T.C.
NECMETTİN ERBAKAN
ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KONYA ŞARTLARINDA GÜNEŞ TAKİPLİ BİR PV SİSTEMİ İLE DİĞER PV SİSTEMLERİNİN VERİMLERİNİN KARŞILAŞTIRILMASI

Furkan BÖRK

YÜKSEK LİSANS TEZİ

Enerji Sistemleri Mühendisliği Anabilim Dalı

Şubat-2024
KONYA

TEZ KABUL VE ONAYI

Furkan Börk tarafından hazırlanan "Konya Şartlarında Güneş Takipli Bir PV Sistemi İle Diğer PV Sistemlerinin Verimlerinin Karşılaştırılması" adlı tez çalışması 01/02/2024 tarihinde aşağıdaki jüri tarafından oy birliği ile Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü Enerji Sistemleri Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS tezi olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Doç. Dr. Mustafa Tahir AKKOYUNLU	……..
Danışman Dr. Öğr. Üyesi Yılmaz AKGÜNEY	……..
Üye Dr. Öğr. Üyesi Sadık ATA	……..

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun …./…./……. gün ve …….. sayılı kararıyla onaylanmıştır.

FBE Müdürü

Prof. Dr. Şerife Yurdagül KUMCU

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this thesis document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Furkan BÖRK
09.01.2024

ÖZET

YÜKSEK LİSANS TEZİ

KONYA ŞARTLARINDA GÜNEŞ TAKİPLİ BİR PV SİSTEMİ İLE DİĞER PV SİSTEMLERİNİN VERİMLERİNİN KARŞILAŞTIRILMASI

Furkan BÖRK

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü

Enerji Sistemleri Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Yılmaz AKGÜNEY

2024, 123 Sayfa

Jüri
Dr. Öğr. Üyesi Yılmaz AKGÜNEY
Doç. Dr. Mustafa Tahir AKKOYUNLU
Dr. Öğr. Üyesi Sadık ATA

Bu tez çalışmasında 3 adet 10 W gücünde fotovoltaik (PV) güneş panellerinin çift eksenli güneş takip sistemi, tek eksenli güneş takip sistemi ve sabit açılı güneş panelinin verim analizleri yapıldı. Kullanılan güneş takip sisteminin temel amacı güneş panellerinin üzerine gelen güneş ışınlarının güneş takip sisteminin izin verdiği eksenler arasında güneş panelinin üzerine dik gelmesi üzerinde çalışmaktadır. Takip sistemi temel olarak Arduino Mega mikro kontrol işlemcisi ve ışığa bağlı dirençler ile sağlanmıştır. Elde edilen sensör verilerinin panellerin hareketi için kullanılması için 3 adet DC motor kullanılmıştır ve DC motordan elde edilen torkun artırılması için dişli kutuları eklenmiştir. Sabit açılı güneş panelinin Konya ili için yaz mevsimi boyunca maksimum verim alabilmesi için 37 ° güney yönünde açı verilmiştir. Tek eksenli güneş takip sistemi sadece doğu batı yönünde hareket ettirilmiş ve güney kuzey yönünde bir açısı yoktur. Çift eksenli güneş takip sistemi hem kuzey ve güney hem de doğu ve batı yönünde güneş takibi sağlamaktadır. Elde edilen ölçümler sonucunda 20 Temmuz tarihinde günlük çift eksenli güneş takip sistemi 64 Wh, tek eksenli güneş takip sistemi 56.62 Wh, sabit açılı güneş paneli ise 45.87 Wh enerji üretmiştir. Sabit açılı güneş paneli ile karşılaştırıldığında çift eksenli güneş takip sisteminin verimi %39.5 ve tek eksenli güneş takip sistemi %23.5 fazla verime sahip olduğu görülmüştür. 16 Ağustos günlük çift eksenli güneş takip sistemi 57.68 Wh, tek eksenli güneş takip sistemi 51.68 Wh, sabit açılı güneş paneli ise 42.89 Wh enerji üretmiştir. Sabit açılı güneş paneli ile karşılaştırıldığında çift eksenli güneş takip sisteminin verimi %34.5 ve tek eksenli güneş takip sistemi %21 fazla verime sahip olduğu görülmüştür. 6 Ocak ile 7 Ocak ile yapılan ölçümlerin sonucunda, çift eksenli güneş takip sistemi 32.6 Wh ve 15.1 Wh, tek eksenli güneş takip sistemi 25.97 Wh ve 9.14 Wh, sabit açılı güneş paneli 20.62 Wh ve 6.74 Wh enerji üretmiştir. Yapılan ölçümlerin sonucunda havanın bulutluluk durumuna bağlı olarak sistemin enerji çıkışları incelenmiştir.

Anahtar kelimeler: Arduino, Fotovoltaik Güneş Paneli, Güneş Takip Sistemi, Konya, Verim Analizi

ABSTRACT

MS

COMPARISON OF THE EFFICIENCY OF A SOLAR TRACKING PV SYSTEM AND OTHER PV SYSTEMS IN KONYA CONDITIONS

Furkan BÖRK

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF NECMETTİN ERBAKAN UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE
IN ENERGY SYSTEMS ENGINEERING

Advisor: Dr. Öğr. Üyesi Yılmaz AKGÜNEY

2024, 123 Sayfa

Jury
Dr. Öğr. Üyesi Yılmaz AKGÜNEY
Doç. Dr. Mustafa Tahir AKKOYUNLU
Dr. Öğr. Üyesi Sadık ATA

In this thesis study, the efficiency analyzes of three 10 W photovoltaic (PV) solar panels, dual-axis solar tracking system, single-axis solar tracking system and fixed angle solar panel were performed. The main purpose of the solar tracking system used is to ensure that the sun rays falling on the solar panels are perpendicular to the solar panel within the axes allowed by the solar tracking system. The tracking system is basically provided by the Arduino Mega micro control processor and light-dependent resistors. Three DC motors were used to use the obtained sensor data for the movement of the panels, and gear boxes were added to increase the torque obtained from the DC motor. In order for the fixed angle solar panel to get maximum efficiency during the summer season for Konya province, the angle is given at 37 ° south. The single-axis solar tracking system is moved only in the east-west direction and does not have an angle in the south-north direction. The dual-axis solar tracking system provides sun tracking in both north and south and east and west directions. As a result of the measurements obtained, on July 20, the daily dual-axis solar tracking system produced 64 Wh, the single-axis solar tracking system produced 56.62 Wh, and the fixed-angle solar panel produced 45.87 Wh of energy. Compared to the fixed-angle solar panel, the efficiency of the dual-axis solar tracking system was 39.5% higher and the single-axis solar tracking system was found to be 23.5% more efficient. On August 16, the dual-axis solar tracking system produced 57.68 Wh of energy, the single-axis solar tracking system produced 51.68 Wh, and the fixed-angle solar panel produced 42.89 Wh of energy. Compared to the fixed-angle solar panel, the efficiency of the dual-axis solar tracking system was 34.5% higher and the single-axis solar tracking system was 21% more efficient. As a result of the measurements made on January 6 and January 7, the dual-axis solar tracking system produced 32.6 Wh and 15.1 Wh, the single-axis solar tracking system produced 25.97 Wh and 9.14 Wh, and the fixed-angle solar panel produced 20.62 Wh and 6.74 Wh. As a result of the measurements, the energy output of the system was examined depending on the cloudiness of the air.

Keywords: Arduino, Efficiency Analysis, Konya, Photovoltaic Solar Panel, Solar Tracking System

ÖNSÖZ

Çalışmamın başlangıç aşamasından, son halini almasına kadar yol gösterici olan ve tüm tecrübesini benimle paylaşan danışmanım Sayın Dr. Öğretim Üyesi Yılmaz AKGÜNEY ile hayatım boyunca maddi & manevi desteklerini hiçbir zaman esirgemeyen ve her zaman yanımda olan çok kıymetli babam Hasan BÖRK'e ve çok kıymetli annem Havva BÖRK'e tüm kalbimle teşekkür ederim.

Furkan BÖRK
KONYA-2024

İÇİNDEKİLER

ÖZET ..... **iii**

ABSTRACT ..... **iv**

ÖNSÖZ ..... **iv**

İÇİNDEKİLER ..... **v**

SİMGELER VE KISALTMALAR ..... **viii**

ŞEKİLLER LİSTESİ ..... **x**

ÇİZELGELER LİSTESİ ..... **xii**

1. GİRİŞ **1**

1.1. Güneş 2

1.1.1. Güneş ışığının özellikleri 3

1.1.2. Güneş ışınlarının geliş açısını etkileyen faktörler 4

1.1.2.1. Gün içerisinde güneşin doğu-batı hareketi 4

1.1.2.2. Güneşin mevsimsel kuzey-güney hareketi 5

1.2. Enerjinin İnsanlık Üzerindeki Etkisi 6

1.3. Güneş Enerjisinin Kullanım Alanları 7

1.3.1. Güneş enerjisinin ısıl uygulamaları 7

1.4. Güneş Enerjisinden Elektrik Üretilmesi 8

1.4.1. CSP güç santralleri 9

1.4.2. PV güç santralleri 10

1.4.3. PV güç santrallerini etkileyen faktörler 10

1.4.3.1. Uygun saydam iletken seçimi 11

1.4.3.2. Arka yüzey pasifleştirilmesi 11

1.4.3.3. Yansıma engelleyici kaplamalar ve yapılar 11

2. KAYNAK ARAŞTIRMASI **12**

3. MATERYAL VE YÖNTEM **17**

3.1. Materyal 17
3.1.1. Güneş paneli 17
3.1.2. Güneş panellerini taşıyan platform 18
3.1.3. Dişli kutusu ve rulmanlar 19
3.1.4. Redüktörlü DC motor 21
3.1.5. Motor konumu ve kalibrasyonu 22
3.1.6. L298N DC motor sürücü modülü 24
3.1.7. Arduino Mega 2560 R3 Mikro Denetleyici 25
3.1.8. Arduino Mikro Hafıza Kardı Modülü 26
3.1.9. DS1302 Gerçek zamanlı saat modülü 27
3.1.10. Foto dirençler 28
3.2. Yöntem 29
3.2.1. Panel güç ölçüm sistemi 29
3.2.2. Hafıza Kartı Kayıt Yazılımı 31

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA 36

4.1. Araştırma sonuçları 36
4.2. Tartışma 51

5. SONUÇLAR VE ÖNERİLER 55

5.1. Sonuçlar 55
5.2. Öneriler 58

6. KAYNAKLAR 60

7. EKLER 65

EK-1 Arduino Mega yazılımı setup_loop.ino dosyası 65
EK-2 Arduino Mega yazılımı motor_Control.cpp dosyası 68
EK-3 Arduino Mega yazılımı motor_Control.h dosyası 72
EK-4 Arduino Mega yazılımı SdCardHandler.cpp dosyası 73
EK-5 Arduino Mega yazılımı SdCardHandler.h dosyası 75
EK-6 Arduino Mega yazılımı ldr_sensor.cpp dosyası 76
EK-7 Arduino Mega yazılımı ldr_sensor.h dosyası 79

EK-8 Arduino Mega yazılımı functions.ino dosyası..... 80
EK-9 20 Temmuz tarihleri arasında yapılan ölçümler çizelge..... 103
EK-10 16 Ağustos tarihleri arasında yapılan ölçümler çizelge..... 105
EK-11 20 Temmuz tarihleri arasında yapılan ölçümler grafikler 107
EK-12 16 Ağustos tarihleri arasında yapılan ölçümler grafikler 108

SİMGELER VE KISALTMALAR

Simgeler

A	Akım (Amper)
mA	mili Amper
g	Ağırlık (gram)
kg	Ağırlık (Kilogram, 1000 g)
m^2	Alan (Metrekare)
kB	Bellek ölçüm birimi (Kilo bayt)
Cos	Kosinüs
δ	Deklinasyon açısı (°)
V	Elektrik gerilimi (Volt)
J	Enerji (Joule)
Φ	Enlem açısı (°)
β	Eğim açısı (°)
N	Güç (Newton)
kW	Güç (Kilo Watt)
θ	Güneş geliş açısı (°)
α_s	Güneş yükseklik açısı (°)
γ_s	Güneş azimut açısı (°)
l	Hacim (Litre, $m^3/1000$)
ω	Saat açısı (°)
°C	Sıcaklık (°C, Derece)
Sin	Sinüs
m	Uzunluk (Metre)
mm	Uzunluk (m/1000)
cm	Uzunluk (Santimetre, m/100)
N	Yılın n. günü
%	Yüzde
θ_z	Zenit açısı (°)
TWh	Tera watt saat
kWh	Kilowatt saat
MHz	Mega hertz
Ω	Ohm

Kısaltmalar

PIC	Programlanabilir Mantıksal Denetleyici
GSYİH	Gayri safi yurt içi hasılat
CSP	Konsantre güneş enerjisi
PV	Fotovoltaik
DC	Doğru akım
NŞA	Normal şartlar altında
LDR	Işığa bağlı direnç
WIFI	Kablosuz Bağlantı Alanı
PWM	Sinyal Genişlik Modülasyonu
USB	Evrensel Seri Veri yolu
ICSP	Devre üzerinden seri programlama
CO_2	Karbon dioksit
Cz-Si	Monokristal Czochralski silikon

ŞEKİLLER LİSTESİ

Şekil 1.1 Güneş ışığının elektromanyetik dalga spektrumu (Steiner, 2008)..... 3
Şekil 1.2. Sabit bir gözlemcinin bir yıldız ile yaptığı açılar (Sparavigna, 2018).. 4
Şekil 1.3. Güneşin mevsimlere göre izlediği yol (Rüstemli vd. , 2013)..... 6
Şekil 1.4. Toplam insan nüfusunun son 300 yıldaki değişimi (Bavel, 2013)..... 6
Şekil 1.5. Vakum tüplü güneş kolektörleri (Yavuz, 2020)..... 8
Şekil 1.6. CSP kullanan bir güç santrali (Moussa, 2018)..... 9
Şekil 1.7. PV kullanan bir güç santrali (Turan, 2015)..... 9
Şekil 1.8. CSP güç santrallinin sistem elemanları (Praveen, 2020)..... 10
Şekil 3.1. Güneş Paneli..... 17
Şekil 3.2. Sistemin kurulum sonrası durumu..... 18
Şekil 3.3. Sistemin kurulum sonrası durumu..... 18
Şekil 3.4. Kullanılan metal profil çerçeveler..... 19
Şekil 3.5. Doğu batı dönüşünde kullanılan dişli sistemi..... 20
Şekil 3.6. Kuzey güney dönüşünde kullanılan dişli sistemi..... 20
Şekil 3.7. Dişli sisteminin Solidworks üzerinde gösterimi..... 21
Şekil 3.8. 6 V redüktörlü DC motor (Kumar, 2018)..... 22
Şekil 3.9. LM393 motor hız sensörü (Electronics, 2017)..... 23
Şekil 3.10. Optik disk..... 24
Şekil 3.11. L298 DC motor sürücü modülü (Rittenberry, 2005)..... 25
Şekil 3.12. Arduino Mega 2560 R3 pin şeması (Kusriyanto ve Putra, 2017)..... 26
Şekil 3.13. Arduino mikro hafıza kardı modülü (EBay, 2013)..... 27
Şekil 3.14. Mikro hafıza kardı modülünün katalog bilgileri (EBay, 2013)..... 27
Şekil 3.15. DS1302 (Maxim Integrated, 2015)..... 28
Şekil 3.16. Foto direnç (Sensors, 2008)..... 29
Şekil 3.17. Panel Güç ölçüm Sistemi..... 31
Şekil 4.1. 20 Temmuz deneysel verileri Gerilim grafiği..... 37
Şekil 4.2. 20 Temmuz deneysel verileri akım grafiği..... 37
Şekil 4.3. 20 Temmuz deneysel verileri güç grafiği..... 38
Şekil 4.4. 20 Temmuz hava sıcaklığı grafiği (Spark, 2023)..... 38
Şekil 4.5. 20 Temmuz rüzgâr hızı grafiği (Spark, 2023)..... 39
Şekil 4.6. 20 Temmuz güneşin yükseklik açısı grafiği (Spark, 2023)..... 39

