
implementation
uses < unit listesi >
{ private declarations }
{ uygulama prosedür ve fonksiyonları }
initialization
{ seçimlik ilk kullanıma hazırlık kodu }
end ;
```

Unit başlığı, ayrılmış bir kelime olan unit' le başlar, daha sonra unit' in tanımlayıcısı (adı) gelir. Daha sonraki öğe, unit' in arabirim bölümünü tanımlayan interface' dir.

Bu bölüm, bu unit'i kullanan diğer uygulama veye unit'ler tarafından görülebilir. Bir unit , başka unit' lerdeki deklarasyonları , bu unit'leri bir uses kalıbı içinde belirtmek sureti ile kullanılabilir.

➢ Dosyalama:

Aynı yapıdaki kayıtların oluşturduğu bütündür. Sürekli olarak saklanması gereken bilgiler ya program içerisinde sabit olarak tanımlanırlar yada dosyalarda saklanırlar. Program içerisine sabit olarak bilgiler yazılabilir.

Turbo Paskal programlama dilinde üç farklı dosyalama şekli mevcuttur.

1. Text dosyalar
2. Tipli (typed) dosyalar
3. Tipsiz (untyped)dosyalar

Dosya Oluşturmak: Kayıt Alanının Tanımlanması: Record tip dosyanın her bir kağıdının ve bilgi alanlarının eşit uzunlukta olması için bu tanımlama yapılır.

Yazılım: TYPE

Record tip ismi=RECORD

Alt alan deg: tip;

Bu dosyalarda saklanacak bilginin mutlaka bir tipi olmalıdır. Dosyada yer alacak bilgi herhangi bir veri tipinde olacağı gibi Record veri tipi ile birden fazla alan tek bir isim altında toplanarak dosyada yer alabilir.

Dosya tanımlama: Kayıt yapısı bir tipe sahip olan dosyayı tanımlamak için VAR tanımlama bölümünde aşağıdaki ifade kullanılır.

Dosya değişkeni: FILE OF tip adı;

Type Dosyalarda Kullanılan Komutlar:

Assign:Dosya değişkenine dosya ismini atamak için kullanılır.

Rewrite:Dosyayı ilk kez oluşturmak ve dosyayı açmak için kullanılır. Dosyada kayıt bulunması durumunda dosyadaki bütün kayıtlar silinir. Dosyayı boş olarak oluşturur ve dosyaya bilgi yazılmasını sağlar.

Reset: Dosyaya bilgi kaydetmek veya dosyadan bilgi okumak için dosyayı açar. Dosyanın önceden oluşturulmuş olması gerekir.

Ioresult:Giriş çıkış işlemleri sırasında oluşabilecek hataları kontrol etmek için kullanılır. Bu fonksiyon sonuçta Word tipinde tamsayı bir değer üretir. Sonucun sıfır olması hata olmadığını, işlemin yapıldığını belirtir. Bu komut özellikle dosyanın açılması sırasında kullanılır. Mevcut olmayan bir dosya Reset komutu ile açılmaya çalışıldığında hata meydana gelir ve program çalışması kırılır. Programın kırılmasını engellemek için {I$-} directive' inden, hatayı kontrol için ise IORESULT fonksiyonundan yararlanılır.

Seek: Dosyanın istenen kayıt numarasına gitmek için kullanılır. 'Kayıt no' okuma işlemi sırasında dosyanın ilk ile son kayıtlar numaraları arasında olabilir. Yani 0. Kayıt ile File Size (..)-1. Kayıt arasında olabilir. Bu durumda belirtilen yere kayıt yazılabilir. Dosyanın en son boş yerine ait kayıt numarasını File Size(..) fonksiyonu verir. En son kayıtın numarası ise File Size (..)-1 dir.

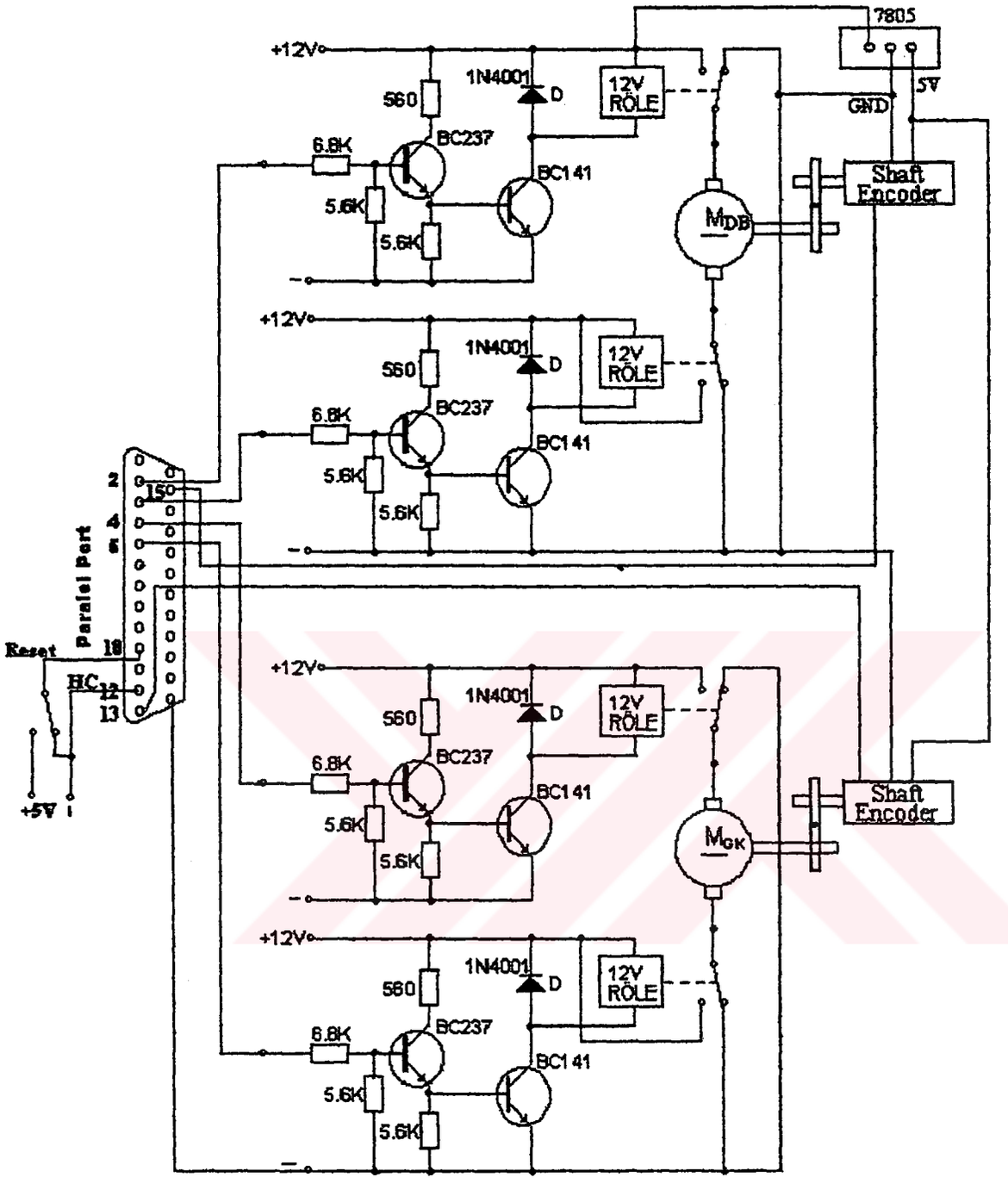
File Size:Dosyadaki mevcut kayıt numarasını verir.

Filepos:Dosyada o anda üzerinde bulunulan kayıtın numarasını verir.

Write: Dosyaya bilginin yazılmasını sağlar.

Read:Dosyadan kayıt okumak için kullanılır.

Close:Açık olan dosyayı kapatmak için kullanılır.



Şekil 3,7 Güneş izleme sisteminin komple şeması

4 SPC (SUN PANEL CONTROL)

Bu program esas itibariyle üç ana kısımdan oluşur. Bunlar :

1.SPC.PAS Dosyası: Bu dosya ana program bloğunu içermekle birlikte, my unit ve SPC unit adlı iki üniteyi kullanır.burada programın ilk çalışmaya başladığı andaki genel gidişatlar ve yönergeler bulunur.

2. My unit Dosyası: Bu dosyada programın bilgi giriş kontrolü ve ekran düzenini sağlayan procedureler bulunmaktadır. Ekranın görünümü ve bilgi girişleri bu procedürlerle yapılır. Ayrıca programdan çıkışı sağlayan procedürü barındırır. Bu unit içerisinde kullanılan ve programa katkıda bulunan procedürler şunlardır:

Cursor On: Bu procedür ekranda yanıp sönen cursorü yok eder.

Cursor Off: Cursor on ile yok edilen cursorü tekrar görünür yapar. Bunun amacı özellikle bilgi girişlerinde veya cursorün nerede olduğunu göstermektedir.

Beep: Bu procedür istenmeyen durumlarda kullanıcıyı sesle uyarır.

M_color: Ekrandaki karekterlerin ve ekranın rengini değiştirir.

Ekran_sakla ( ) : Herhangi bir mesaj görüntüleneceği zaman o anki ekranı ekran adlı değişkene saklar.

Ekran_getir ( ) : Ekran_ sakla ile saklanan ekranı tekrar geri getirir.

Ekran_tipi: Ekran saklamadan önce ekranın tipini belirlemek için kullanılır.Ekran_sakla dan önce kullanılır.

Boya_( ): Ekranın istenilen kordinatlar arasında kalan bölümünü istenilen renge boyar.

Çerçeve ( ) : Ekranın istenilen kordinatlarında, istenilen renkte çerçeve çizer.

End_prog : çağrıldığı zaman bir takım değişkenleri ilk konumlarına alarak programı sona erdirir.

H_menü ( ): Bu procedür ekranın istenen kordinatlarında, istenilen renklerde, istenen eleman sayısında yatay veya dikey hareketli menü oluşturur.