Şekil 4.7. 16 Ağustos deneysel verileri gerilim grafiği 40
Şekil 4.8. 16 Ağustos deneysel verileri akım grafiği 41
Şekil 4.9. 16 Ağustos deneysel verileri güç grafiği 41
Şekil 4.10. 16 Ağustos hava sıcaklığı grafiği (Spark, 2023) 42
Şekil 4.11. 16 Ağustos rüzgâr hızı grafiği (Spark, 2023) 42
Şekil 4.12. 16 Ağustos güneşin yükseklik açısı grafiği (Spark, 2023) 43
Şekil 4.13. 6 Ocak deneysel verileri gerilim grafiği 44
Şekil 4.14. 6 Ocak deneysel verileri akım grafiği 44
Şekil 4.15. 6 Ocak 2024 deneysel verileri güç grafiği 45
Şekil 4.16. 6 Ocak hava sıcaklığı grafiği (Spark, 2023) 45
Şekil 4.17. 6 Ocak rüzgâr hızı grafiği (Spark, 2023) 46
Şekil 4.18. 6 Ocak güneşin yükseklik açısı grafiği (Spark, 2023) 46
Şekil 4.19. 6 Ocak bulutluluk oranı grafiği (Spark, 2023) 47
Şekil 4.20. 7 Ocak deneysel verileri Gerilim grafiği 48
Şekil 4.21. 7 Ocak deneysel verileri akım grafiği 48
Şekil 4.22. 7 Ocak deneysel verileri güç grafiği 49
Şekil 4.23. 7 Ocak hava sıcaklığı grafiği (Spark, 2023) 49
Şekil 4.24. 7 Ocak rüzgâr hızı grafiği (Spark, 2023) 50
Şekil 4.25. 7 Ocak güneşin yükseklik açısı grafiği (Spark, 2023) 50
Şekil 4.26. 7 Ocak bulutluluk oranı grafiği (Spark, 2023) 51
Şekil 4.27. Panellerin Toplam Enerji Üretimleri 54
Şekil 7.1. 20 Temmuz deneysel verileri akım grafiği 107
Şekil 7.2. 20 Temmuz deneysel verileri Gerilim grafiği 107
Şekil 7.3. 20 Temmuz deneysel verileri güç grafiği 107
Şekil 7.4. 16 Ağustos deneysel verileri gerilim grafiği 108
Şekil 7.5. 16 Ağustos deneysel verileri akım grafiği 108
Şekil 7.6. 16 Ağustos deneysel verileri güç grafiği 109

ÇİZELGELER LİSTESİ

Çizelge 3.1. Güneş panelinin teknik özellikleri 17
Çizelge 3.2. L298 DC motor sürücü modülü 25
Çizelge 4.1. 15-31 Temmuz tarihleri arasında yapılan ölçümler 36
Çizelge 4.2. 1-24 Ağustos arasında yapılan ölçümler 39
Çizelge 4.3. 6 Ocak arasında yapılan ölçümler 43
Çizelge 4.4.7 Ocak arasında yapılan ölçümler 47
Çizelge 6.1. 15-31 Temmuz tarihleri arasında yapılan ölçümler 103
Çizelge 6.2. 1-24 Ağustos arasında yapılan ölçümler 105

1. GİRİŞ

Dünya'nın temel enerji kaynağı olan güneş, insanlığa yüzyıllardır hizmet etmiştir. Günümüzde yaygın olarak kullanılan fosil yakıtların bile antik çağlardaki bitkilerin fosilleşmesiyle oluştuğu göze aldığında, insanlık açısından sınırsız bir enerji kaynağı potansiyeli olan güneş enerjisi, artan enerji ihtiyacını karşılamak için sürdürülebilir bir çözüm olarak görülmektedir.

Ülkelerin güneş enerjisi potansiyellerini etkileyen önemli faktörlerden biri ise üzerlerine düşen güneş radyasyonu miktarıdır ve ülkelerin Dünya üzerinde bulundukları konuma göre değişim gösterir. Avrupa kıtası açısından bakıldığında, Türkiye'nin bulunduğu konuma göre İspanya'dan sonra Avrupa'daki ikinci en yüksek güneş enerjisi potansiyeline sahip olduğu görülebilir (Solargis, 2021). Türkiye'nin diğer Avrupa ülkelerine oranla sahip olduğu bu avantaja rağmen, Türkiye'nin enerji ihtiyacının sadece %4 ünün (Gensed, 2021) güneş enerjisi ile karşılanmaktadır. Avrupa kıtasındaki Norveç ve İsviçre gibi ülkelerin Türkiye'ye oranla %40-50 daha az güneş enerjisi potansiyeline sahip olmalarına karşın bu ülkelerin enerji ihtiyaçlarının %60'ı güneş enerjisi ile karşılanmaktadır.

Türkiye'nin sahip olduğu güneş enerjisi potansiyelinin artan enerji ihtiyacını karşılamak için ithal edilen fosil yakıtların yerine geçebilecek bir kaynak olması yolunda son 10 yıl içerisinde önemli çalışmalar yapılmıştır. Ülkemizin gelişmekte olan ülkeler arasında olmasında gelişme hızını etkileyen önemli faktörlerden biri olan enerji, özellikle sanayi ve ziraat çalışmalarının ilerlemesini engellemektedir. Enerji üretiminin ithal kaynaklar ile elde edilmesiyle oluşan maliyet artışları ve fosil yakıt rezervlerinin yetersizliği, Türkiye'nin güneş enerjisi üretimi açısından yapmış olduğu yatırımları hızlandırmasına yol açmıştır.

Bu durum Dünya genelinde incelendiğinde, ülkelerin yıllık tükettikleri kilowatt-saat (kWh/yıl) oranları ile Gayri safi yurt içi hasılatlarının (GSYİH) arasındaki bağ rahatlıkla görülebilir (Adom, 2011). Fakat elde edilen ekonomik kazancın verimli bir şekilde ülke çıkarında kullanılabilmesi için üretimde kullanılan enerjinin yerli kaynaklardan üretilmesi gerekir. Fakat Türkiye gibi yeterli fosil yakıt rezervi bulunmayan ülkeler ya enerji ihtiyacını dışa bağımlı hale getirmekte ya da yenilenebilir enerji kaynaklarına yönelmektedir (Bayraç, 2009) .

Ülkemizin bulunduğu konum açısından güneş enerjisi potansiyelinin Avrupa kıtasındaki diğer ülkelere oranla fazla olmasına karşın, Türkiye'nin kuzey yarım küredeki orta kuşak bölgesinde bulunması sebebiyle güneş ışınlarının geliş açıları gün içerisinde ve yıl boyunca sürekli değişmektedir. Güneş ışınlarının geliş açılarındaki bu değişim, özellikle güneş enerjisinden yararlanmak için yaygın olarak kullanılan fotovoltaik güneş panellerinin verimlerini önemli ölçüde etkilemektedir. Bu durumun çözülmesine yönelik kullanılan yöntemlerden biri olan güneş takip sistemleri, fotovoltaik güneş panellerinin güneş ışınlarının geliş açılarına göre ayarlanmasını sağlayarak, güneş ışınlarının panel yüzeyine dik açıda gelmesini sağlamakta ve fotovoltaik güneş panellerinin verimlerini önemli ölçüde etkilemektedir.

Güneş takip sistemlerinin fotovoltaik güneş panellerinin verimlerini artırmasını etkileyen faktörlerin değerlendirilmesi özellikle potansiyel yatırımcılar açısından sorun oluşturulmaktadır. Bu faktörler arasında kullanılacak olan güneş takip sistemlerinin çeşitleri, güneşi tek veya çift eksende takip eden olarak ayrılmaktadır. Buna ilaven gün içerisinde güneş ışınlarının geliş açılarındaki değişimin miktarına bağlı olarak güneş takip sistemini çalışma aralıkları ve panellerin açı değiştirmesiyle oluşan gölgelerden dolayı, güneş takip sistemiyle elde edilebilecek verim artışlarını düşürebilmektedir. Ayrıca güneş takip sistemlerinde güneş panellerinin hareket ettirilmesi için gereken ekipmanların sistem maliyetini artırması, potansiyel yatırımcıların güneş takip sistemlerini tercih etmemelerine neden olmaktadır.

Yapacağım bu çalışmada Konya ilinde bulunan sabit bir fotovoltaik güneş panelinde çeşitli güneş takip sistemleri kullanılmasıyla elde edilen verim artışını ve güneş takip sisteminin geri ödeme süresine olan etkilerini araştırarak, gelecekteki potansiyel yatırımcılara yol gösterici bir kaynak olacağı düşünülmektedir.

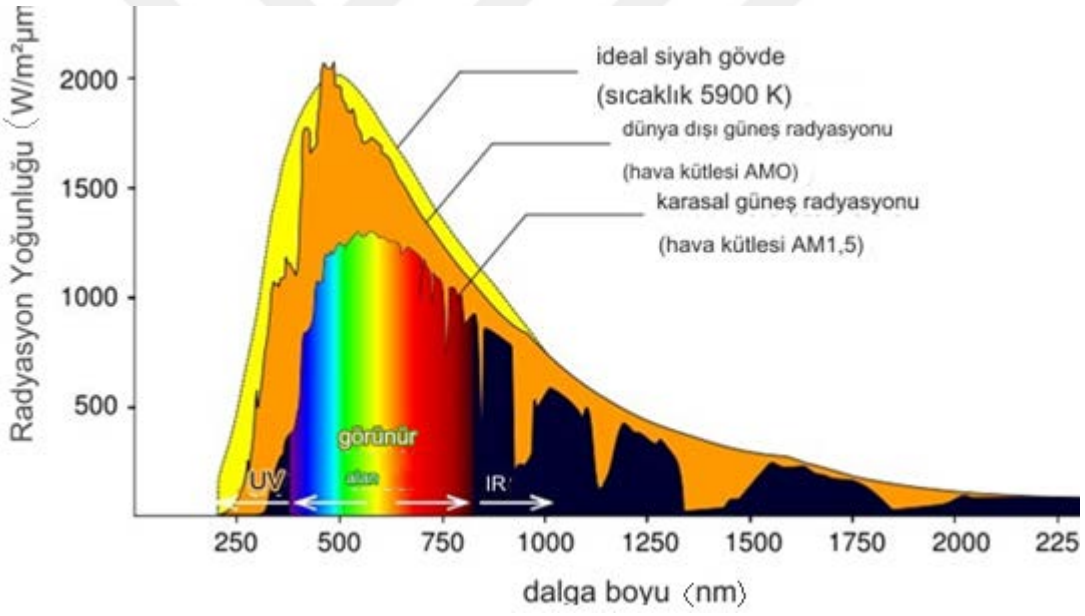
1.1. Güneş

Güneş sisteminin merkezinde bulunan güneş, tek başına güneş sisteminin toplam kütlesinin %99.8'ni oluşturmaktadır (Dennis, 2000). Güneşin yüzey sıcaklığı 5500 °C dir ve sahip olduğu bu sıcaklıktan kaynaklanan siyah cisim ışımasıyla Dünya'ya ulaşan ışınlar bitkiler tarafından fotosentez yoluyla enerjiye dönüştürülür ve bu sayede Dünya'daki hayatın devamlılığında kritik rol oynar.

Dünya'nın birincil enerji ihtiyacı yaklaşık 107,000 TWh/Yıl'dır. Güneşten Dünya'ya gelen enerji ise bu değerin 20 katına eşittir ve bu enerji güneşin yaydığı toplam enerjinin 20 milyonda birine eşittir. Güneşin sahip olduğu bu enerji potansiyeli insanlığın sürdürülebilir enerji kaynağı arayışına çözüm olacağı düşünülmektedir.

1.1.1. Güneş ışığının özellikleri

Her gün gördüğümüz ışık güneşten yayılıp yer yüzeyine gelen toplam enerjinin yalnızca bir bölümüdür. Güneşten Dünya'nın atmosferine ulaşan ışınların bir kısmı atmosferden yansıyarak uzaya geri döner. Geri kalan ışınların büyük bir kısmı görünür ışık spektrumunda olmasına karşın belli bir kısmı mor ötesi ve kızıl ötesi ışınlardan oluşmaktadır. Şekil 1.1'de Dünya'ya ulaşan güneş ışığının elektromanyetik dalga spektrumu verilmiştir.



Şekil 1.1 Güneş ışığının elektromanyetik dalga spektrumu (Steiner, 2008)

Güneşten gelen ışınların büyük bir kısmı yeryüzüne ulaşırken yansıma ve atmosferik gazlar gibi sebeplerden yeryüzüne ulaşamaz. Bu ışınların %30'u atmosferden, bulutlardan ve yeryüzünden yansır ve uzaya geri döner. Güneş ışınların %19'u ise atmosferik gazlar ve bulutlar tarafından emilir. Güneş ışınlarının %26'sı ise atmosferden geçerken dağılmaya uğrar ve yayılım şeklinde yeryüzüne ulaşır. Yani güneşten gelen enerjinin sadece %51'i yeryüzüne ulaşabilmektedir.

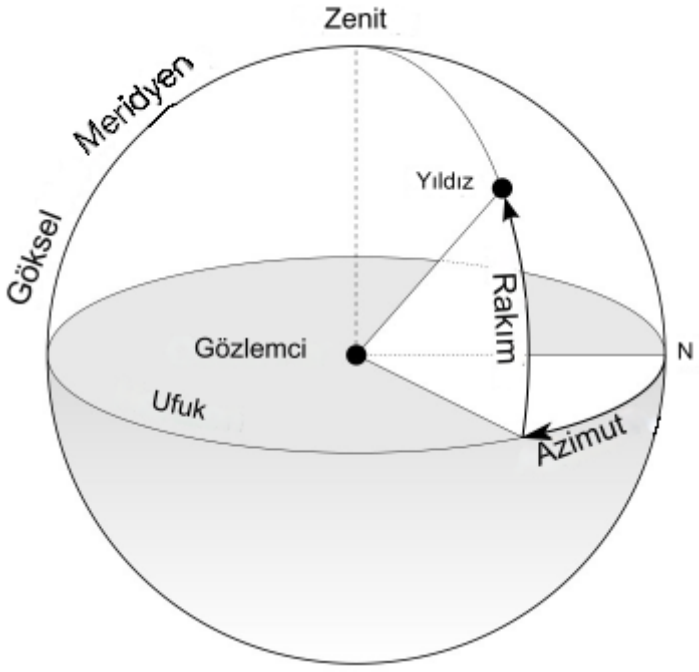
1.1.2. Güneş ışınlarının geliş açısını etkileyen faktörler

1.1.2.1. Gün içerisinde güneşin doğu-batı hareketi

Dünya kendi ekseni etrafında 1 tam tur döndüğünde güneş ışınları doğudan batıya doğru 360 derecelik bir döngü yapar. Dünya üzerinde sabit bir noktadan izlendiğinde ortalama 12 saat içerisinde güneş ışınlarının 180 derecelik bir değişim izlediği görülebilir fakat yerel rakım ve ufuk görünürlüğünün etkisinden ortalama 150 derecelik bir değişimin gözlenebilir olduğu söylenebilir. Sabit açılı bir fotovoltaik panel için gün doğumu ve günbatımı sırasında oluşan 75 derecelik açı farkı, fotovoltaik panelin verimini %75 'in üzerine azaltır.

Fotovoltaik panelin 1 günde 12 saat güneşlenme süresi olduğu ve gün ortasında panele güneş ışınlarının dik geldiği düşünülürse sabit bir fotovoltaik panelin gün içerisindeki ortalama verimi %35-45 arasında olur. Bu verim kaybını engellemek amacıyla tek eksenli güneş takip sistemleri kullanılır.

Şekil 1.2'te Yıldız ve gözlemci arasındaki yükseklik açısı gösterilmiştir. Yükseklik açısı gün doğumu sırasında 0 derecede olduğu ve gün içerisinde gün batımına kadar doğudan batı yönünde 180 derece bir açı izlediği gösterilmiştir.



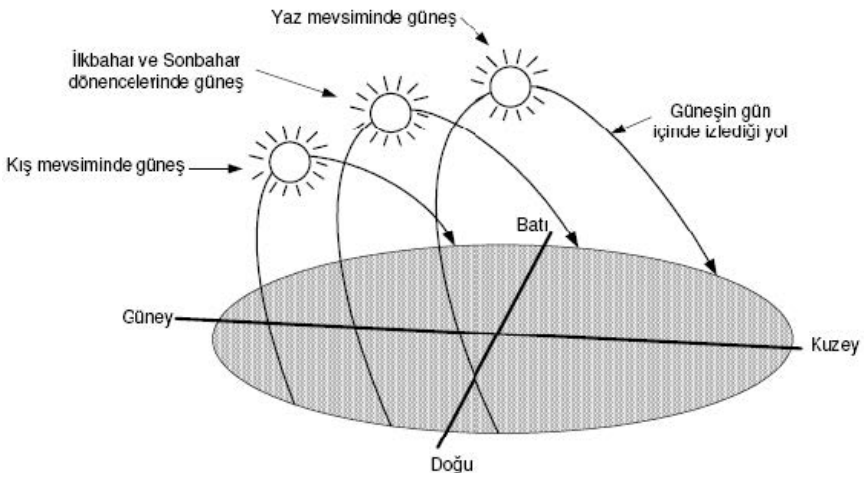
Şekil 1.2. Sabit bir gözlemcinin bir yıldız ile yaptığı açılar (Sparavigna, 2018)

1.1.2.2. Güneşin mevsimsel kuzey-güney hareketi

Dünyanın kuzey ve güney kutupları arasındaki 23.44 derecelik açı sapması nedeniyle, Güneş bir yıl içerisinde kuzeyden güneye 46.88 derecelik bir açı değiştirir. Dünya üzerinde sabit bir noktada bulunan gözlemcinin Güneş ile arasında bulunan dikey açı farkıdır ve sapma açısı olarak adlandırılır.

Ekvatorda bulunan sabit bir fotovoltaik panel için 20 Mart tarihindeki ilkbahar ekinoksu sırasında sapma açısı 0 derece olur. 20 Mart ilkbahar ekinoksu sonrasında 20-21 Haziran tarihindeki gündönümüne kadar sapma açısı kuzeye doğru artış gösterir ve 20-21 Haziran tarihindeki gündönümü sırasında sapma açısı 23,44 dereceye ulaşarak maksimum değere varır. 21 Haziran tarihindeki gündönümü sonrasında sapma açısı azalmaya başlar ve 22-23 Eylül tarihindeki sonbahar ekinoksuna kadar azalmaya devam eder. 22-23 Eylül tarihindeki sonbahar ekinoksunda sapma açısı tekrar 0 derece olur. 22-23 Eylül tarihindeki sonbahar ekinoksundan sonra sapma açısı güneye doğru artış gösterir ve bu artış 21-22 Aralık tarihindeki gündönümüne kadar devam eder. 21-22 Aralık tarihindeki gündönümü sırasında sapma açısı kuzey kutbu doğrultusunda minimum değer olan-23.44 derece olur. 21-22 Aralık tarihindeki gündönümü sonrasında sapma açısı kuzeye doğru bir artış gösterir ve tekrar 20 Mart tarihindeki ilkbahar ekinoksuna kadar artmaya devam ederek 0 derece olur.

Dünyadaki bu eksen eğikliği aynı zamanda kuzey ve güney yarım kürenin güneşlenme süresini etkilemesi sonucunda mevsimler meydana gelir. Kuzey yarım küre için 0 derece enlemindeki ekvator ile 30 derece kuzey enlemi arasında 20 Mart tarihindeki ilkbahar ekinoksu ile 22-23 Eylül tarihindeki sonbahar ekinoksu arasında güneşlenme süresi ortalama günde 12-14 saat olur ve yaz mevsimi yaşanır. Aynı zaman aralıkları sırasında güney yarım küre için 0 derece enlemindeki ekvator ile 30 derece güney enlemi arasında güneşlenme süresi ortalama 10-12 saat olur ve kış mevsimi yaşanır. Ayrıca bu olaya etki eden tek faktör güneşlenme süresi değildir. İlkbahar ekinoksu ile sonbahar ekinoksu arasında güney yarım küredeki bölgeler aynı bölgenin ekvatora göre simetrik enleminde bulunan kuzey yarım küredeki bölgelere oranla güneş ışınlarındaki sapma açısı 23,44 derece fazla olur. Güneş ışınlarının sapma açısındaki bu artış güneşlenme süresini değiştirdiği gibi güneş ışınlarının yeryüzüne çarptıktan sonra yansıma sonucu uzaya geri dönmesine sebep olur. Şekil 1.3'te güneşin mevsimlere göre izlediği yol verilmiştir.

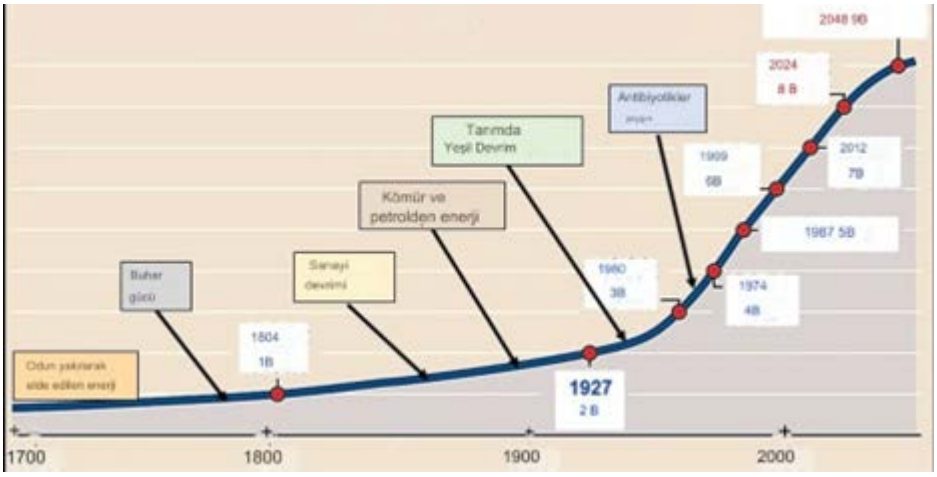


Şekil 1.3. Güneşin mevsimlere göre izlediği yol (Rüstemli vd., 2013)

1.2. Enerjinin İnsanlık Üzerindeki Etkisi

İnsanlar tarih boyunca hayatta kalmak için enerjiyi kullanmıştır. Tarih öncesi kabilelere bakıldığında ateşin keşfedilmesiyle insanlık soğuk iklim şartlarında hayatta kalabilmiş ve bu sayede daha geniş bir coğrafyaya yayılmıştır. 1820-1840 yılları arasındaki sanayi devrimiyle insanların buhar ile çalışabilen makineler üretmeye başlamasıyla insan gücünün yerini almış ve sanayileşmenin başlamasını sağlamıştır. Dünya çapında sanayinin gelişmesi ile artan üretim ve beraberindeki refah artışı sonucunda insan nüfusunda önemli bir artış yaşanmıştır.

Şekil 1.4'teki grafikte görüldüğü gibi, sanayi devriminin etkisiyle toplam insan nüfusunda belirgin bir artış yaşanmıştır. Devamında petrolün enerji kaynağı kullanmaya başlaması ile insan nüfusundaki artışta hızlanmıştır.



Şekil 1.4. Toplam insan nüfusunun son 300 yıldaki değişimi (Bavel, 2013)

Fosil yakıtların kullanılmaya başlanmasıyla artan refah düzeyi ve beraberinde yaşanan teknolojik gelişmeler ile insanlığın enerjiye olan ihtiyacı da önemli ölçüde artmıştır. Artan enerji ihtiyacının karşılanmasında yoğun olarak kullanılan fosil yakıtların ürettiği CO_2 gibi sera gazlarının atmosferde birikmesi sonucunda küresel ısınma ve asit yağmurları gibi yaşanan olaylar günümüzde daha belirgin olarak görülmektedir.

Artan enerji ihtiyacını karşılamak için tüketilen fosil yakıtlar sonucu karbon emisyonunda artma görülmüştür. Bu durumu engellemek için birçok ulusun katıldığı Kyoto anlaşması imzalanmıştır (Gerden, 2018). Bu anlaşma ile beraber birçok ülke karbon emisyonları azaltmak için güneş, rüzgâr veya jeotermal gibi yenilebilir enerji kaynaklarına yönelmiştir (Hellmich ve Kiesel, 2021).

1.3. Güneş Enerjisinin Kullanım Alanları

Güneş enerjisinden yararlanma konusundaki çalışmalar özellikle 1970'lerden sonra hız kazanmış, güneş enerjisi sistemleri teknolojik olarak ilerleme ve maliyet bakımından düşme göstermiş, güneş enerjisi çevresel olarak temiz bir enerji kaynağı olarak kendini kabul ettirmiştir (Kürklü, 2017).

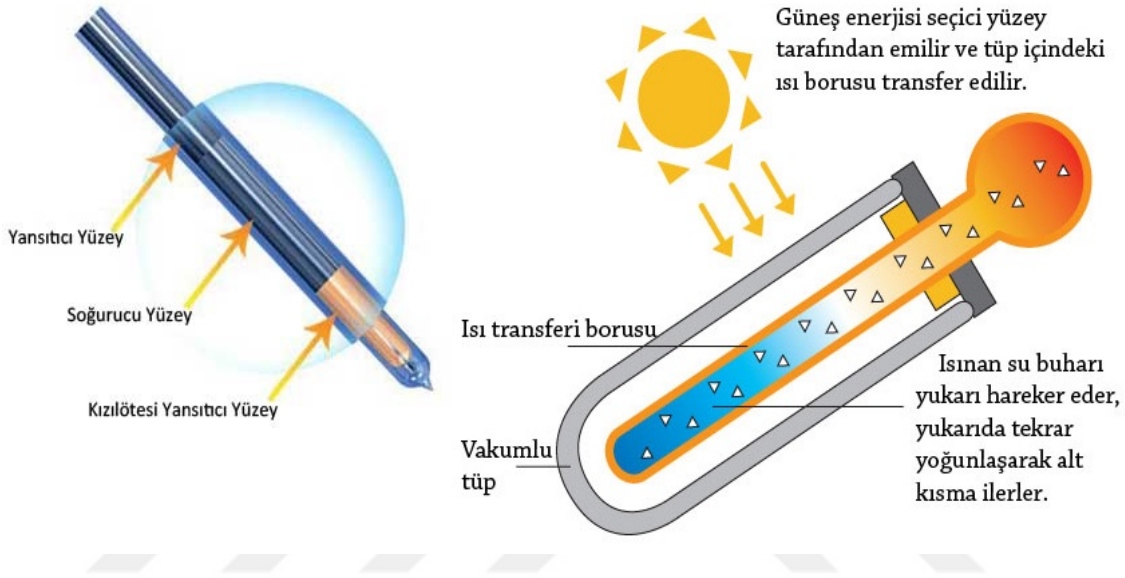
Güneş enerjisi temel olarak iki amaç için kullanılabilmektedir. Bu amaçlardan ilki ısı, ikincisi ise elektrik üretimidir. Güneş enerjisinin hem ısı hem de elektrik üretimi için kullanıldığını kombine sistemlerde verimlilik diğer sistemlere oranla daha yüksektir.

1.3.1. Güneş enerjisinin ısıl uygulamaları

Güneş enerjisinden yararlanmanın çok sayıda yöntemi olmasına rağmen, bu yöntemler içerisinde en olgunlaşmış ve yaygın olarak kullanılanı ısıl uygulamalarıdır. Bu sistemler en temel anlamda güneş enerjisini ısıl enerjiye dönüştürürler. Elde edilen ısı enerjisi; ısıtma ve soğutma uygulamaları, su arıtma, buhar üretimi, elektrik üretimi, endüstriyel kurutma, pişirme vb. çok sayıda farklı şekilde kullanılabilmektedir. Güneş enerjisinin ısıl uygulamaları üç temel sistem elemandan oluşmaktadır. Bunlar; güneş kolektörü, ısı değiştiricisi ve depolama ünitesidir. Bu sistem ekipmanlarının birlikte olduğu sistemlerde bulunmaktadır. Ancak bu üç sistem elemanın işlevini yerine getirecek elemanların biri ya da tamamı birlikte de bulunabilmektedir. Güneş enerjisinin ısıl uygulamalarında elde edilen ısı enerjisi doğrudan kullanılabileceği gibi başka sistemlere

entegre edilebilmektedir. Bu sistemler en temelde güneş enerjisini ısı enerjisine dönüştürüp bir ısı transferi akışkanına aktarırlar. Sıcaklığı yükselen veya faz değiştiren ısı transferi akışkanı direkt olarak depolanıp veya kullanılabileceği gibi bir ısı değiştirici aracılığı ile başka bir ortama da aktarılabilir veya depolanabilir.

Bu işlemler yapılan uygulamalara göre farklılık göstermektedir. Şekil 1.5'de konutların ısıtılmasında yaygın kullanılan vakum tüplü bir güneş paneli gösterilmiştir.



Şekil 1.5. Vakum tüplü güneş kolektörleri (Yavuz, 2020)

1.4. Güneş Enerjisinden Elektrik Üretilmesi

Güneş enerjisinden elektrik üretiminde başlıca iki yöntem uygulanmaktadır. Dolaylı yöntemde güneş enerjisinin yoğunlaştırıcı sistemler kullanılarak odaklanması sonucunda üretilen kızgın buhardan, bilinen yöntemlerle elektrik üretilir. Güneş ışınlarının odaklanması ile çalışan konsantre güneş enerjisi (CSP) ile çalışan elektrik santrali Şekil 1.6'de verilmiştir. Doğrudan yöntemde ise fotovoltaik ve termoelektrik çeviriciler yer alır (Öztürk, 2004). Şekil 1.7'da fotovoltaik (PV) kullanan bir güç santrali gösterilmiştir.

Şekil 1.6. CSP kullanan bir güç santrali (Moussa, 2018)

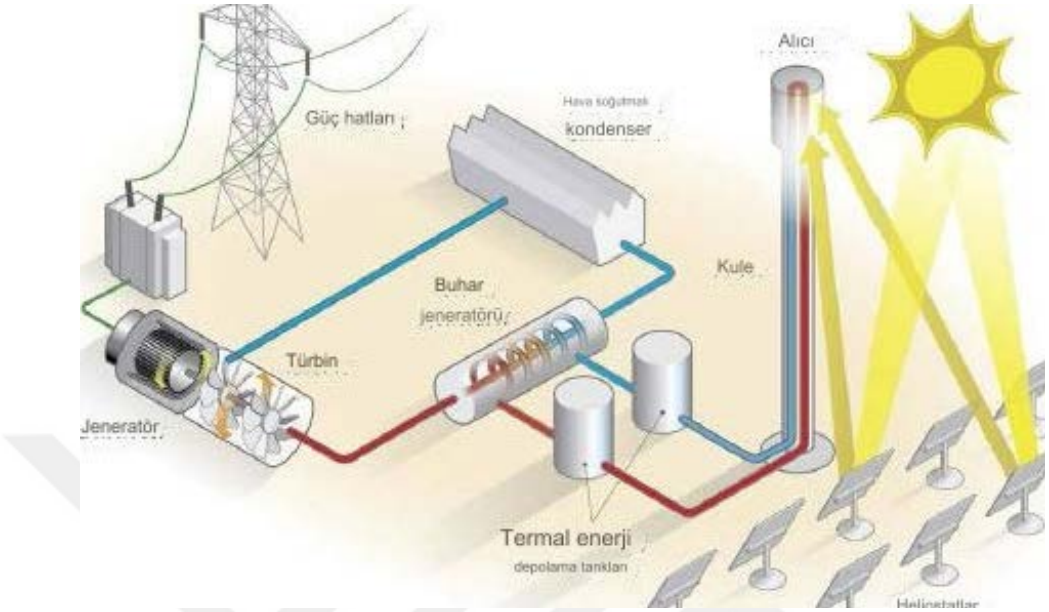
Şekil 1.7. PV kullanan bir güç santrali (Turan, 2015)

1.4.1. CSP güç santralleri

CSP güç santralleri PV santrallerinden farklı olarak güneş ışınlarını direkt elektriğe dönüştürmek yerine ışığın içerisindeki termodinamik enerjiyi, türbin vasıtası ile elektriğe dönüştürme prensibiyle çalışır.