Read_( ) : Ekrandan bilgi istenilen bir maske altında bilgi girişimi sağlar.Eğer bilgi istenen formata uymuyor ise uyarı mesajı verir.

3.SPC Util Dosyası: Bu dosyada SPC programının çalışması için gerekli procedürler bulunur. Dosya pascalın standart üniteleri olan crt ve dos uniteleri ile beraber my_unit ünitesini kullanır. SPC_Util tanımlama bloğunda panelin pozisyon ve zamanla ilgili hesaplamaları tutan değişkenler mevcuttur. Burada panel_data panelin pozisyonu ve panel ile ilgili diğer oranları saklayan bir kayıttır. Dialog ise ekranda çıkan mesajları saklayan değişkendir.

Massage_array: Mesajların içeriğini dizi değişkenidir.

Bu ünit içerisinde kullanılan ve panelin çalışmasını kontrol eden prosedürler şunlardır:

Procedüre load_first_valuce: Bu procedür program çalışmaya başladığında data dosyasıyla kayıtlı olan ilk değerleri değişkenlere yükler.

Load_old_voluces_to_sav: Bu procedür panelin enson kullanıldığı pozisyon bilgilerini saklama değişkenine aktarır.

Find_data_file: Bu procedür data dosyasının var olup olmadığını kontrol eder.Eğer bulamaz ise yeni bir data dosyası oluşturmak için ekranda uyarı mesajı verir.

Open_data_file: Açılmış olan dosyada kayıtlı pozisyon bilgilerini okumak için dosyayı kullanıma hazırlar.

Read_save_pos_data: Açılmış olan dosyada kayıtlı pozisyon bilgilerini okur. Eğer işlem gerçekleşmez ise uyarı mesajı verir.

Sav_data_file: Panelin pozisyon bilgilerini data dosyasına kayıt eder.

Change_ratio: Panelin saate bağlı dönme ve açısal programlarını verir. Burada kullanılan oran Atp (Açı pals) oranıdır.

Vist_values: Panelin o anki ve kayıtlı olan pozisyon bilgilerini verir.

Hardwear_control: Panelin PC' ye takılı olup olmadığını kontrol eder. Bu işlem için portun 12 nolu pini kullanılır.

Get_current_date_time: O andaki sistem tarih ve saatini belirler. Bunları yeni pozisyonun tarih zaman değişkenlerine aktarır.

Calculet_dif_pos: Yeni pozisyon ile eski pozisyonarasındaki farkı hesaplayıp bu değerlere göre panel motorlarının yönü ve dönme miktarını belirler. Dönme miktarlarını optik shaft encoderlerden alınması gereken pals sayısıdır.

New_values_load: Panel yeni pozisyona geldikten sonra, bu bilgileri eski değer olarak atar.

Out_port: Panel motor sürücülerine kumanda sinyallerini gönderir.

İn_port: Optik encoder, reset sinyali ile sistemin PC' ye bağlı olup olmadığına ait sinyalleri alır.

İn_port_control: Alınan sinyalleri ayrıştırarak kontrol eder ve buna göre gerekli değişkenlere atama yapar.

Function Compare_position: Hesaplanan pozisyon ile panelin o anki pozisyonu karşılaştırır. Fark sıfır ise (true)sonucunu üretir.

Run_motors: Yapılan hesaplama sonucuna göre panel yeni pozisyona ulaşıncaya kadar motor sürücülerine gerekli sinyalleri iletir. Karşılaştırma işlemi Compare_position fonksiyonu ile yapılır.

Reset_position: Paneli mekanik olarak reset posizyonuna çekmek için motorları yönlendirir. Panelin resete ulaşıp ulaşmadığını reset_code değişkeni ile control eder.

Linexy: Ekranda tablo çizer.

Write_values: O andaki tüm pozisyon bilgilerini ekrana aktarır.

Load_program: Programın genel gidişatını yönlendirir.

SPC (Sun Panel Control) programı ekte sunulmuştur.

5 SONUÇLAR

Güneş izleme sistemi çok değişik şekillerde yapılabilir. Yapılan sistemin hata oranı ve her koşulda çalışabilmesi önemlidir. Optik sensörlerle yapılan güneş izleme sistemlerinin dış etkenlerden etkilenmesi hata oranını arttırır. Yapılan çalışmada 0.5 derece hassasiyet sağlanabilmiştir. Bilgisayar ile kontrol daha hassas çalışmanın yapıldığı ve her türlü ayarlamaların son derece kolay olması nedeniyle diğer sistemlerden daha üstündür. Yapılan güneş izleme sistemi, hareketinin denklemi belli olan her şeyi izlemeye uygundur.

KAYNAKLAR

Venkatesan, K.,1997 Matching D.C. motors to photovoltaic generators for maximum power tracking, **Univ of Puerto Rico,Mayaguez.**

Wood, N.,1996 Solar tracking system with linear regression error correction, **Coloradostate univ.**

Arboiro,J. C.,Sala G.1997 Self- learning tracking, **Pphoed ISSN.**

Rabbani, M.,Basirat-Tabrizi , H., Taheri,S. 1996, Design of a computercontrolled sun tracking system., **Amirkabir univ of Technology, Tehran.**

Baldemir F., 1991 , Güneş Pilleri, **F.Ü. Fen-Edebiyat Fakültesi, Fizik bölümü**

Alsan S. , 1990, Lojik Devreler ve Darbe Dereleri, **Yıldız üniversitesi**

Bereketlioğlu 'E. , **1994,** Endüstriyel Kontrol El Kitabı, **M.E.B.**

Elo Elektronik Dergisi **1984 Cilt: Sayı:24**

Akgöbek, A.,1995 Turbo Pascal, **Beta.**

Smart, W.M., 1957, Küresel Astronomi, **Cambridge üniv.**

Kızılırmak, A., 1977, Gökbilim Dersleri, **Ege univ.**

Altınbaşak, O., Taşbaşı, A, 1997, Turbo Pascal, Melisa.

Hıll F.J. ve Peterson G.R. , 1980 , Switching Theory and Logical Desing, **Wiley**

Gürdal O. , 1996 , Algılayıcılar Dönüştürücüler, **Gazi Üniversitesi**

EK: SPC (Sun Panel Control) programı

```
Program SPCprogram;

uses Crt,Dos,My_Unit,SPC_Util;

{************************************************************************ }
        begin
m_color(0,11);
messages_load;
set_interface;
in_port;
hardware_control;
port[$378]:=0;
find_data_file;
open_data_file;
read_sav_pos_data;
load_first_values;
load_menu;
        end.

{************************************************************************ }
{======================================================================== }

        unit SPC_Util;
interface
{======================================================================== }
uses crt,dos,my_unit;
```

```
type
panel_data=record
x_pos :integer;
y_pos :integer;
s_time:datetime;
p_time:longint;
x_atp :real;
x_tta :real;
y_atp :real;
y_tta :real;
end;
dialog=record
baslik      :string;
message_line :string;
end;
        var
hour,min,sec,hund,year,month,day,dow:Word;
x:byte;
kx,ky:shortint;
code:integer;
data_file:file of panel_data;
data_file_name:string[12];
message_array:array[1..20] of dialog;
ekran:pointer;
old_pos,new_pos,dif_pos,sav_pos:panel_data;
old_pos_int,new_pos_int,dif_pos_int,sav_pos_int:longint;
direction_x_motor,direction_y_motor,output,input:byte;
wait_ok,reset_code:boolean;
wait_time:longint;
per_time:real;
```

```
x_pulse,y_pulse,x_pulse_before,y_pulse_before:byte;
count_x_pulse,count_y_pulse,calculate_x_pulse,calculate_y_pulse:longint;
ttp_x_p,ttp_y_p,atp_x_p,atp_y_p:integer;
ratio_ttp_x_pos,ratio_ttp_y_pos,
ratio_atp_x_pos,ratio_atp_y_pos,
ratio_tta_x_pos,ratio_tta_y_pos,
atp_x,atp_y,tta_x,tta_y:real;
{ ═══════════════════════════════════════════════════════════════ }
procedure read_key(key:char);
procedure menu_key_control;
function  yes_or_no:boolean;
function  leading_zero(w:Word):String;
procedure main_window;
procedure child_window;
function  word_to_bin(w,bas,bit:word):string;
procedure set_interface;
procedure load_menu;
procedure messages_load;
procedure load_first_values;
procedure write_message(message_no:byte);
procedure reset_data_file;
procedure load_old_values_to_sav;
procedure find_data_file;
procedure open_data_file;
procedure read_sav_pos_data;
procedure save_data_file;
procedure change_ratio;
procedure list_values;
procedure hardware_control;
procedure get_current_date_time;
```

```
procedure calculate_dif_pos;
procedure new_values_load;
procedure out_port;
procedure in_port;
function  compare_position:boolean;
procedure run_motors;
procedure reset_position;
procedure write_values;
procedure linexy;
procedure load_program;
implementation
{=============================================================================}
        procedure read_key(key:char);
var
tus:char;
        begin
repeat
beep;
tus:=readkey;
until tus=key;
        end;
{ ============================================================================}
        procedure menu_key_control;
var
key:char;
        begin
if keypressed then
begin
key:=readkey;
if key in [#80,#72] then load_menu;
```

```
end;
        end;
{═══════════════════════════════════════════════════════════════════════}
        function yes_or_no;
var
tus:char;
        begin
repeat
beep;
tus:=upcase(readkey);
until tus in ['E','H'];
if tus='E' then yes_or_no:=true else yes_or_no:=false;
        end;
{═══════════════════════════════════════════════════════════════════════}
        function leading_zero(w:Word) : String;
var
s : String;
        begin
Str(w:0,s);
if Length(s) = 1 then
s := '0'+s;
leading_zero := s;
        end;
{═══════════════════════════════════════════════════════════════════════}
        procedure main_window;
        begin
window(1,1,80,25);
        end;
{═══════════════════════════════════════════════════════════════════════}
        procedure child_window;
```

```
        begin
window(2,6,59,22);
        End;
{═══════════════════════════════════════════════════════════════════════}
        function word_to_bin(w,bas,bit:word):string;
var
s:string;
i,j,k:word;
        begin
s:='';
{$Q-}
for i:=bas to bit do
begin
k:=round(exp(i*ln(2)));
if (w and k)>0 then s:='1 '+s else s:='0 '+s;
end;
{$Q+}
word_to_bin:=s;
        End;
{═══════════════════════════════════════════════════════════════════════}
        procedure set_interface;
        begin
clrscr;
cursor_off;
boya(25,25,1,80,1);
boya( 2, 2,1,80,6);
boya( 3, 3,1,80,6);
boya( 4, 4,1,80,0);
boya(24,24,1,80,0);
cerceve('t', 1,1,80, 3,0,3);
```