CSP güç santrallerinin PV güç santrallerine oranla en büyük avantajı gün içerisinde kazanılan termodinamik enerjinin depolanabilmesi ve ihtiyaç anında kullanılabilmesidir. CSP güç santrallerinin dezavantajları ise güneş enerjisini doğrudan çevirmek yerine dolaylı yollarla elektriğe çevirmesinin sonucunda çok büyük bir yatırım ve alana ihtiyaç duyulmasına sebep olurlar, bu nedenle de konut veya küçük çaplı projeler için uygun değillerdir (Balghouthi vd., 2016). Şekil 1.8'da CSP sistemiyle çalışan bir güç santrallerinin elemanları gösterilmiştir. Şekil 1.8'da görüldüğü üzere santral çeşitli

elemanların güneşten gelen enerjiyi termodinamik ısı enerjisine dönüştürmesi ve bu enerjinin türbin & jeneratör ile elektrik enerjisine çevirmesi prensibinde çalışmaktadır.



Şekil 1.8. CSP güç santrallinin sistem elemanları (Praveen, 2020)

1.4.2. PV güç santralleri

PV güç santralleri güneş ışınlarını yarı iletken malzemeler kullanarak doğrudan elektrik enerjisine çevirmesi prensibiyle çalışır. Çalışma prensipleri güneş ışınları (fotonlar) fotovoltaik hücreye çarptıklarında, hücrenin içinde bulunan yarı iletken malzeme tarafından absörbe edilirler. Fotondan absörbe edilen bu enerji yarı iletken silikon malzemedeki elektronları uyartılmış hale getirir, uyarılan negatif yüklü elektronlar kazandıkları ekstra enerji sayesinde bağlı oldukları atomlardan ayrılabilirler. Sistemin bu noktasında önemli bir husus yarı iletken malzemenin özel dizaynının serbest haldeki elektronları belirli ve tek bir yönde hareket etmeye kısıtlamasıdır. Bunun sağlanması için genellikle yarı iletken silikonun içerisine boron veya fosfor ilave edilir ve bu işleme silikonun doping edilmesi adı verilir.

1.4.3. PV güç santrallerini etkileyen faktörler

PV güç santrallerini etkileyen faktörler 2 gruba ayrılırsa bunlar hücre verimi ve foton verimi olabilir. Hücre verimini etkileyen faktörler ana başlıklar altında: uygun

saydam iletken seçimi, arka yüzey pasifleştirilmesi, yansıma engelleyici kaplamalar ve yapılar, hücre içi soğutma ve koruyucu kaplamalardır. Hücre verimi alanındaki bütün etken ve geliştirmeler hücrenin yapısı ve elektriksel karakteristiği hakkındadır.

1.4.3.1. Uygun saydam iletken seçimi

Uygun saydam iletken seçimi, hücrenin ışık alan yüzeyindeki elektron akışını aktaracak iletkenlerin saydamlığı ve elektriksel dirençleridir. Kaybı en aza indirmek için genellikle geçirgenliği yüksek ve elektrik iletkenliği yüksek olan indiyum kalay oksit filmleri kullanılır. Bununla beraber alternatif olarak iletken polimerler veya iletken nano kablolar da kullanılabilir.

Burada önemli bir husus vardır, malzemenin sıklığını (ya da yoğunluğunu) artırmak elektrik iletkenliğinin artmasını sağlar fakat bununla beraber artan yoğunluk, malzemenin geçirebileceği foton aralığını kısıtlar. En yüksek verimi elde etmek için nano kablo hatları oluşturmak veya iletken ağ oluşturmak yeterlidir (Sadeque, 2020).

1.4.3.2. Arka yüzey pasifleştirilmesi

Arka yüzey pasifleştirilmesi (pasivasyon), yüzey pasivasyonu fotovoltaik hücrelerin veriminde kritik önem arz etmektedir. Hücrenin ön yüzeyinde birçok geliştirmeler yapıldığı halde arka yüzey geliştirmeleri hala araştırma aşamasındadır (Bajpai ve Kumar, 2011). Pasivasyon işlemi temel anlamda, bir malzemeyi korozyona karşı korumak için korunacak materyal ile oksitleyici etkenden arasına bir koruyucu tabaka konulması esasına dayanır. Fotovoltaik hücrelerde oksidasyonun istenmemesinin ana sebebi oldukça hassas bir incelikteki p-n bağlantısının elektriksel özelliklerini etkilemesi ve panelin çıkışındaki akım ve gerilimi değiştirmesidir.

Dr. Lachlan E. Black'ın 2015-2018 yıllarında yaptığı çalışmalar (Marliyani binti Omar, 2009) sonucunda hücrenin arka yüzeyinin elektriksel izolasyonu, ince silika veya alüminyum oksit filminin üzerine silikon nitrat kaplanmış ve bu sayede verim artırışı sağlanmıştır. Bu çalışma sayesinde mono kristal Czochralski silikon (Cz-Si) materyallerin verimi %17 den %21 e çıkarılmıştır (Ahmed, Hasan ve Majumber, 2010).

1.4.3.3. Yansıma engelleyici kaplamalar ve yapılar

Yansıtma engelleyici sistemler temel olarak gelen fotonların hücreye çarptıktan sonra dalga boyları veya hücre kusurları sebebiyle geri yansımalarını engellemek üzere kullanılırlar. Bu konuda kullanılan kaplamalar fotonların hücrenin yüzeyine girdikten sonra geri çıkmalarını engelleyerek gelen tüm güneş ışınını fotovoltaik 'e aktararak verimi artırabilirler fakat bu kaplamalar ciddi bir risk arz etmektedir. Fotonlar yarı iletken malzemeye çarptıklarında malzemedeki elektronları yörüngelerinden çıkartarak, elektronların malzemede bulunan elektron boşluklarında ("hole") içerisinde hareketini sağlarlar. Normal şartlar altında (NŞA) eğer gelen fotonun dalga boyu uzun ise elektronları uyarılmış hale gelmeleri için gereken seviyede enerji sağlayamaz ve malzemenin içerisinden pasif bir biçimde geçerler. Eğer gelen fotonun dalga boyu panelin kullanabileceği aralıktan kısa ise (enerjisi yüksek ışınlar) NŞA bir hücre için gelen foton hasar bırakmadan geri yansır. Fakat eğer hücrede yansıtma engelleyici bir kaplama bulunuyorsa bu durumda bu fotonlar geri yansıyamayabilir ve enerjileri hücre tarafından kullanılamadığı içinde hücreye ciddi bir hasar verebilir (Chowdhury ve Rahman, 2017).

Hücre yapısını değiştirerek yansıma engelleme, hücrenin üst tabakasını çeşitli yöntemlerle değiştirerek yansıyan fotonların tekrardan hücre içine dönmesini sağlama işlemidir. Bu yöntem aynı zamanda yarı iletken tarafından kısmi absörbe edilmiş fotonların kalan enerjilerini kullanarak, hücre verimini artırabilir.

2. KAYNAK ARAŞTIRMASI

(C. I. Yousif, 2002) örneklendirdiği mekatronik izleme sistemini kontrol eden AT mega mikrodenetleyici, güneş paneli montajına ayarlanan dört sensöre dayanmaktadır. Çift eksenli güneş takip paneli, sabit güneş paneline göre yaklaşık %31.5 daha yüksek çıkış gücü elde etmiştir.

(Armakan, 2003) teorik olarak sabit bir güneş pili ve çift eksenli hareket eden güneş pili arasındaki enerji kazanımını inceleyerek karşılaştırmış ve çift eksenli solar takip sisteminin yaklaşık %40 daha fazla enerji ürettiğini belirtmiştir.

(Abdallah, 2004) yaptığı çalışmada dört farklı mekanik yapıya sahip güneş takip sistemlerinin enerji verimliliklerini deneysel olarak karşılaştırarak, çift eksenli takip sisteminin %43.87, tek eksenli dikey takibin %37.53, tek eksenli doğu-batı takibin

%34.43 ve tek eksenli kuzey-güney takibin ise %15.69 daha verimli çalıştığını tespit etmiştir.

(Mehmet, 2006), iki sabit ve takip mekanizmalı sistem kurarak, bu sistemlerin verilerini kontrol edebilmek amacıyla mikrodenetleyici kontrollü bir arabirim oluşturmuştur. Günün farklı saatlerinde üretilen gerilimin; güneşin doğduğu ilk saatlerde az, öğlen saatlerinde en yüksek ve gün batımında ise yine azaldığı için gerilim değerinin de azaldığı belirtilmiştir. Araştırma sonucunda güneş takip sisteminin, sabit sisteme göre %35 oranında daha verimli olduğu ispatlanmıştır.

(Yuksekkaya vd. , 2006) önerdiği akıllı ev otomasyon sistemi, GSM teknolojisi ve internet üzerinden konuşma tanımadan yararlanmaktadır. Kablosuz ev otomasyon teknolojisini sorunsuz bir şekilde entegre ederek esneklik, maliyet etkinliği ve verimli uzaktan izleme ve kontrol sunmaktadır.

(Bilgin, 2006) tez çalışması, sensör tabanlı bir güneş takip sistemi tasarlayarak, güneş ışınlarının fotovoltaik (PV) panele dik açı ile gelmesini sağlamıştır. Çalışmada, PV hücrelerin kimyasal yapısının yanı sıra güneş ışınımının panel ile yaptığı açının önemi vurgulanmış ve tek eksenli takip sistemi ile sabit sisteme kıyasla %31 civarında, çift eksenli takip sistemi ile %37 civarında bir verim artışı gözlemlenmiştir.

(Uzunok, 2007) İskenderun'da yaptığı araştırmada farklı markalarda güneş pillerinin verim açısından değerlendirilmesine yönelik olarak, tasarladığı iki eksenli güneş takip sisteminde %17.07'lik bir verim artışı gördü ve ayrıca bir maliyet analiz çalışması yaptı.

(Çalışkan ve Öztürk, 2008), çeşitli güneş takip sistemlerinin çalışma prensiplerini, çalışma verimlerini, birbirlerine göre faydaları ve sakıncalarını ve diğer özellikleri araştırdılar. Fotovoltaik panel yüzeyine düşen ışınım şiddeti ve üretilen güç, mevsimlere göre değişen güneşin geliş açısı nedeniyle önemli ölçüde değişmiştir. Ayda bir kez panelin eğim açısı değiştirildiğinde, aynı fotovoltaik panelden daha fazla güç elde edilebilir.

(Kansal, 2008) çalışmasında, PIC 16F877A mikrodenetleyici ile kontrol edilen bir step motor kullanan tek eksenli güneş takip sistemi tanıtılmıştır. Fotovoltaik modül, güneş

ışınımı yoğunluğunu tespit etmek için bir sensör görevi görmüştür. Sonuçlar, güneş takip sistemlerinin sabit panellere kıyasla yaklaşık %8 daha fazla enerji toplayabildiğini göstermiştir.

(Ghassoul, 2009) çalışmasında, Siemens mikro PIC 18F452 kullanılarak tasarlanan tek yönlü bir güneş izleme cihazı öne çıkarılmıştır. Bu sistem, biri Güneş'e doğru maksimum eğim açısını tespit eden, diğeri ise hücreleri buna göre yönlendiren foto dirençlerini içermektedir. Ancak, çalışmanın sadece kavramsal aşamada olduğu ve tasarımın ötesinde test edilmediği önemle belirtilmiştir.

(Gill vd., 2009) çalışmasında, gömülü teknolojiye dayalı bir kablosuz ev ağ geçidi geliştirilmiştir. Bu ağ geçidi, ZigBee ile Wifi arasında verileri dönüştürerek akıllı ev otomasyonunda istikrarlı bir performans sağlamıştır.

(Barsoum, 2011) tanımladığı güneş takip cihazı, dört LDR kullanarak güneş radyasyonu değişimlerini tespit etmektedir. Bu sistem, bir PIC 16F84A mikrodenetleyicisi tarafından kontrol edilmektedir.

(Beyoğlu, 2011) Balıkesir İl'inde teşvik amaçlı güneş enerjisi potansiyeli ve maksimum güç takip sistemine sahip sabit ve çift eksenli iki fotovoltaik (PV) sisteminin kurulmasını inceledi. Eş zamanlı çalışmalar, çift eksenli sistemlerin sabit sistemlere göre %39 daha verimli olduğunu gösterdi. Bu çalışma, sistemi boyutlandırmak için 85 W güneş paneli ve 40 Ah akü kullanarak maliyeti ve kayıpları azaltmayı göz önünde bulundurdu. Birkaç gün boyunca yağan ve bulutlu hava sistemin yeterince şarj olmasını engellemiş ve bunun sonucunda akü de tükenmiştir. Yeni sistem kurulduğunda hava şartlarının uzun süre bulutlu ve kapalı olduğu yerlerde kayıpların artacağı göz önünde bulundurularak, akü ve panelin yüksek güçte seçilmesi önerilir.

(Arsalan, 2013) çalışmasında ise, iki foto direnç içeren 8051 mikrodenetleyiciye dayanan tek eksenli bir güneş izleme cihazı gösterilmiştir. Takip sistemi, hem otomatik hem de manuel olarak çalışmaktadır.

(Bangali ve Shaligram, 2013) çalışması, cep telefonu kısa mesajlarının akıllı ev sistemlerinde GSM üzerinden nasıl uygulanabileceğini araştırmıştır. PIC 18F4520 ve sim 548c donanım ve yazılım yapılandırmalarını birleştirerek AT komutlarını kullanarak sistemin mimarisini derinlemesine incelemiştir.

(Piyare, 2013), ev otomasyon sistemini kontrol etmek için IP adresi aracılığıyla internet tabanlı bir yaklaşım benimsemiştir. Bu yöntem, bir Android uygulaması ve bir mikro web sunucusu kullanarak cihaz kontrolünü ve izlenmesini mümkün kılmıştır; bu da programın maliyet etkinliği ve esneklik gibi avantajlarını vurgulamıştır.

Bu lisans tez çalışmasında, literatürde yer alan çalışmaların ayrıntılı olarak gözden geçirilmesinin ardından özgün bir güneş takip sistemi geliştirilmiştir. İki eksenli ve Arduino kontrollü olan güneş takip sistemi tasarlanmıştır. Sistemi tasarlarken daha önceden yapılmış birçok projeyi incelenmiştir. Bu veriler ışığında sistemin nasıl daha verimli ve daha uygun olduğuna karar verilmiştir (Aydın ve Kobya, 2015).

(Başak ve Şıran, 2017) Yapılan bu proje ile en çok kullanılan sistem modellerinden transfer fonksiyonu veya durum uzayı modeli girilerek sistemlerin analizi kolaylıkla yapılabilmektedir. Kullanıcıların herhangi bir MATLAB bilgisine sahip olmadan MATLAB çalıştırarak sistemlerin kontrol sinyalli uygulanmadan zaman ve frekans cevap eğrilerine erişilebilmektedir. Yapılan deneylerde MATLAB ile elde edilen sonuçlar teorik olarak hesaplanan sonuçlar ile örtüşmektedir. Modeli bilinen sistemlerin PID ile kontrol edilmesinde MATLAB yardımı ile deneysel simülasyon yapılması sağlanmıştır. Bu sayede sisteme herhangi bir hasar verilmeden istenilen değere ulaşılıp ulaşılmadığı kontrol edilebilir. Simülasyon işlevinin yanı sıra, PID kontrolörün yapısı ve PID parametrelerinin ayrı girilmesi ile sistemlerin zaman ve frekans cevap eğrilerindeki değişmeler gözlemlenmektedir. Bu sayede endüstride çok geniş bir kullanım yeri olan PID kavramının öğrenilmesi kolaylaştırılmıştır. P, I ve D parametrelerindeki artış ve azalışlar ile sistemin tepkisi gözlemlenebilmektedir.