```
cerceve('c', 1,5,60,23,0,3);
cerceve('c',61,5,80,23,0,3);
gotoxy( 18,2);
m_color(10,6);write('*** Güneş Paneli Kontrol Programı (SPC) v1.0 ***');
gotoxy( 1,25);
m_color(15,2);write(' Eyyüp ÖKSÜZTEPE ');
gotoxy(19,25);
m_color(14,1);write('Yüksek Lisans Bitirme Tezi için yazılmıştır..');
write(' Menü tuşları ',#24,#25);
        end;
{=============================================================================}
        procedure load_menu;
const
panel_menu:menutip=('    Start_Program      ','-',' Reset_Position   ','   Change_Ratio
','Change_Ref_Pos ',' Chg_PeryotTime ',' List_Values   ','-',' Close_Program  ',");
        begin
main_window;
h_menu('y',panel_menu,6,63,9,2,x,true);
case x of
1:begin
load_program;
end;
3:begin
reset_position;
write_message(7);
read_key(#13);
ekran_getir(ekran);
end;
4:begin
change_ratio;
```

```
load_program;
end;
5:begin
end;
6:begin
ekran_sakla(ekran);
çerceve('c',10,9,50,14,10,6);
gotoxy(17,9);write(' □al□Ÿma Peryodunu Değiştir ');
window(11,10,49,13);
m_color(0,6);
clrscr;
main_window;
gotoxy(13,11);write(' Eski ‡al□Ÿma peryodu = ',wait_time);
gotoxy(13,12);write(' Yeni ‡al□Ÿma peryodu = ');
cursor_on;
yread(per_time);
cursor_off;
if per_time<>0 then wait_time:=round(per_time);
ekran_getir(ekran);
load_program;
end;
7:begin
list_values;
load_program;
end;
9:begin
save_data_file;
end_prog;
end;
27:load_program;
```

```
end;
        end;
{=============================================================}
        procedure messages_load;
        begin
data_file_name:='SPC_DATA.DAT';
message_array[1].baslik:=' Uyarı ';
message_array[1].message_line:='Port bağlantısını yeniden kontrol ediniz..! Devam mı ?
(E/H)';
message_array[2].baslik:=' Arama Hatası ';
message_array[2].message_line:=' '+data_file_name+' dosyası bulunamadı.. Yeniden
yazılacak..)';
message_array[3].baslik:=' Dosya Okuma Hatası ';
message_array[3].message_line:=' '+data_file_name+' dosyası okunamadı..';
message_array[4].baslik:=' Programdan çıkış ';
message_array[4].message_line:='Çıkmak istedi§inizden eminmisiniz ? (E/H)';
message_array[5].baslik:=' Dosya Açma Hatası ';
message_array[5].message_line:=' '+data_file_name+' dosyası açılamadı..';
message_array[6].baslik:=' Dosya Yazma Hatası ';
message_array[6].message_line:=' '+data_file_name+' dosyasına şu anki bilgiler
yazılamadı..';
message_array[7].baslik:=' Reset Position ';
message_array[7].message_line:=' Posizyon resetlendi .. ';
        end;
{=============================================================}
        procedure load_first_values;
        begin
output          :=0;
wait_ok         :=false;
wait_time       :=sav_pos.p_time;
```

```
ratio_atp_x_pos    :=sav_pos.x_atp;
ratio_atp_y_pos    :=sav_pos.y_atp;
ratio_tta_x_pos    :=sav_pos.x_tta;
ratio_tta_y_pos    :=sav_pos.y_tta;
old_pos.x_pos      :=sav_pos.x_pos;
old_pos.y_pos      :=sav_pos.y_pos;
old_pos.s_time.day  :=sav_pos.s_time.day;
old_pos.s_time.month:=sav_pos.s_time.month;
old_pos.s_time.year :=sav_pos.s_time.year;
old_pos.s_time.hour :=sav_pos.s_time.hour;
old_pos.s_time.min  :=sav_pos.s_time.min;
old_pos.s_time.sec  :=sav_pos.s_time.sec;
count_x_pulse      :=0;
count_y_pulse      :=0;
        end;
{=============================================================================}
        procedure write_message(message_no:byte);
var
mes_win_x1,mes_win_y1,mes_win_x2,mes_win_y2,win_x:byte;
        begin
mes_win_x1:=((80-length(message_array[message_no].message_line)) div 2);
mes_win_x2:=length(message_array[message_no].message_line)+mes_win_x1+3;
mes_win_y1:=11;
mes_win_y2:=15;
main_window;
ekran_tipi;
ekran_sakla(ekran);
cerceve('c',mes_win_x1,mes_win_y1,mes_win_x2,mes_win_y2,0,4);
win_x:=((mes_win_x2-mes_win_x1)-length(message_array[message_no].baslik))div 2;
gotoxy(mes_win_x1+win_x,mes_win_y1);
```

```
write(message_array[message_no].baslik);
window(mes_win_x1+1,mes_win_y1+1,mes_win_x2-1,mes_win_y2-1);
clrscr;
write(' ',message_array[message_no].message_line,' ');
window(1,1,80,25);
        end;
{=============================================================}
        procedure reset_data_file;
        begin
assign(data_file,data_file_name);
rewrite(data_file);
seek(data_file,1);
sav_pos.x_pos      :=20;
sav_pos.y_pos      :=20;
sav_pos.p_time     :=10;
sav_pos.x_atp      :=2;
sav_pos.y_atp      :=2;
sav_pos.x_tta      :=4.1666666667e-3*480;
sav_pos.y_tta      :=1.4586504310e-6;
get_current_date_time;
sav_pos.s_time.day  :=day;
sav_pos.s_time.month:=month;
sav_pos.s_time.year :=year;
sav_pos.s_time.hour :=5;
sav_pos.s_time.min  :=0;
sav_pos.s_time.sec  :=0;
write(data_file,sav_pos);
close(data_file);
        end;
{=============================================================}
```

```
        procedure load_old_values_to_sav;
        begin
if dif_pos.x_pos<0 then kx:=-1 else kx:=1;
if dif_pos.y_pos<0 then ky:=-1 else ky:=1;
sav_pos.x_pos:=(old_pos.x_pos+round(kx*ratio_atp_x_pos*count_x_pulse))mod 360;
sav_pos.y_pos:=(old_pos.y_pos+round(ky*ratio_atp_y_pos*count_y_pulse))mod 360;
sav_pos.p_time     :=wait_time;
sav_pos.x_atp      :=ratio_atp_x_pos;
sav_pos.y_atp      :=ratio_atp_y_pos;
sav_pos.x_tta      :=ratio_tta_x_pos;
sav_pos.y_tta      :=ratio_tta_y_pos;
sav_pos.s_time.day  :=old_pos.s_time.day;
sav_pos.s_time.month:=old_pos.s_time.month;
sav_pos.s_time.year :=old_pos.s_time.year;
sav_pos.s_time.hour :=old_pos.s_time.hour;
sav_pos.s_time.min  :=old_pos.s_time.min;
sav_pos.s_time.sec  :=old_pos.s_time.sec;
        end;
{=============================================================================}
        procedure find_data_file;
type
pathstr=string[79];
dirstr =string[67];
namestr=string[ 8];
extstr =string[ 3];
var
p:pathstr;
d:dirstr;
n:namestr;
e:extstr;
```