(Musa, 2017), güneş takip sistemlerinde hız kontrolünü kolaylaştırmak ve istenilen devirlerde motorları çalıştırmak için fırçalı sabit mıknatıslı doğru akım motoru kullandı. Damperli motor, DC motora bağlı olarak tasarlanmıştır ve on beş dişli sistemi

içerir. Bu dişli veya redüktör yardımıyla damperli motorun momenti 16 Nm'ye yükseltilmiştir. Damperli motorlar, doğru akımda çalıştıkları için değişken yükler altında hızlarını kolayca ayarlayabilirler. Bu motorlar, tasarladığı güneş takip sisteminin her iki ekseni için kullanılmıştır. Sistemdeki panellerin doğudan batıya ve kuzeyden güneye doğru yavaş ve güvenilir bir şekilde hareket etmesi sağlanmıştır. İki eksen toplam 3 W güç tüketiyor. Mevcut güneş takip sistemlerinde takip için harcanan güç miktarı da bu nedenle azaltılmalıdır.

(Menak, 2018) araştırması, güneş panellerinin verimini artırmak hedefiyle geliştirdiği iki eksenli güneş takip sistemi ile sabit bir düzenek üzerine deneysel bir çalışma sunmaktadır. Tasarlanan hareketli düzenek, güneş takibini hem çift eksende hem de tek eksende gerçekleştirebilen bir sistem olarak öne çıkmaktadır. Ayrıca, deneyler sırasında kullanılan sistemin tüm bileşenleri, Matlab/Simulink ortamında modellenmiştir. Deneysel düzenekte bulunan hareketli ve sabit panellerden alınan ölçümler, iki farklı gün boyunca karşılaştırılarak verimlilik analizleri yapılmıştır. Birinci ve ikinci günün karşılaştırma sonuçlarına göre, hareketli sistem sabit sistemden daha yüksek bir verim göstermiştir. Hareketli sistem, birinci günde %37.234 ve ikinci günde %35.756 daha verimli olduğu gözlemlenmiştir. Bu sonuçlar, hareketli sistemlerin sabit sistemlere kıyasla %35-40 arasında daha yüksek verim sağladığına işaret etmektedir.

(Aksoy Tırmıkçı ve Yavuz, 2018) çalışması, Türkiye'nin Sakarya ili için özel olarak tasarlanmış yükseklik-azimut iki eksenli güneş takip sisteminin mekanik yapısını sunmaktadır. Bu tasarım, bir güneş paneli, iki doğrusal aktüatör ve ilgili bağlantılardan oluşmaktadır. Araştırmanın temel hedefi, uygun maliyetli, Sakarya'nın hava koşullarına dayanıklı, kolay monte edilebilir, taşınabilir, bakım gerektirmeyen ve uzun ömürlü bir güneş takip sistemi geliştirmektir. Bu sistem, Güneş'in konumunu hem yükseklik hem de azimut ekseninde etkili bir şekilde takip edebilmektedir. Çalışmanın sonuçları, Sakarya ili ve çevre bölgelerde, bir kontrol ünitesi ilavesiyle optimum şekilde kullanılabilecek bir güneş takip sistemi tasarımının mümkün olduğunu göstermektedir.

3. MATERYAL VE YÖNTEM

3.1. Materyal

3.1.1. Güneş paneli

Bu araştırmada üç adet PV güneş paneli kullanılmıştır. Panelin görüntüsü Şekil 3.1'de gösterilmiştir. Çalışmada kullanılan panellerin her biri 10 W gücündedir. Sistemin kurulum sonrası durumu Şekil 3.2. ve Şekil 3.3. de gösterilmiştir. Panellerin teknik özellikleri çizelge 3.1'de verilmiştir.

Şekil 3.1. Güneş Paneli

Çizelge 3.1. Güneş panelinin teknik özellikleri

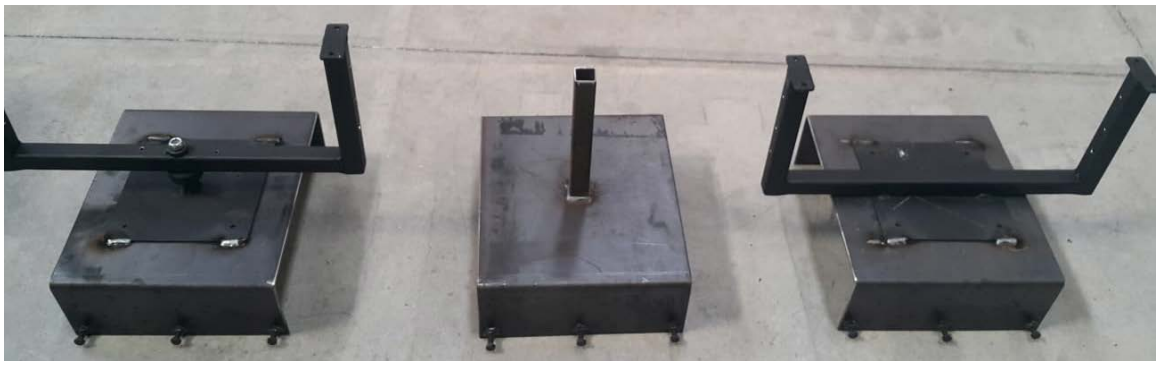
Güneş Paneli	Özellik Değeri
Maksimum güç	10 W
Açık devre voltajı	21.50 V
Kısa devre akımı	0.64 A
Maksimum güç noktası voltajı	17 V
Maksimum güç noktası akımı	0.59 A

Şekil 3.2. Sistemin kurulum sonrası durumu

Şekil 3.3. Sistemin kurulum sonrası durumu

3.1.2. Güneş panellerini taşıyan platform

Güneş takip sisteminde panellerin taşıyacak ve güneş takibi için gereken panel açı düzeltmelerini sağlaması amacıyla 3 adet çelik çerçeve tasarlanmıştır. Şekil 3.4'te çift eksenli (solda), tek eksenli (sağda) ve sabit açılı (orta) panellerde kullanılan metal profil çerçeve gösterilmiştir.



Şekil 3.4. Kullanılan metal profil çerçeveler

3.1.3. Dişli kutusu ve rulmanlar

DC motorlardan elde edilen torkun artırılması ve motorun daha kontrollü bir şekilde açı değişimi sağlaması amacıyla toplam 3 adet dişli kutusu Solidworks üzerinde dizayn edilmiş ve arkasından 3 boyutlu yazıcı ile üretilmiştir. Dişlilerin üretiminde plastik olarak yüksek dayanıklılığı sebebiyle Akrilonitril bütadien stiren (ABS) tercih edilmiştir. Ayrıca ABS plastiğin güneşin morötesi ışınlarından zarar görebildiği için (E. Yousif ve Haddad, 2013) üretilen dişliler sonrasında metalik bir boya ile kaplanmıştır.

Hem çift hem de tek eksenli güneş takip sistemlerinde eksensel doğu-batı dönüşünü sağlamak amacıyla metal profil çerçevenin iki ucuna rulman eklenmiş ve bu rulman bir daire çubuğa takılmıştır. Çubuğun sağ kısmına eklenen yarım dişli ile panellerin doğu batı yönünde 180 derece dönüşü sağlanmıştır. Sistemde kullanılan yarım dişli devamındaki 4 adet dişli takımı ile motora bağlanmıştır. Bu sistemin toplam dişli çevrim oranı 1:100 ve Şekil 3.5'te gösterilmiştir.

Çift eksenli güneş takip sisteminde kuzey güney hareketini sağlamak amacıyla, kullanılan metal profil çerçevenin ortasına konik rulman yatağı yelleştirilmiştir. Rulmanın alt kısmında bulunan dişli kutusu ile panelin herhangi bir engel olmadan 360 derece dönebilmektedir. Sistemde Kullanılan 4 adet dişli takımı ile motora bağlanmıştır. Bu sistemin toplam dişli çevrim oranı 1:140'tır. Şekil 3.5 ve Şekil 3.6 üzerinde Solidworks üzerinde tasarlanan dişli kutusu ve sistem üzerindeki durumu gösterilmiştir.

Şekil 3.5. Doğu batı dönüşünde kullanılan dişli sistemi

Şekil 3.6. Kuzey güney dönüşünde kullanılan dişli sistemi

Şekil 3.7. Dişli sisteminin Solidworks üzerinde gösterimi

3.1.4. Redüktörlü DC motor

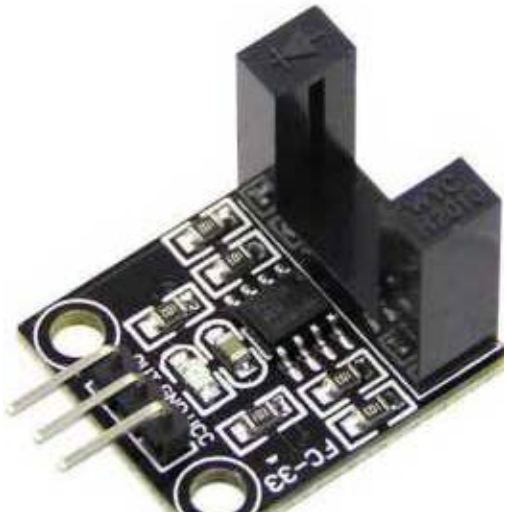
Araştırma projesinde kullanılan 6 V redüktörlü DC motorun tercih edilmesindeki faktörler arasında; 6 V DC gerilim ve 120 mA yük akımı ile çok düşük bir güç tüketimine sahip olmasıyla beraber, robotik ve hobi amaçlı projelerde yaygın olarak kullanılmasından dolayı hem ucuz hem de bir çok hazır motor sensör ve parçalarına uyumluluğu olması ve ayrıca üzerinde bulunan 1/120 dişli çevrim oranlı redüktör sistemi sadece istenen torku artırmak ile kalmayıp, motorun dönüş hızının da azalmasını sağlamaktadır. Ayrıca 3 boyutlu yazıcı ile üretilen dişli kutusu sistemine eklendiğinde, sistemlerin toplam dişli çevrim oranı 1:12000 (doğu batı dişli sistemi) ve 1:15000 (güney kuzey dişli sistemi) olmaktadır. Bu yüksek dişli çevrim oranı sadece motorun torkunu ve kontrolünü artırmak ile kalmayıp aynı zamanda motorun aktif olmadığı zamanlar için otomatik dişli kilit sistemi görevi de sağlamaktadır. Bu sayede sistemde rüzgâr gibi çevresel faktörlerden oluşabilecek istenmeyen açı kaymaları engellenmektedir. Yapılan testlerde panelin üzerine 20 kg yük bağlanması durumunda bile motorun sistemi hareket ettirebildiği fakat motor durduğunda sistemin sabit kaldığı gözlenmiştir. Şekil 3.8'da sistemde kullanılan redüktörlü 6 V DC motor gösterilmiştir.

Şekil 3.8. 6 V redüktörlü DC motor (Kumar, 2018)

3.1.5. Motor konumu ve kalibrasyonu

Araştırma projesinde tercih edilen redüktörlü motorun konumunun ve yaptığı açı değişimlerinin takip edilebilmesi amacıyla LM393 hız sensörü kullanılmıştır. LM393 motor hız sensörü modülü, motorların dönüş hızının ölçümü ve tespiti için tasarlanmış bir cihazdır. Bu modülün özünde, kızılötesi iletim ve alım modülüyle birleştirilmiş LM393 voltaj karşılaştırıcı çipi kullanılır. Modülün ana bileşenleri arasında LM393 yongası, bir kızılötesi verici, bir kızılötesi alıcı, hassasiyet ayarı için bir potansiyometre ve bir güç göstergesi LED'i bulunur. Şekil 3.9'de yaygın kullanılan bir LM393 motor hız sensörü gösterilmiştir.

LM393 motor hız sensörü modülünün çalışma prensibi, verici tarafından dönen nesneye doğru bir kızılötesi sinyalin yayılmasını içerir. Kızılötesi alıcı daha sonra yansıyan sinyali yakalar. Yansıyan sinyaldeki değişiklik dönüş hızının göstergesidir. LM393 çipi bu bilgiyi işler ve dönen nesnenin hızıyla orantılı bir çıkış sinyali üretir.

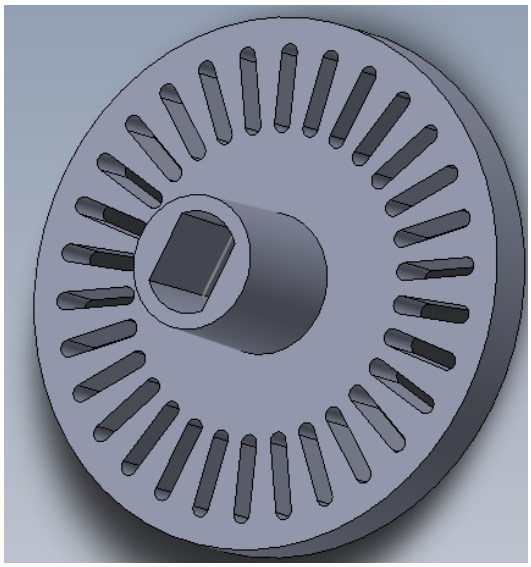


Şekil 3.9. LM393 motor hız sensörü (Electronics, 2017)

Temel olarak, optik sensörün bir tür yarıklı tekerlek veya benzeri bir mekanizma tarafından fiziksel olarak kesildiğinde işlenmiş darbe dizileri veren basit bir cihazdır (optik sensör genellikle bir ışık yayıcıdan oluşur). Aydınlatmayı sağlayan diyot ve bu aydınlatmanın varlığını veya yokluğunu algılayan bir foto transistörden oluşur. Burada kullanılan iletici optik sensör, bir kızılötesi ışık yayan diyot ve bir foto transistörden oluşur. Bu, hem dış ışık kaynaklarından gelen paraziti önler hem de iki bileşenin belirli bir radyasyon frekansı için eşleştirilmesiyle, istenmeyen parazitlere karşı daha da verimli çalışmasını sağlar.

Optik enkoder, en sık kullanılan çeşittir. Optik algılama sisteminde, ışık kaynağından yayılan ışık demeti belirli engeller ve opak desenlerle şekillendirilmiş kod diskinden geçirilir. Kodlayıcı şaftı dönerken kaynaktan gelen ışık demeti, dönen kod diskindeki desenlere göre engellenir ya da diğer tarafa geçer. Şekil 3.10'de 3 boyutlu yazıcı ile yaptığım 30 bölüme sahip optik disk gösterilmiştir.

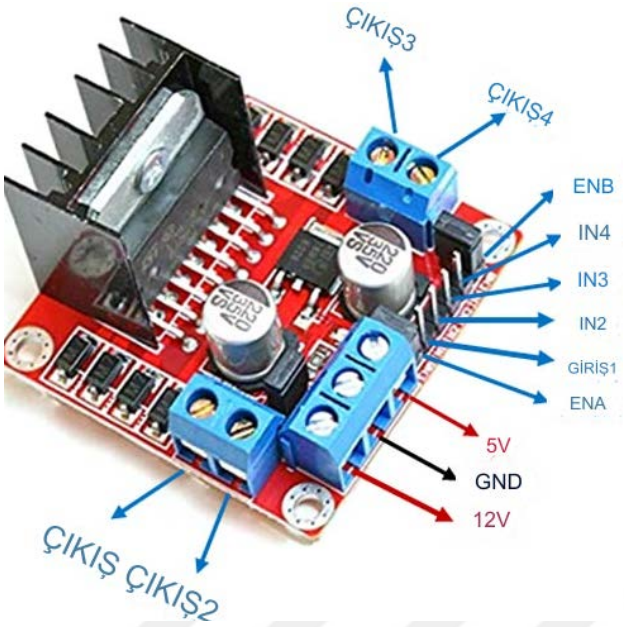
Bu sinyalizasyona darbe sinyali denir. Karşı tarafa geçen ışık demeti, karşı tarafı aydınlatır. Bu "açık" demektir. Karşı tarafı aydınlatmadığında karanlık, yani "kapalı" anlamına gelir. Bu iki sinyal, fotodedektör ile sayaca ya da denetleyiciye gönderilir.