```
s:string;
        begin
getdir(0,s);
p:=fsearch(data_file_name,s);
if p='' then
begin
write_message(2);
read_key(#13);
reset_data_file;
ekran_getir(ekran);
end;
        end;
{=============================================================}
        procedure open_data_file;
var
code:integer;
        begin
assign(data_file,data_file_name);
{$I-}
reset(data_file);
code:=IOResult;
{$I+}
if code<>0 then
begin
write_message(5);
read_key(#13);
reset_data_file;
ekran_getir(ekran);
end;
        end;
```

```
{═══════════════════════════════════════════════════════════════}
        procedure read_sav_pos_data;
        begin
seek(data_file,1);
{$I-}
read(data_file,sav_pos);
{$I+}
if IOResult<>0 then begin write_message(3);readkey;ekran_getir(ekran);end;
close(data_file);
        end;
{═══════════════════════════════════════════════════════════════}
        procedure save_data_file;
        begin
load_old_values_to_sav;
open_data_file;
seek(data_file,1);
{$I-}
write(data_file,sav_pos);
{$I+}
if IOResult<>0 then begin write_message(6);readkey;ekran_getir(ekran);end;
close(data_file);
        end;
{═══════════════════════════════════════════════════════════════}
        procedure change_ratio;
        begin
main_window;
ekran_sakla(ekran);
cerceve('c',6,8,54,17,0,7);
gotoxy(21,8);write(' Oranları Değiştir ');
window(7,9,53,16);
```

```
m_color(0,7);
clrscr;
cursor_on;
main_window;
gotoxy(24,10);write(' Eski Oran      Yeni Oran');
gotoxy(24,11);write('============  ============');
gotoxy(10,12);write('x_atp oranı = ',ratio_atp_x_pos:0);
gotoxy(10,13);write('x_tta oranı = ',ratio_tta_x_pos:0);
gotoxy(10,14);write('y_atp oranı = ',ratio_atp_y_pos:0);
gotoxy(10,15);write('y_tta oranı = ',ratio_tta_y_pos:0);
gotoxy(38,12);yread(atp_x);
gotoxy(38,13);yread(tta_x);
gotoxy(38,14);yread(atp_y);
gotoxy(38,15);yread(tta_y);
if atp_x<>0 then ratio_atp_x_pos:=atp_x;
if atp_y<>0 then ratio_atp_y_pos:=atp_y;
if tta_x<>0 then ratio_tta_x_pos:=tta_x;
if tta_y<>0 then ratio_tta_y_pos:=tta_y;
cursor_off;
ekran_getir(ekran);
        end;
{=============================================================================}
        procedure list_values;
var
tus:char;
        begin
main_window;
ekran_sakla(ekran);
cerceve('c',6,8,56,20,14,7);
gotoxy(25,8);write(' List Values ');
```

```
window(7,9,55,19);
clrscr;
repeat
get_current_date_time;
calculate_dif_pos;
window(7,9,55,19);
m_color(0,7);
gotoxy(10,1);write('Saved Values    Current Values');
gotoxy(2, 3);write('X_pos : ',sav_pos.x_pos :12,'    ',new_pos.x_pos  :12,'  deg');
gotoxy(2, 4);write('Y_pos : ',sav_pos.y_pos :12,'    ',new_pos.y_pos  :12,'  deg');
gotoxy(2, 5);write('X_atp : ',sav_pos.x_atp :12,'    ',ratio_atp_x_pos:12,'  deg/pulse');
gotoxy(2, 6);write('Y_atp : ',sav_pos.y_atp :12,'    ',ratio_atp_y_pos:12,'  deg/pulse');
gotoxy(2, 7);write('X_tta : ',sav_pos.x_tta :12,'    ',ratio_tta_x_pos:12,'  sec/deg');
gotoxy(2, 8);write('Y_tta : ',sav_pos.y_tta :12,'    ',ratio_tta_y_pos:12,'  sec/deg');
gotoxy(2, 9);write('Peryot: ',sav_pos.p_time:12,'    ',wait_time      :12,'  sec');
gotoxy(2,10);write('Date  : ');
gotoxy(2,11);write('Time  : ');
gotoxy(12,10);write(leading_zero(sav_pos.s_time.day));
write('/',leading_zero(sav_pos.s_time.month));
write('/',leading_zero(sav_pos.s_time.year));
gotoxy(12,11);write(leading_zero(sav_pos.s_time.hour));
write(':',leading_zero(sav_pos.s_time.min));
write(':',leading_zero(sav_pos.s_time.sec));
gotoxy(28,10);write(leading_zero(new_pos.s_time.day));
write('/',leading_zero(new_pos.s_time.month));
write('/',leading_zero(new_pos.s_time.year):4,'  d/m/y');
gotoxy(28,11);write(leading_zero(new_pos.s_time.hour));
write(':',leading_zero(new_pos.s_time.min));
write(':',leading_zero(new_pos.s_time.sec),'h:m:s':9);
if keypressed then tus:=readkey;
```

```
until tus=#13;
ekran_getir(ekran);
        end;
{=============================================================}
        procedure hardware_control;
        begin
if (input and 135)=135 then
else
begin
write_message(1);
if yes_or_no then ekran_getir(ekran) else end_prog;
end;
        end;
{=============================================================}
        procedure get_current_date_time;
const
days : array [0..6] of String[9] =
('Pazar','Pazartesi','Salı','Çarşamba','Perşembe','Cuma','Cumartesi');
        begin
GetTime(hour,min,sec,hund);
GetDate(year,month,day,dow);
main_window;
m_color(10,0);
gotoxy(63,4);
Write('Time:',leading_zero(hour),':',leading_zero(min),':',leading_zero(sec),'.',leading_zero(
hund));
gotoxy( 1,4);
Write('Date  : ',leading_zero(day),'/',leading_zero(month),'/',year,'..',days[dow]);
new_pos.s_time.hour :=hour;
new_pos.s_time.min  :=min;
```