Şekil 3.10. Optik disk

3.1.6. L298N DC motor sürücü modülü

L298N DC Step Motor Sürücü 4.8 V - 46 V arasındaki iki motoru sürmek için hazırlanmış bir motor sürücü kartıdır. İki kanallı bu motor sürücü kanal başına 2 A akım vermektedir. Kart üzerinde L298N motor sürücü entegresi kullanılmıştır. Birbirinden bağımsız olarak iki ayrı motoru kontrol edebilir. Kanal başına 2 A akım verebilmektedir. Üzerinde dahili regülatörü vardır. Yüksek sıcaklık ve kısa devre koruması vardır. Motor dönüş yönüne göre yanan ledler vardır. Kart üzerinde dahili soğutucu vardır. Akım okuma pinleri dışa verilmiş haldedir. Şekil 3.11'da L298N DC motor sürücü modülü gösterilmiştir.



Şekil 3.11. L298 DC motor sürücü modülü (Rittenberry, 2005)

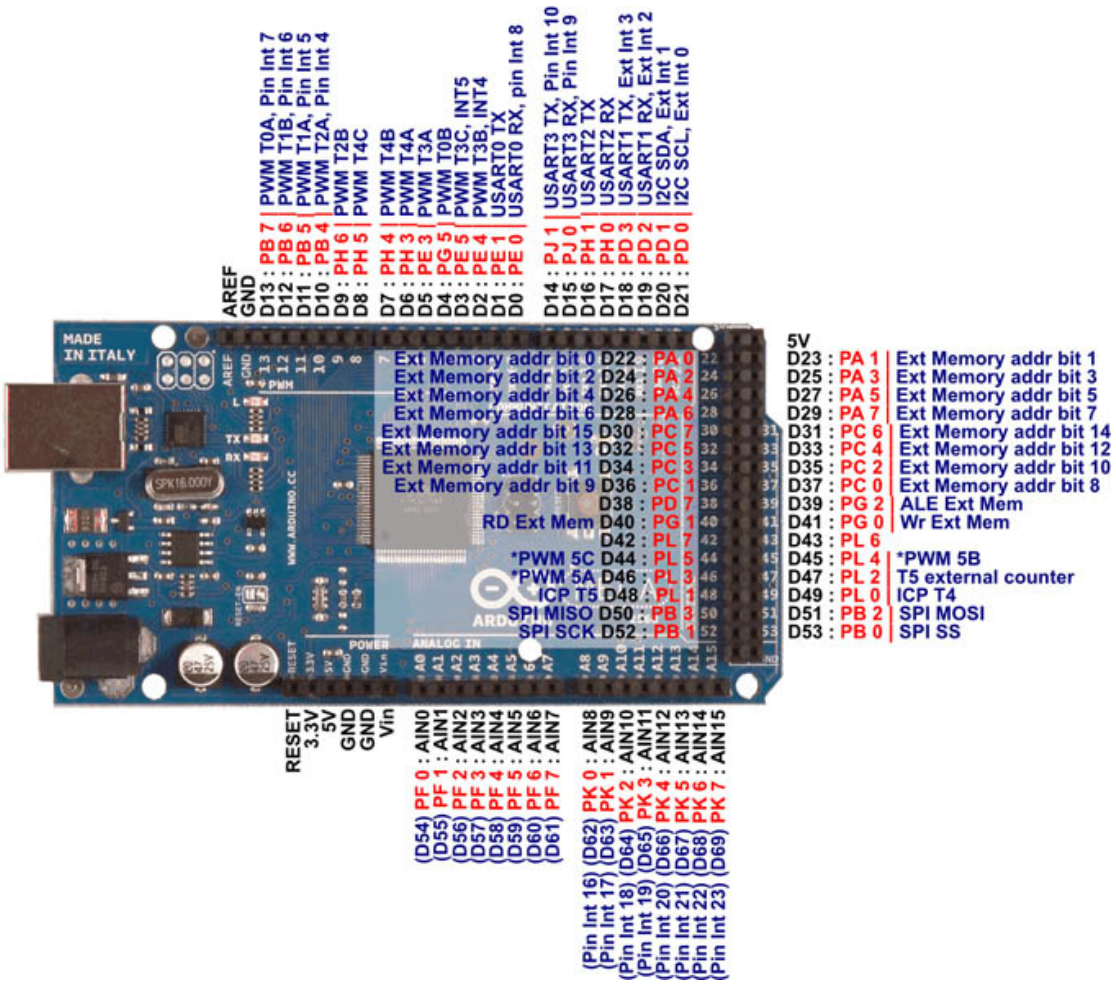
Çizelge 3.2. L298 DC motor sürücü modülü

L298N	Özellikler
Boyutları	57 mm x 43 mm
Ağırlık	29 gr
ENA	Sol motor kanalını aktif etme pini
ENB	Sağ motor kanalını aktif etme pini
IN1	Sol motor 1. girişi
IN2	Sol motor 2. girişi
IN3	Sağ motor 1. girişi
IN4	Sağ motor 2. girişi
OUT1	Sol motor 1. çıkışı
OUT2	Sol motor 2. çıkışı
OUT3	Sağ motor 1. çıkışı
OUT4	Sağ motor 2. çıkışı
GND	Toprak bağlantısı
VS	Besleme voltaj girişi(4.8 V-46 V)
VSS	Kart besleme girişi (6 V-12 V)

3.1.7. Arduino Mega 2560 R3 Mikro Denetleyici

Arduino Mega 2560 R3, ATmega2560 tabanlı bir mikro denetleyici kartıdır. 54 adet dijital giriş/çıkış pini (15 tanesi PWM çıkışı olarak kullanılabilir), 16 adet analog giriş, 4 adet UART (donanımsal seri port), 16 MHz kristal osilatör, USB bağlantısı, güç

girişi, ICSP başlığı ve bir sıfırlama düğmesi. Mikrodenetleyiciyi desteklemek için gereken her şeyi içerir; Başlamak için bir USB kablosuyla bilgisayara bağlamanız veya AC 'den DC'ye adaptör veya pille çalıştırmanız yeterlidir. Mega 2560 anakartı, Uno ve eski anakartlar Duemilanove veya Diecimila için tasarlanmış çoğu shield ile uyumludur. Şekil 3.12'da Arduino Mega 2560 R3 'ün kısımları gösterilmektedir.

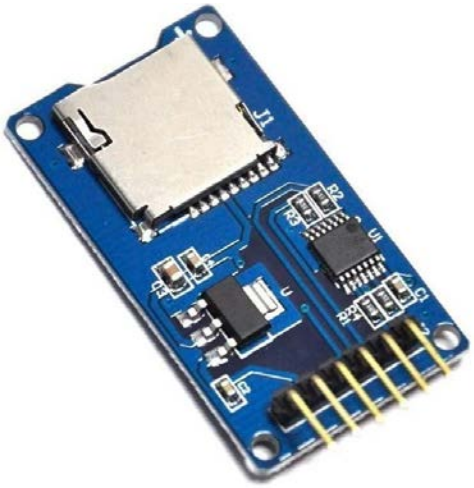


Şekil 3.12. Arduino Mega 2560 R3 pin şeması (Kusriyanto ve Putra, 2017)

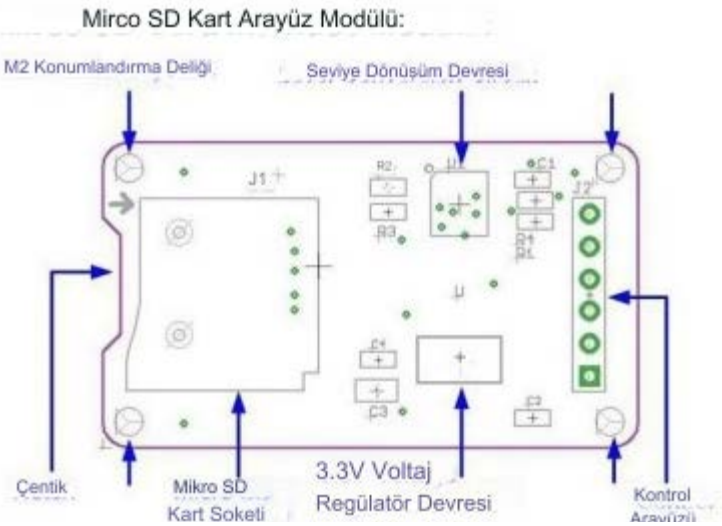
3.1.8. Arduino Mikro Hafıza Kardı Modülü

Şekil 3.13'de gösterilen Arduino Mikro Hafıza Kardı Modülü ile fotovoltaik panel üzerinde üretilen gerilim ve akım değerlerini otomatik olarak bir mikro hafıza kartına aktarmaktadır. Bu modülün kullanılmasındaki önemli bir neden ise çoğu uygulamanın aksine Arduino Mega 2560 R3 Mikrodenetleyici sadece zamanlama modülünden gelen veriler doğrultusunda aktif hale gelmekte bu sayede önemli bir tasarruf sağlanmaktadır.

Şekil 3.13'de Arduino Mikro Hafıza Kardı Modülünün katalog bilgileri ve Şekil 3.14'da Arduino Mikro Hafıza Kardı Modülünün devresi gösterilmektedir.



Şekil 3.13. Arduino mikro hafıza kardı modülü (EBay, 2013)



Şekil 3.14. Mikro hafıza kardı modülünün katalog bilgileri (EBay, 2013)

3.1.9. DS1302 Gerçek zamanlı saat modülü

Gerçek zamanlı saat modülü; yıl, gün saat, dakika ve saniye bilgilerini sürekli olarak içerisinde bulundurması sebebiyle zaman verilerinin okunması için kullanılmaktadır. Şekil 3.15'de kullanılan RTC modülü görülmektedir. Sistemde DS1302 RTC kullanılmıştır ve Arduino Mega 2560 R3 mikrodenetleyici üzerine entegre

edilmiştir. DS1302 RTC, zaman sapmalarının az olması sebebiyle tercih edilmektedir. Elektrik bulunmadığı durumlarda harici pil ile de çalışabilmektedir. Ayrıca sistem çalışmasa bile zaman bilgisini hafızasında tutabilmektedir.

Şekil 3.15. DS1302 (Maxim Integrated, 2015)

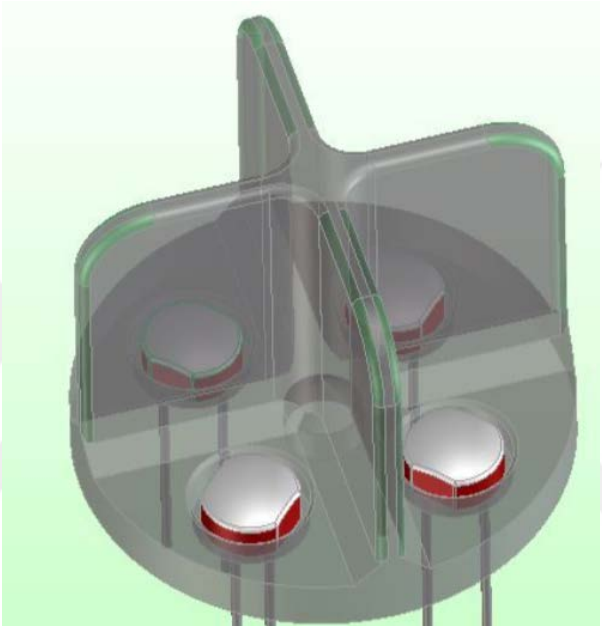
3.1.10. Foto dirençler

Optik sensör türleri içerisinde akla gelen ilk elektronik elemandır. İngilizce Photo Resistor anlamına gelmesine karşın foto dirençler yaygın bir şekilde LDR adı ile ifade edilir. İsminden de anlaşılacağı üzere LDR, Light Dependet Resistance kelimelerinin kısaltılmış halidir. Ortamdaki ışık şiddetine karşı direnç değerinde değişim gösterir. Direnç değeri aydınlıkta azalan, karanlıkta ise artan elemana foto direnç denir.

Direnç değeri aydınlıkta azalan, karanlıkta ise artan elemana foto direnç (LDR) denir. Tam aydınlık bir alanda yani üzerine güneş ışığı geldiğinde direnç değeri 5-10 Ω değerleri arasına kadar düşebilir. Tam karanlık bir ortamda yani üzerine az ya da hiç ışık düşmezken direnç değeri 200 M Ω gibi oldukça yüksek direnç değerleri gösterir. Yani foto direnç, üzerine düşen ışık arttıkça direnç değeri lineer olmayan bir şekilde azalır. Bu yüzden ışık şiddetinin artması direnç değerinin düşmesine, ışık şiddetinin azalması ise direnç değerinin artmasına sebep olur. Bu özelliğinden dolayı ışık şiddeti farkı ile kontrol edilmek istenilen tüm elektronik devrelerde kullanılabilir.

Bir diğer bilinmesi gereken önemli bilgi ise foto dirençler AC ve DC akım türlerinde aynı özellikleri gösterirler. Şekil 3.16'te çift eksenli güneş takibi için kullanılan foto direnç parçası gösterilmiştir.

Foto dirençlerin yapısında bulunan kalsiyum sülfat ve kadmiyum selenid gibi bazı maddeler üzerlerine düşen ışık ile ters orantılı olarak direnç değişimi gösterir. Bu tür maddeler yalıtkan bir taban üzerine yerleştirilir ve içinde ince sarmallar halinde iletken bir tel geçirilir (çoğunluk olarak bakır). Bu iletkenin iki ucu dışarıya çıkartılarak elemanın ayakları teşkil edilir. Son olarak elemanın yüzeyi saydam bir madde ile kaplanır böylece ışık geçirirken dayanımı artırılmış olur ve foto direnç kullanıma hazırdır.



Şekil 3.16. Foto direnç (Sensors, 2008)

3.2. Yöntem

3.2.1. Panel güç ölçüm sistemi

Güneş takip sistemlerinde, güneş panellerinden elde edilen gücün ölçülmesi, sistemin verimliliğini ve performansını değerlendirmek için önemlidir. Bu amaçla kullanılan güç ölçme devreleri, panelin maksimum güç durumundaki voltaj ve akım değerlerine uygun bir dirence sahip voltaj bölücü devreleridir.