```
new_pos.s_time.sec  :=sec;
new_pos.s_time.year :=year;
new_pos.s_time.month:=month;
new_pos.s_time.day  :=day;
        end;
{=============================================================}
        procedure calculate_dif_pos;
        begin
ratio_ttp_x_pos:=ratio_atp_x_pos*ratio_tta_x_pos;
ratio_ttp_y_pos:=ratio_atp_y_pos*ratio_tta_y_pos;
atp_x_p         :=round(360/ratio_atp_x_pos);
atp_y_p         :=round(360/ratio_atp_y_pos);
packtime(old_pos.s_time,old_pos_int);
packtime(new_pos.s_time,new_pos_int);
dif_pos_int:=new_pos_int-old_pos_int;
new_pos.x_pos:=((round(dif_pos_int/ratio_tta_x_pos)+old_pos.x_pos)mod 360);
new_pos.y_pos:=((round(dif_pos_int/ratio_tta_y_pos)+old_pos.x_pos)mod 360);
dif_pos.x_pos:=new_pos.x_pos-old_pos.x_pos;
dif_pos.y_pos:=new_pos.y_pos-old_pos.y_pos;
unpacktime(dif_pos_int,dif_pos.s_time);
dif_pos.s_time.year:=dif_pos.s_time.year-1980;
calculate_x_pulse:=(abs(round(dif_pos.x_pos/ratio_atp_x_pos)mod atp_x_p));
calculate_y_pulse:=(abs(round(dif_pos.y_pos/ratio_atp_y_pos)mod atp_y_p));
if dif_pos.x_pos>0 then direction_x_motor:=16;
if dif_pos.x_pos<0 then direction_x_motor:= 8;
if dif_pos.x_pos>0 then direction_y_motor:= 2;
if dif_pos.y_pos<0 then direction_y_motor:= 1;
        end;
{=============================================================}
        procedure new_values_load;
```

```
        begin
old_pos.x_pos      :=new_pos.x_pos;
old_pos.y_pos      :=new_pos.y_pos;
old_pos.p_time     :=wait_time;
old_pos.x_atp      :=ratio_atp_x_pos;
old_pos.y_atp      :=ratio_atp_y_pos;
old_pos.x_tta      :=ratio_tta_x_pos;
old_pos.y_tta      :=ratio_tta_y_pos;
old_pos.s_time.day  :=new_pos.s_time.day;
old_pos.s_time.month:=new_pos.s_time.month;
old_pos.s_time.year :=new_pos.s_time.year;
old_pos.s_time.hour :=new_pos.s_time.hour;
old_pos.s_time.min  :=new_pos.s_time.min;
old_pos.s_time.sec  :=new_pos.s_time.sec;
        end;
{=============================================================}
        procedure out_port;
        begin
output:=(direction_x_motor or direction_y_motor);
port[$378]:=output;
        end;
{=============================================================}
        procedure in_port;
        begin
input:=port[$379];
        end;
{=============================================================}
        procedure in_port_control;
        begin
x_pulse:=input and 16;
```

```
y_pulse:=input and 8;
if (x_pulse_before <> x_pulse) then
begin
inc(count_x_pulse);
x_pulse_before:=x_pulse;
end;
if (y_pulse_before <> y_pulse) then
begin
inc(count_y_pulse);
y_pulse_before:=y_pulse;
end;
hardware_control;
if (input and 64)>=64 then reset_code:=true else reset_code:=false;
        end;
{=============================================================}
        function compare_position:boolean;
        begin
if        ((count_x_pulse>=calculate_x_pulse)and(count_y_pulse>=calculate_y_pulse))then
compare_position:=true else compare_position:=false;
        end;
{=============================================================}
        procedure run_motors;
var
tus:char;
        begin
repeat
get_current_date_time;
calculate_dif_pos;
if  count_x_pulse>=calculate_x_pulse then direction_x_motor:=direction_x_motor and 3;
if  count_y_pulse>=calculate_y_pulse then direction_y_motor:=direction_y_motor and 12;
```

```
out_port;
in_port;
in_port_control;
write_values;
menu_key_control;
until (compare_position);
output:=0;
out_port;
wait_ok:=false;
        end;
{=============================================================}
        procedure reset_position;
        begin
repeat
direction_x_motor:=8;
direction_y_motor:=1;
out_port;
in_port;
in_port_control;
until reset_code;
reset_data_file;
        end;
{=============================================================}
        procedure linexy;
var
i,j:integer;
        begin
m_color(9,0);
j:=2;
repeat
```

```
for i:=1 to 58 do
begin
gotoxy(i,j);
if i in[12,24,36,48] then write(#197) else write(#196);
end;
inc(j,2);
until j>13;
j:=12;
repeat
for i:=2 to 12 do
begin
gotoxy(j,i);
if i in[2,4,6,8,10,12,14] then write(#197) else write(#179);
end;
inc(j,12);
until j>58;
        end;
{═══════════════════════════════════════════════════════════════════════}
        procedure write_values;
var
z:byte;
        begin
child_window;
linexy;
m_color(12,0);
gotoxy(13, 1);write('Save_Value ');
gotoxy(26, 1);write('Old_Value ');
gotoxy(38, 1);write('New_Value ');
gotoxy(49, 1);write('Dif_Value');
gotoxy( 2, 3);write('X_Position');
```

```
gotoxy( 2, 5);write('Y_Position');
gotoxy( 2, 7);write('Date (dmy)');
gotoxy( 2, 9);write('Time (hms)');
gotoxy( 2,11);write('Pack_Time ');
m_color(7,0);
gotoxy(14, 3);write(sav_pos.x_pos:10);
gotoxy(26, 3);write(old_pos.x_pos:10);
gotoxy(38, 3);write(new_pos.x_pos:10);
gotoxy(49, 3);write(dif_pos.x_pos:10);
gotoxy(14, 5);write(sav_pos.y_pos:10);
gotoxy(26, 5);write(old_pos.y_pos:10);
gotoxy(38, 5);write(new_pos.y_pos:10);
gotoxy(49, 5);write(dif_pos.y_pos:10);
gotoxy(14, 7);write(leading_zero(sav_pos.s_time.day));
write('/',leading_zero(sav_pos.s_time.month));
write('/',leading_zero(sav_pos.s_time.year));
gotoxy(14, 9);write(leading_zero(sav_pos.s_time.hour));
write(':',leading_zero(sav_pos.s_time.min));
write(':',leading_zero(sav_pos.s_time.sec));
gotoxy(26, 7);write(leading_zero(old_pos.s_time.day));
write('/',leading_zero(old_pos.s_time.month));
write('/',leading_zero(old_pos.s_time.year));
gotoxy(26, 9);write(leading_zero(old_pos.s_time.hour));
write(':',leading_zero(old_pos.s_time.min));
write(':',leading_zero(old_pos.s_time.sec));
gotoxy(38, 7);write(leading_zero(new_pos.s_time.day));
write('/',leading_zero(new_pos.s_time.month));
write('/',leading_zero(new_pos.s_time.year));
gotoxy(38, 9);write(leading_zero(new_pos.s_time.hour));
write(':' ,leading_zero(new_pos.s_time.min));
```

```
write(':' ,leading_zero(new_pos.s_time.sec));
gotoxy(50, 7);write(leading_zero(dif_pos.s_time.day));
write('/',leading_zero(dif_pos.s_time.month));
write('/',leading_zero(dif_pos.s_time.year));
gotoxy(50, 9);write(leading_zero(dif_pos.s_time.hour));
write(':' ,leading_zero(dif_pos.s_time.min));
write(':' ,leading_zero(dif_pos.s_time.sec));
gotoxy(13,11);write(sav_pos_int:10);
gotoxy(25,11);write(old_pos_int:10);
gotoxy(37,11);write(new_pos_int:10);
gotoxy(49,11);write(dif_pos_int:10);
m_color(10,0);
gotoxy( 15,13);write('counted   ');
gotoxy( 26,13);write('calculate ');
m_color(15,0);
gotoxy(  2,15);
write('X_Pulses :',Count_x_pulse:8,calculate_x_pulse:12);
gotoxy( 2,16);
write('Y_Pulses :',Count_y_pulse:8,calculate_y_pulse:12);
m_color(9,0);
for z:=13 to 16 do begin gotoxy(36,z);write(#179);end;
m_color(7,0);
gotoxy(49,13);write('X X   Y Y');
gotoxy(49,14);write('I G   I G');
gotoxy(38,15);write('Output  : ');
gotoxy(38,16);write('Input   : ');
gotoxy(49,17);write('H R   X Y');
m_color(12,0);
gotoxy(48,15);write(word_to_bin(output,0,4):11);
gotoxy(48,16);write(word_to_bin(input,3,7):11);
```

```
        end;
{=============================================================}
        procedure load_program;
var
key:char;
        begin
gotoxy(1,24);
child_window;
m_color(0,0);
CLRSCR;
write_values;
repeat
get_current_date_time;
if ((min*60)+sec) mod wait_time=0 then wait_ok:=true;
if wait_ok then
begin
run_motors;
new_values_load;
save_data_file;
count_x_pulse:=0;
count_y_pulse:=0;
end;
menu_key_control;
until 1=2;
        end;
{=============================================================}
        end.
```

```
{==================================================================}
unit My_Unit;
 {------------------------------------------------------------}
interface
{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}
uses Crt,Dos;
type
menutip=array[1..10] of string;
{==================================================================}
procedure Cursor_On ;(* Kursörü ekranda görünür yapar *)
procedure Cursor_Off;(* Kursörü ekranda görünmez yapar *)
procedure Beep     ;(* Beep sesi çıkartır *)
procedure M_Color(Text,Background:Byte);(* Yazının ve arkazeminin rengini ayarlar *)
procedure End_Prog ;(* Programı sona erdirir *)
procedure Show(Satir:Byte;gg:Boolean);
procedure XRead(var Data:Real);(* Sadece sayı ve rakam okuması için *)
procedure yRead(var Datay:Real);(* Sadece say ve rakam okuması için *)
procedure Ekran_Tipi ;(* Ekran saklarken kullanılmalı *)
procedure Ekran_Sakla(var Ekr : Pointer);(* Ekranı saklar *)
procedure Ekran_Getir(var Ekr : Pointer);(* Saklanan ekranı geri getirir *)
procedure Boya(xBas,xBit,yBas,yBit,Renk:Byte);
procedure Cerceve (Sek:Char;x1,y1,x2,y2,tc,tb:Byte) ;
procedure H_Menu (sekil  : char;
menu   : menutip;
satir,sutun,elsay,artis : byte;
var x  : byte;
escevet: boolean );
{==================================================================}
implementation
var
```

```
Reg : Registers;
Ekr_Tipi:Word;
Ekran_1:Pointer;
{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}
        procedure Cursor_On;
        begin
Reg.CX:=$C0D;
Reg.BX:=0;
Reg.AX:=$100;
Intr($10,Reg);
        end;
{***********************************************************************}
        procedure Cursor_Off;
        begin
Reg.CX:=$2000;
Reg.BX:=0;
Reg.AX:=$100 ;
Intr($10,Reg);
        end;
{***********************************************************************}
        procedure Beep;
        begin
Sound(1000);Delay(20);NoSound;
        end;
{***********************************************************************}
        procedure M_Color(Text,Background:Byte);
        begin
TextColor(Text);
TextBackground(Background);
        end;
```

```
{*************************************************************************}
        procedure End_Prog;
        begin
window(1,1,80,25);
m_color(7,0);
ClrScr;
Cursor_On;
Port[$378]:=0;
Halt;
        end;
{*************************************************************************}
        procedure Show(Satir:Byte;gg:Boolean);
var
i,cc   :Byte;
ch,stus:Char;
        begin
ch:='Í';
GotoXY(1,Satir);
for i:=0 to 78 do begin M_Color(i mod(15)+1,0);Write(ch);end;
while gg do
begin
for i:=0 to 78 do
begin
GotoXY(i+1,Satir); M_Color(i mod(8)+9,0); Write('*');
Delay(10);
GotoXY(i+1,Satir); M_Color(i mod(8)+1,0); Write(ch);
end;
gg:=False;
end;
M_Color(7,0);
```

```
        end;
{*************************************************************************}
        procedure XRead(var Data:Real);
label cc,zz,vv;
var
Sayac,SayacN,SayacX : Byte;
S : string;
ss: string[1];
tus,ttus,sx: Char;
xx,yy,q: Byte;
Code : Integer;
        begin
repeat
xx:=WhereX;
yy:=WhereY;
tus:=#13;
S:='';
Sayac :=0;
SayacN:=0;
SayacX:=0;
vv:   repeat
if sayac >=2 then Sayac :=Sayac -1;
if sayacn>=2 then SayacN:=SayacN-1;
if sayacx>=3 then SayacX:=SayacX-1;
tus:=ReadKey;
if Ord(tus)=8 then
begin
ss:=Copy(S,Length(S),1);
if ss= '-' then SayacX:=SayacX-1;
if ss= '.' then SayacN:=SayacN-1;
```

```
if ss= 'e' then Sayac :=Sayac -1;
if ss= 'E' then Sayac :=Sayac -1;
Delete(S,Length(S),1);
GotoXY(xx,yy);
Write(S+' ');
GotoXY(xx+Length(S)-1,yy);
end;
if tus=#27 then XRead(Data);
if tus=#13 then goto cc;
if tus in['1','2','3','4','5','6','7','8','9','0','e','E','.','-']then
begin
if (tus='e') or (tus='E') then sayac:=sayac+1;
if (tus='.') then SayacN:=sayacN+1;
if (tus='-') then SayacX:=sayacX+1;
if (SayacN>=2) or (SayacX>=3) or (Sayac>=2) then begin Beep;goto vv;end;
S:=S+tus;
GotoXY(xx,yy);
Write(S);
end else
begin
Beep;
end;
until 1=2;
cc:  Val(S,Data,Code);
if Code <> 0 then
begin
if Code=1 then
begin
GotoXY(16,24);
Write(' Bu eleman için bir değer atamanız gerekiyor.! ');
```

```
end else
begin
GotoXY(21,24);
Write(' Yanlış değer girildi , düzeltiniz..! ');
end;
repeat
Beep;
ttus:=ReadKey;
if ttus=#13 then
begin
DelLine;
InsLine;
Show(24,False);
Beep;
GotoXY(xx+Length(S),yy);
goto vv;
end;
if ttus=#27 then End_Prog;
until 1=2;
end;
Exit;
until 1=2;
        end;
{*************************************************************************}
        procedure yRead(var Datay:Real);
label cc,zz,vv;
var
Sayac,SayacN,SayacX : Byte;
S: string;
ss: string[1];
```