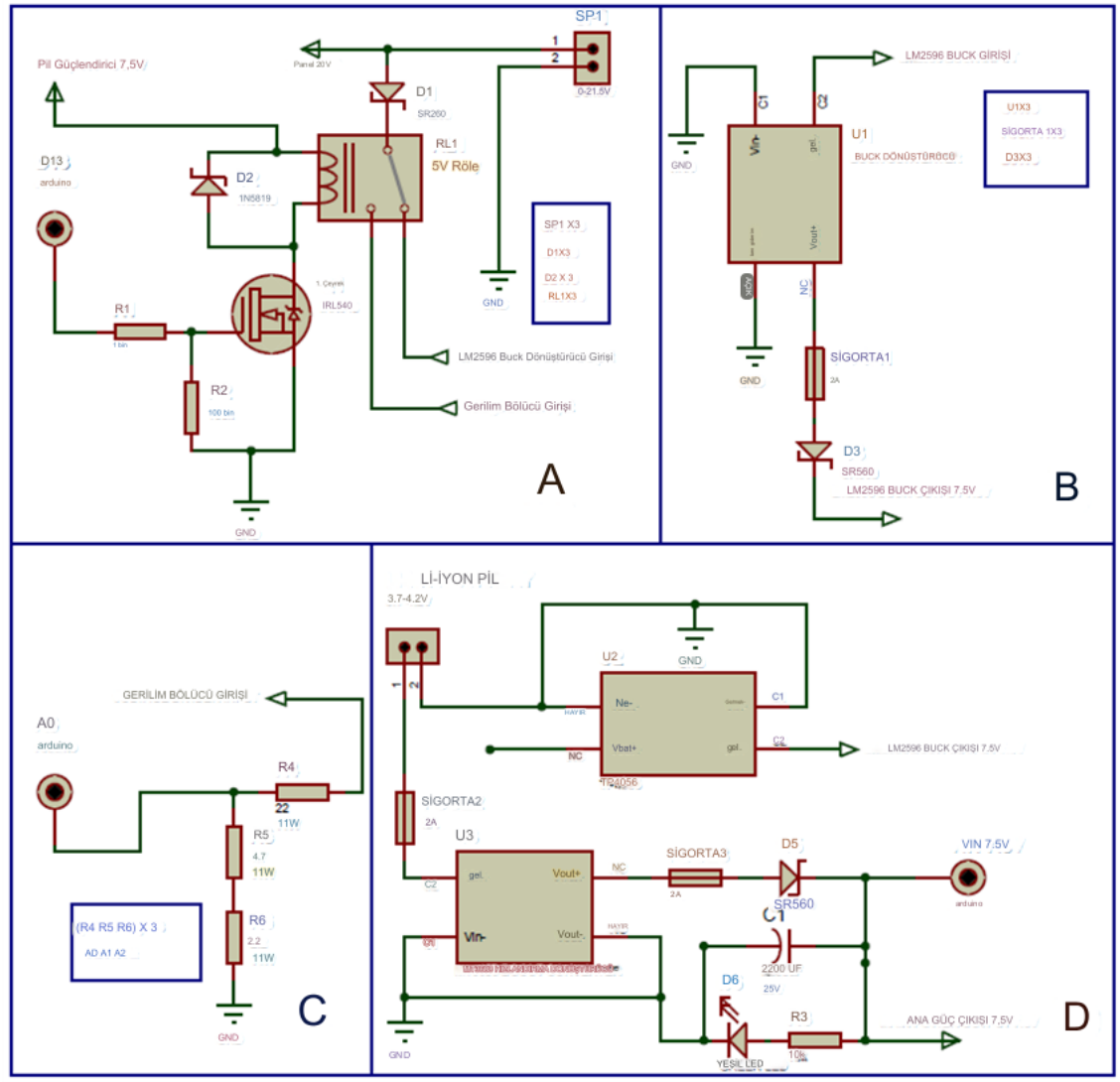
Bu çalışmada, bir güneş takip sistemi projesinde panellerden elde edilen gücün ölçülmesi için bir yük devresi tasarlanmıştır. Devre, 22 Ω 11 W, 4.7 Ω 11 W, 2.2 Ω 11

W değerinde 3 adet dirençten oluşan bir voltaj bölücü devresi Şekil 3.17 C bölümünde gösterilmiştir.

Devrenin tasarımında, panelin maksimum güç durumundaki voltaj ve akım değerleri dikkate alınmıştır. Panelin maksimum güç durumundaki voltaj değeri 17 V, akım değeri ise 0.59 A olarak alınmıştır. Bu değerlere göre, devrenin çıkış voltajının 4 V olması sağlanmıştır.

Devrenin çıkış voltajını ölçmek için Arduino Mega 2560 R3 Mikro denetleyicinin analog pinleri kullanılmıştır. Mikro denetleyicinin ölçüm aralığı 0-5V ve çözünürlüğü 0-1023 arasındadır.

Arduino Mega 2560 R3 Mikro denetleyicinin, DC motorların, L298N motor sürücülerinin ve sistemdeki diğer elektronik parçaların sabit bir güçte çalışabilmesi için sisteme 4 adet 2 Ah kapasitesinde lityum iyon batarya eklenmiştir. Lityum iyon bataryalardan elde edilen 3.7-4.2 V voltaj, MT3608 voltaj yükseltici modülü ile 7.5 V değerine yükseltilmiştir. Panellerden elde edilen 0-21 V çıkış voltajının, lityum iyon bataryaları şarjı amacıyla kullanılmak istenildiği için sisteme 3 adet LM2596 voltaj düşürücü modülü eklenmiştir, bu modüllerin çıkış voltajı 7.5 V tur. LM2596 voltaj düşürücü modülünden elde edilen 7.5 V voltajın lityum iyon bataryaların şarj etmesini sağlamak için sisteme 4 adet TP4056 lityum iyon şarj modülü ve sistemin güç ölçümü ve batarya şarjı arasında geçiş yapılması için 3 adet 5V röle eklenmiştir. Arduino röleleri güç ölçümü sırasında değiştirmektedir. Şekil 3.4 A bölümünde geçiş için kullanılan röle devresi, Şekil 3.17 B bölümünde voltaj düşürücü LM2596 modülünün devresi, Şekil 3.17 D bölümünde TP 4056 şarj modülü, lityum iyon batarya ve MT3608 voltaj yükseltici devresi gösterilmiştir.



Şekil 3.17. Panel Güç ölçüm Sistemi

3.2.2. Hafıza Kartı Kayıt Yazılımı

Sistemde bulunan mikro hafıza kardı modülü ile sadece panellerin güç ölçümlerinin kaydedilmesi ile kalmayıp aynı zamanda sistemin durum rapor kaydedici, batarya şarj seviyeleri kaydedici ve kalıcı verilerin kaydedilmesi amacıyla kullanılmaktadır.

Kullanılan Arduino yazılımının boyutunun büyük olmasından dolayı, yazılım başlık (header) ve kaynak dosyalarına (source) ayrılmıştır. Panelin ölçüm sisteminin yazılımında öncelikle, genel analog ölçüm için kullanılan, ldr_sensor dosyasının sensor_analog fonksiyonu aşağıda gösterilmiştir.

```
void LdrSensor::sensor_analog(int  x, int y, int z1 = 0, int z2 = 0, int z3 = 0, int z4 = 0) {
  int analogInputPins[] = { z1, z2, z3, z4 };
  int numPins = 0;
  // Aktif pin sayısını belirle
  for (int i = 0; i < sizeof(analogInputPins) / sizeof(analogInputPins[0]); i++) {if (analogInputPins[i] != 0) {numPins++;} else { break; }}
  int readings[numPins][x];
  int index[numPins] = { 0 };
  unsigned long total[numPins] = { 0 };
  // İlk okumaları atla
  while (index[0] < y) {
   for (int i = 0; i < numPins; i++) {
    analogRead(analogInputPins[i]);
    delay(5);
    index[i]++;
   }}
  // Analog girişleri birden çok kez oku ve ortalamayı hesapla
  if (!(index[0] < y)) {
    for (int i = 0; i < x; i++) {
     for (int j = 0; j < numPins; j++) {
      readings[j][index[j]] = analogRead(analogInputPins[j]);
      delay(5);
      total[j] += readings[j][index[j]];
      index[j] = (index[j] + 1) % x;
     }}
   // Her pin için stabil okumaların ortalamasını hesapla
   int average[numPins];
   for (int i = 0; i < numPins; i++) {
    average[i] = total[i] / x;
    sensoranalogA[i] = (1023 - (average[i]));
   }
```

```
  // Değişkenleri bir sonraki döngü için sıfırla
  for (int i = 0; i < numPins; i++) {
    total[i] = 0;
    index[i] = 0;
  }}
 return;
}
```

Kullanılan bu fonksiyon sayesinde, istenilen ölçüm hassasiyetine göre devamlı bir analog ölçümü yapılıp, bu ölçümler bir dizi içerisinde depolanmaktadır. Ölçüm sonunda bu dizinin ortalamasının alınmasıyla istenen analog ölçüm değerleri elde edilmiştir. Bu sistemin önemli bir avantajı, özellikle panellerin yük direncinin gerilim değerleri ölçülürken, sadece tek bir analog okuma değeri yerine, 15 saniyelik bir süreçte her bir panel için 1000 adet ölçüm yapılmıştır. Bu sayede anlık gerilim değişimleri ve gürültü gibi istenmeyen etkenlerden kaçınılmıştır. Fakat bu sistemin dikkat edilmesi gereken bir dezavantajı, analog ölçümlerinin kayıt alındığı dizinin boyutu ve bunun Arduino'nun hafızasına olan etkisidir. Örneğin sadece panel ölçümü için kullanılan 3000 adet değerin boyutu yaklaşık 6000 bittir ve kullanılan Arduino mikro işlemcinin toplam belleği 8192 bittir. Panellerin ölçümü ve kaydı için aşağıdaki panel ölçüm fonksiyonu kullanılmıştır.

```
void LdrSensor::panel_measure() {
  Serial.println("start reading");
  digitalWrite(relayHON, HIGH);
  delay(100);
   sensor_analog(numReadingspanel, numInitialReadingsToDiscard, panelA1, panelA2, panelA3);
  delay(100);
  digitalWrite(relayHON, LOW);
  Serial.println("end reading");
  for (int i = 0; i < 3; i++) {
   panelA[i] = (1023 - (sensoranalogA[i]));
  }
  for (int i = 0; i < 3; i++) {
```

```
  Serial.print("Panel");
  Serial.print(i + 1);
  Serial.print(" = ");
  Serial.println(panelA[i]);
 }
 Serial.println("end function");
 return;}
```

Bu fonksiyon öncelikle panellerin çıkış uçlarını kullanılan röle sayesinde batarya şarjı yerine yük dirençlerine bağlamaktadır. Sonrasında bir önceki kısımdaki analog ölçüm fonksiyonu ile panellerin bağlı olduğu yük dirençlerinin gerilimi ölçülür ve ölçüm sonrasında röle tekrar panellerin çıkış uçlarını batarya şarj sistemine bağlar. Elde edilen ölçüm değerleri panel dizine aktarılır. Devamında bu ölçümlerin mikro hafıza kartına aktarılması ve gereken dosyanın oluşturulması için aşağıdaki fonksiyon kullanılmıştır.

```
void sd_card_panel_write() {
 rtc_update();
 bool countr = false;  // true means file is new
 Serial.println("Start sd_card_panel_write function start");
 int counti;
 test = reset;
 test += " ";
 test += myday;
 test += " ";
 test += mymonth;
 test += " ";
 test += myyear;
 test += ".csv";
 const char *fileName = test.c_str();
 Serial.println("Start sd_card_panel_write function start1");
 if (SD.exists(fileName)) {
  Serial.println("File exists!");
  countr = false;
```

```
} else {
  Serial.println("File  does not exist!");
  countr = true;
}
Last_Rtc_File = test;
if (countr == true) {
  counti = sd_card_file_storage(true, 0);
} else {
  counti = sd_card_file_storage(false);
  Serial.print("previous  counter =");
  Serial.println(counti);
  counti++;
  counti = sd_card_file_storage(true, counti);
  Serial.print("current  counter =");
  Serial.println(counti);   }
Serial.println(test);
if (counti == 0) {
  sd_write(test.c_str(), "Date,Panel1ADC,Panel2ADC,Panel3ADC");  }
Serial.println("Start  sd_card_panel_write function start end while");
if (counti == 10) {
  Serial.println("read  file time");
  sd_read(test.c_str());  }
sd_addrow(test.c_str(), date_hms, ldr_dual_C.panelA[0], ldr_dual_C.panelA[1],
ldr_dual_C.panelA[2]);
Serial.println("End  sd_card_panel_write function end");
return;}
```

Kullanılan bu fonksiyonun temel amacı, kayıt için kullanılan csv dosyasının ismi ve aynı dosya üzerine sıralı bir biçimde kayıt alınmasında kullanılmıştır. Ayrıca fonksiyon aynı dosya üzerine 50 satır yani 1 günlük panel ölçümleri yazıldığında, dosyayı okuma modunda kontrol etmektedir. Kayıt dosyalarının isimlerinin düzenlenmesi ve ölçüm yapılan saatin kaydı için DS1302 saat modülünden elde edilen veriler kullanılmıştır. Kullanılan Arduino yazılımı ekte verilmiştir.

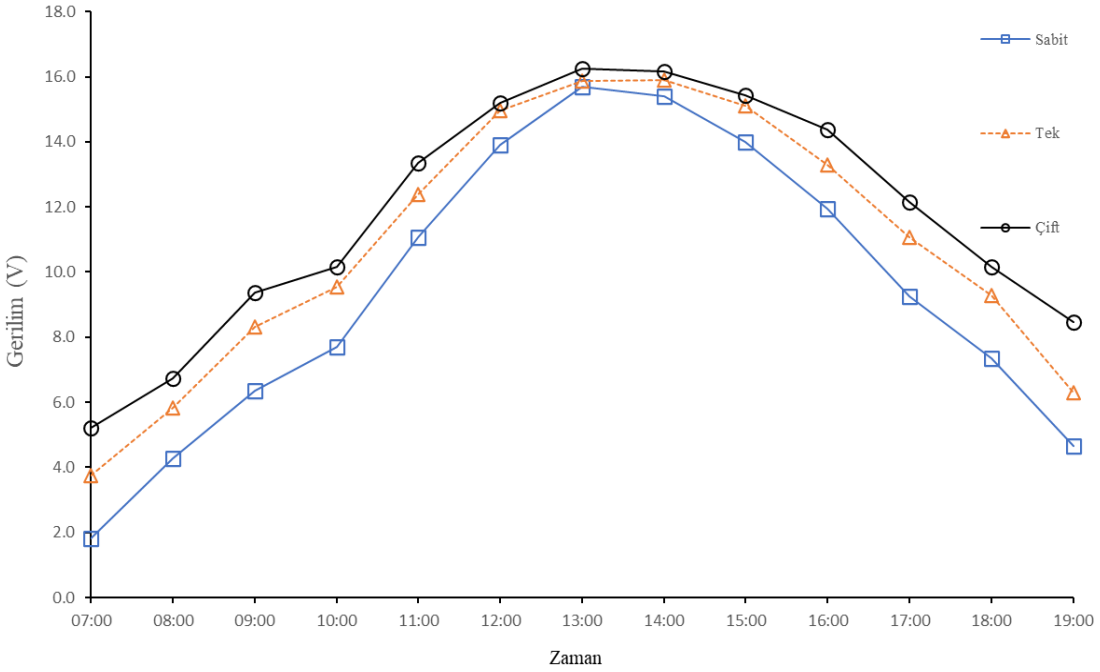
4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

4.1. Araştırma sonuçları

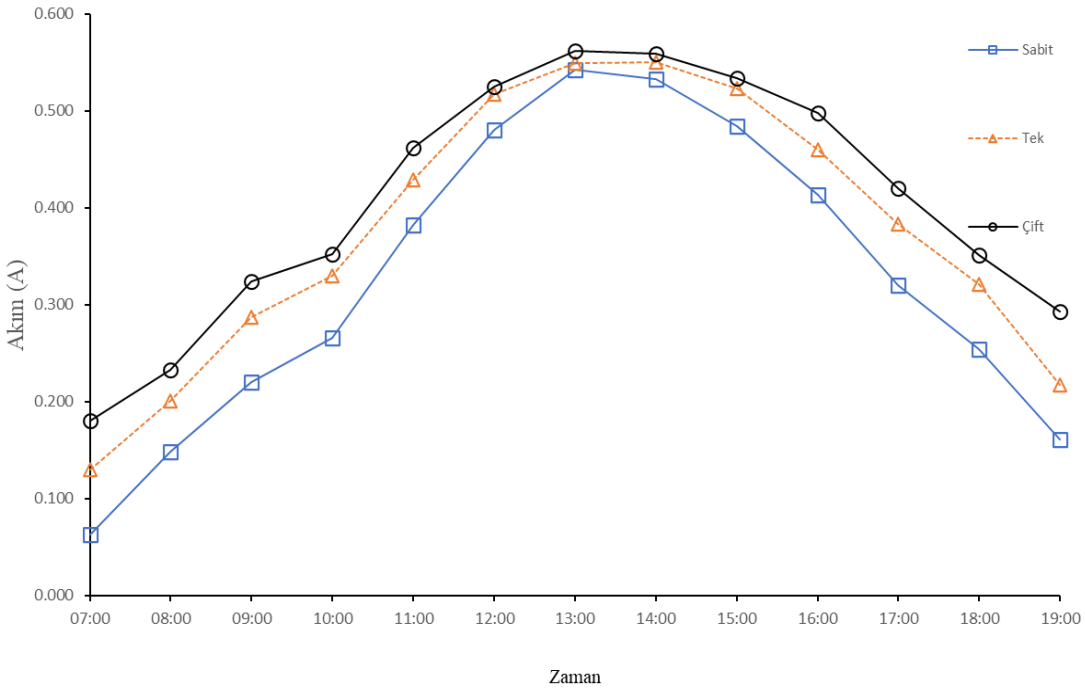
Konya ilinde bulunan güneş takip sisteminin 20 Temmuz ,16 Ağustos ,6 Ocak ve 7 Ocak tarihlerinde yapılan, çift eksenli, tek eksenli ve sabit açılı panellerden elde edilen gerilim ve akım ölçüm sonuçları aşağıdaki şekil ve çizelgelerde gösterilmiştir. Diğer veriler yani rüzgâr, hava sıcaklığı ve güneş yükseklik açısı verileri Etimesgut Hava Üssü kaynaklı olup weatherspark internet sitesinden alınmıştır. Diğer deney yapılan günlere ait detaylı deney sonuçları EK 9, 10, 11 ve 12' de verilmiştir.