```
tus,ttus,sx : Char;
xx,yy,q : Byte;
Code : Integer;
        begin
repeat
xx:=WhereX;
yy:=WhereY;
tus:=#13;
S:='';
Sayac :=0;
SayacN:=0;
SayacX:=0;
vv:   repeat
if sayac >=2 then Sayac :=Sayac -1;
if sayacn>=2 then SayacN:=SayacN-1;
if sayacx>=3 then SayacX:=SayacX-1;
tus:=ReadKey;
if Ord(tus)=8 then
begin
ss:=Copy(S,Length(S),1);
if ss= '-' then SayacX:=SayacX-1;
if ss= '.' then SayacN:=SayacN-1;
if ss= 'e' then Sayac :=Sayac -1;
if ss= 'E' then Sayac :=Sayac -1;
Delete(S,Length(S),1);
GotoXY(xx,yy);
Write(S+' ');
GotoXY(xx+Length(S)-1,yy);
end;
if tus=#13 then goto cc;
```

```
if tus in['1','2','3','4','5','6','7','8','9','0','e','E','.','-']then
begin
if (tus='e') or (tus='E') then sayac:=sayac+1;
if (tus='.') then SayacN:=sayacN+1;
if (tus='-') then SayacX:=sayacX+1;
if (SayacN>=2) or (SayacX>=3) or (Sayac>=2) then begin Beep;goto vv;end;
S:=S+tus;
GotoXY(xx,yy);
Write(S);
end else
begin
Beep;
end;
until 1=2;
cc:  Val(S,Datay,Code);
if Code <> 0 then
begin
if Code=1 then
begin
exit;
end else
begin
GotoXY(21,24);
Write(' Yanlış değer girildi , düzeltiniz..! ');
end;
repeat
Beep;
ttus:=ReadKey;
if ttus=#13 then
begin
```

```
DelLine;
InsLine;
Beep;
GotoXY(xx+Length(S),yy);
goto vv;
end;
until 1=2;
end;
Exit;
until 1=2;
        end;
{*************************************************************************}
        procedure Ekran_Tipi;
        begin
if Mem[$0000:$049]=7 then Ekr_Tipi:=$B100 else Ekr_Tipi:=$B800;
        end;
{*************************************************************************}
        procedure Ekran_Sakla ( var Ekr : Pointer);
        begin
GetMem(Ekr,4000);
Move(ptr(Ekr_Tipi,0)^,Ekr^,4000);
        end;
{*************************************************************************}
        procedure Ekran_Getir ( var Ekr : Pointer);
        begin
Move(Ekr^,ptr(Ekr_Tipi,0)^,4000);
FreeMem(Ekr,4000);
        end;
{*************************************************************************}
        procedure Boya(xBas,xBit,yBas,yBit,Renk:Byte);
```

```
var
i,j:Integer;
        begin
M_Color(Renk,Renk);
for i:=xBas to xBit do
for j:=yBas to yBit do
begin
GotoXY(j,i);Write('Û');
end;
        end;
{*************************************************************************}
        procedure Cerceve (Sek:Char;x1,y1,x2,y2,tc,tb:Byte) ;
var
i,j   : Integer;
Cizgi : string[80];
        begin
M_Color(tc,tb);
case Sek of
'T','t' :begin
FillChar(Cizgi,80,#196);Cizgi[0]:=Char(x2-x1-1);
GotoXY(x1,y1);Write(Chr(218),Cizgi,Chr(191));
for i:=y1+1 to y2-1 do
        begin
GotoXY(x1,i);Write(Chr(179));
GotoXY(x2,i);write(Chr(179));
end;
GotoXY(x1,y2);Write(Chr(192),Cizgi,Chr(217));
end;
'c','C' :begin
FillChar(Cizgi,80,#205);Cizgi[0]:=Char(x2-x1-1);
```

```
GotoXY(x1,y1);Write(Chr(201),Cizgi,Chr(187));
for i:=y1+1 to y2-1 do
begin
GotoXY(x1,i);Write(Chr(186));
GotoXY(x2,i);Write(Chr(186));
end;
GotoXY(x1,y2);Write(Chr(200),Cizgi,Chr(188));
end;
end;
        end;
{*************************************************************************}
        procedure H_Menu(sekil  : char;
menu   : menutip;
satir,sutun,elsay,artis : byte;
var x  : byte;
escevet: boolean );
var
sat,sut,i : byte;
tus : char;
        begin
sekil:=upcase(sekil);
sat:=satir;sut:=sutun;
for i:= 1 to elsay do
begin
gotoxy(sut,sat);
m_color(15,10);
if menu[i]='-' then m_color(3,3);
write(menu[i]);
if sekil='Y' then sat:=sat+artis else sut:=sut+artis;
end;
```

```
sut:=sutun;sat:=satir;
gotoxy(sut,sat);m_color(15,5);write(menu[1]);x:=1;
repeat
tus:=readkey;
if keypressed then tus:=readkey;
m_color(15,10);
case sekil of
'S': if (tus=#77) or (tus=#75) then
begin
gotoxy(sut,sat);write(menu[x]);
if tus=#77 then  {SAG OK}
if x>elsay-1 then
begin  x:=1;sut:=sutun;end
else
begin
x:=x+1;sut:=sut+artis;
while (menu[x]='-') do
begin
x:=x+1;sut:=sut+artis;
end;
end;
if (tus=#75) then {SOL OK}
if x<2 then
begin x:=elsay;sut:=sutun+(elsay-1)*artis;end
else
begin
x:=x-1;sut:=sut-artis;
while (menu[x]='-') do
begin
x:=x-1;sut:=sut-artis;
```

```
end;
end;
gotoxy(sut,sat);m_color(15,5);write(menu[x]);
end;
'Y': if (tus=#80) or (tus=#72) then
begin
gotoxy(sut,sat);write(menu[x]);
if tus=#80 then  {ASAGI OK}
if x>elsay-1 then
begin  x:=1;sat:=satir;end
else
begin
x:=x+1;sat:=sat+artis;
while (menu[x]='-') do
begin
x:=x+1;sat:=sat+artis;
end;
end;
if tus=#72 then {YUKARI OK}
if x<2 then
begin x:=elsay;sat:=satir+(elsay-1)*artis;end
else
begin
x:=x-1;sat:=sat-artis;
while (menu[x]='-') do
begin
x:=x-1;sat:=sat-artis;
end;
end;
gotoxy(sut,sat);m_color(15,5);write(menu[x]);
```

```
end;
end;
if (tus =#27) and (escevet) then
begin
gotoxy(sut,sat);
m_color(15,10);
write(menu[x]);
x:=27;exit;
end;
until tus=#13;
gotoxy(sut,sat);m_color(15,10);write(menu[x]);
        end;
{++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++}
        end.
```