Çizelge 4.1. 20 Temmuz tarihleri arasında yapılan ölçümler

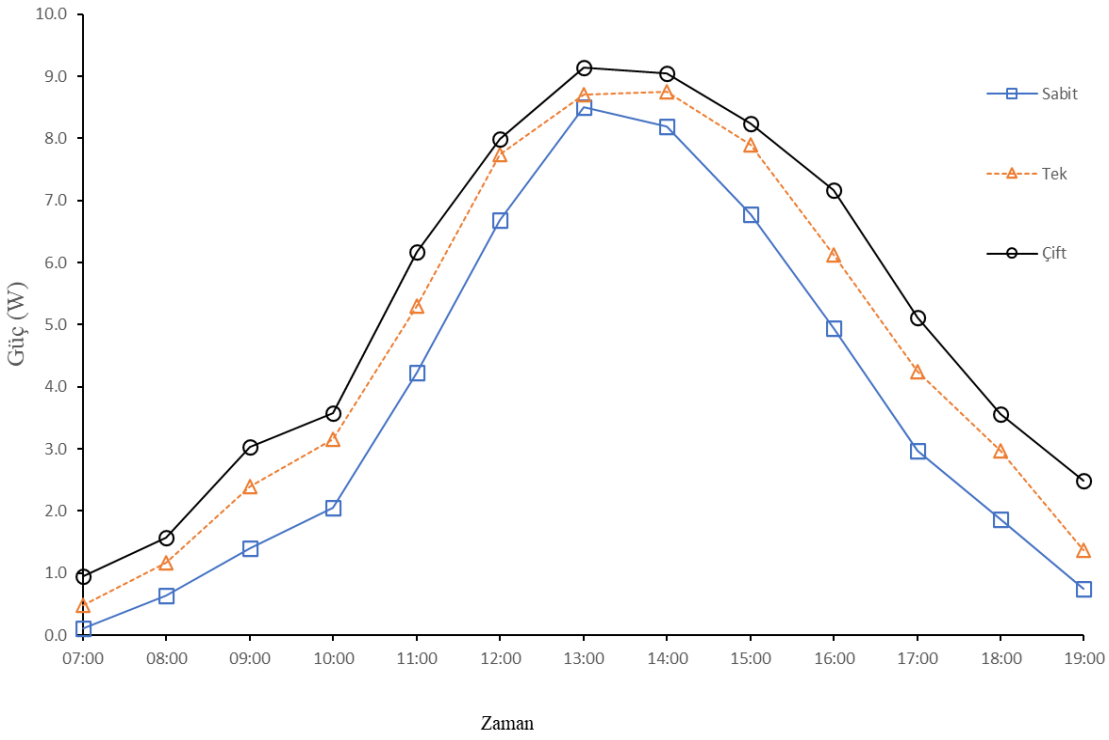
	Gerilim (V)			Akım (A)			Güç(W)		
Zaman	Sabit	Tek	Çift	Sabit	Tek	Çift	Sabit	Tek	Çift
07:00	1.822	3.746	5.220	0.063	0.130	0.181	0.115	0.486	0.943
08:00	4.278	5.814	6.735	0.148	0.201	0.233	0.633	1.170	1.570
09:00	6.367	8.311	9.355	0.220	0.288	0.324	1.403	2.390	3.028
10:00	7.697	9.540	10.174	0.266	0.330	0.352	2.050	3.149	3.582
11:00	11.054	12.385	13.347	0.383	0.429	0.462	4.228	5.308	6.164
12:00	13.900	14.964	15.190	0.481	0.518	0.526	6.685	7.749	7.984
13:00	15.681	15.865	16.254	0.543	0.549	0.562	8.508	8.709	9.142
14:00	15.394	15.906	16.172	0.533	0.550	0.560	8.200	8.754	9.050
15:00	14.002	15.108	15.435	0.485	0.523	0.534	6.784	7.898	8.244
16:00	11.955	13.306	14.391	0.414	0.460	0.498	4.946	6.127	7.166
17:00	9.253	11.075	12.160	0.320	0.383	0.421	2.963	4.244	5.116
18:00	7.349	9.273	10.154	0.254	0.321	0.351	1.869	2.976	3.567
19:00	4.647	6.285	8.475	0.161	0.217	0.293	0.747	1.367	2.485



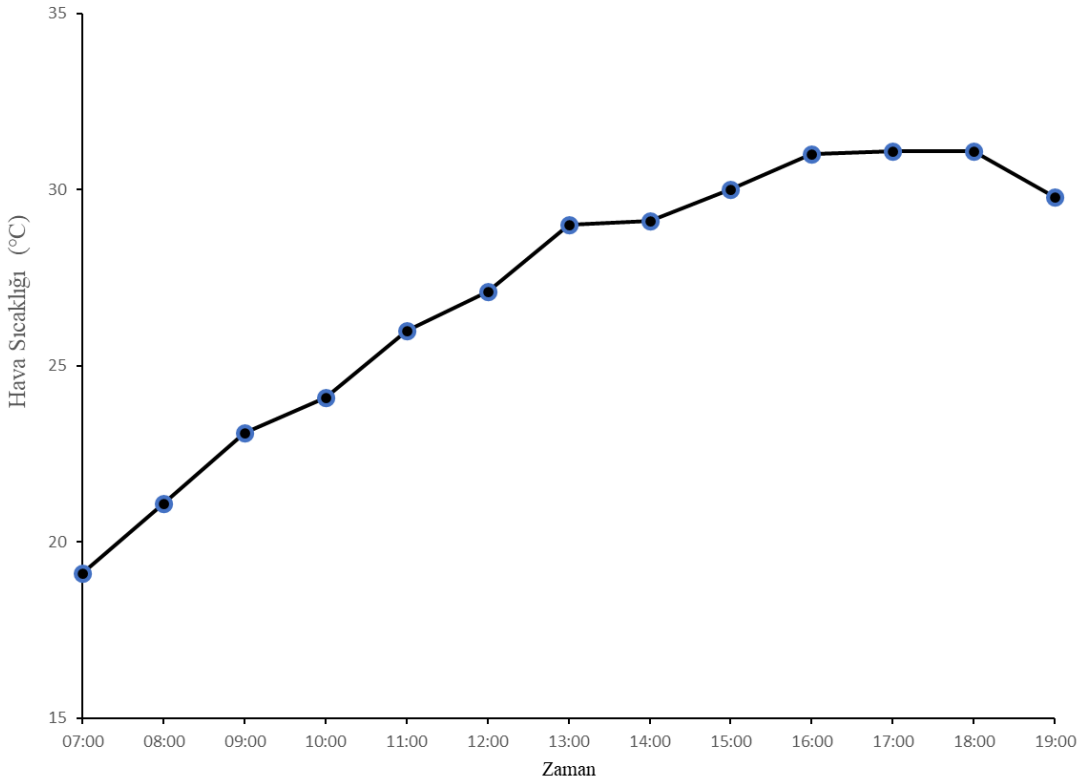
Şekil 4.1. 20 Temmuz deneysel verileri Gerilim grafiği



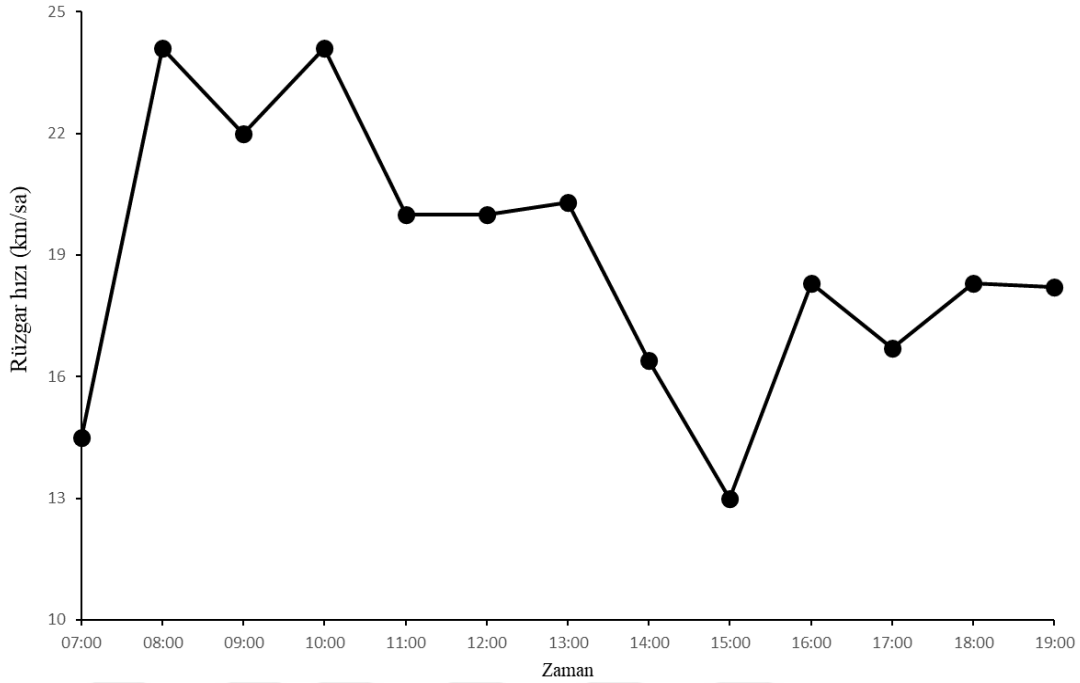
Şekil 4.2. 20 Temmuz deneysel verileri akım grafiği



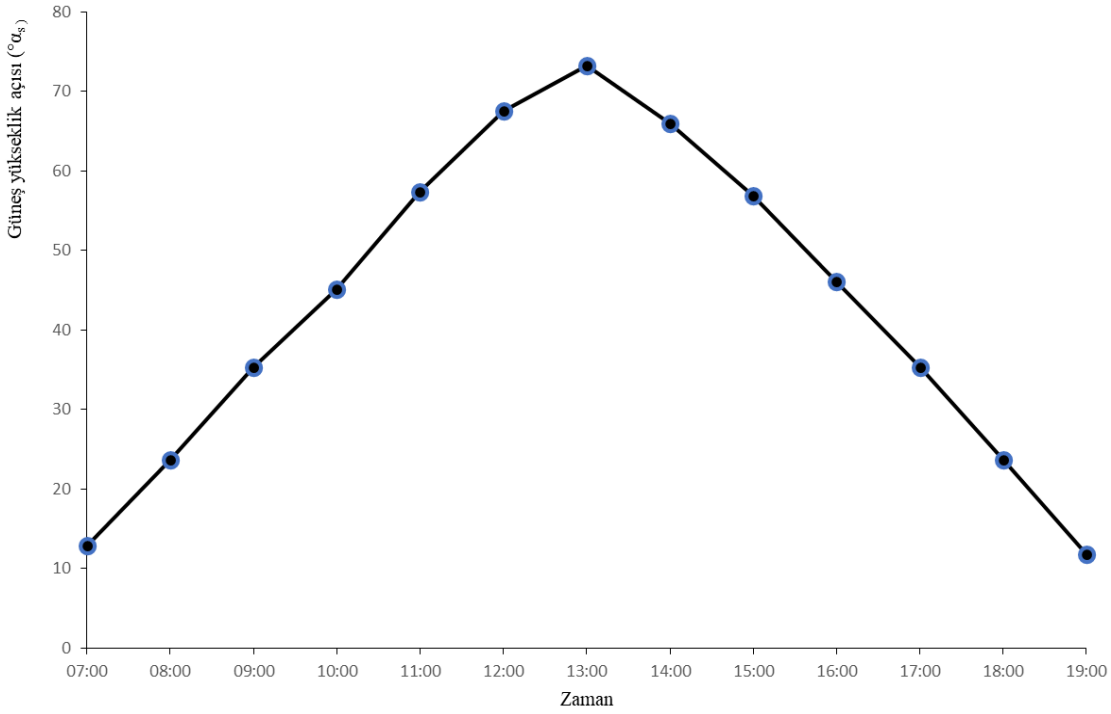
Şekil 4.3. 20 Temmuz deneysel verileri güç grafiği



Şekil 4.4. 20 Temmuz hava sıcaklığı grafiği (Spark, 2023)



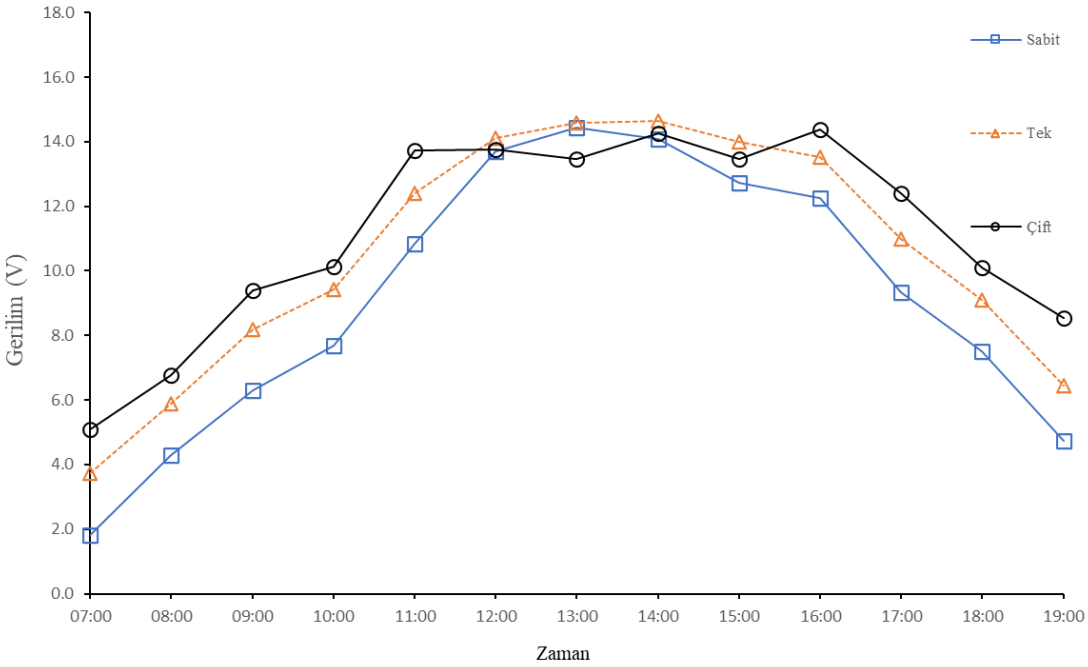
Şekil 4.5. 20 Temmuz rüzgâr hızı grafiği (Spark, 2023)



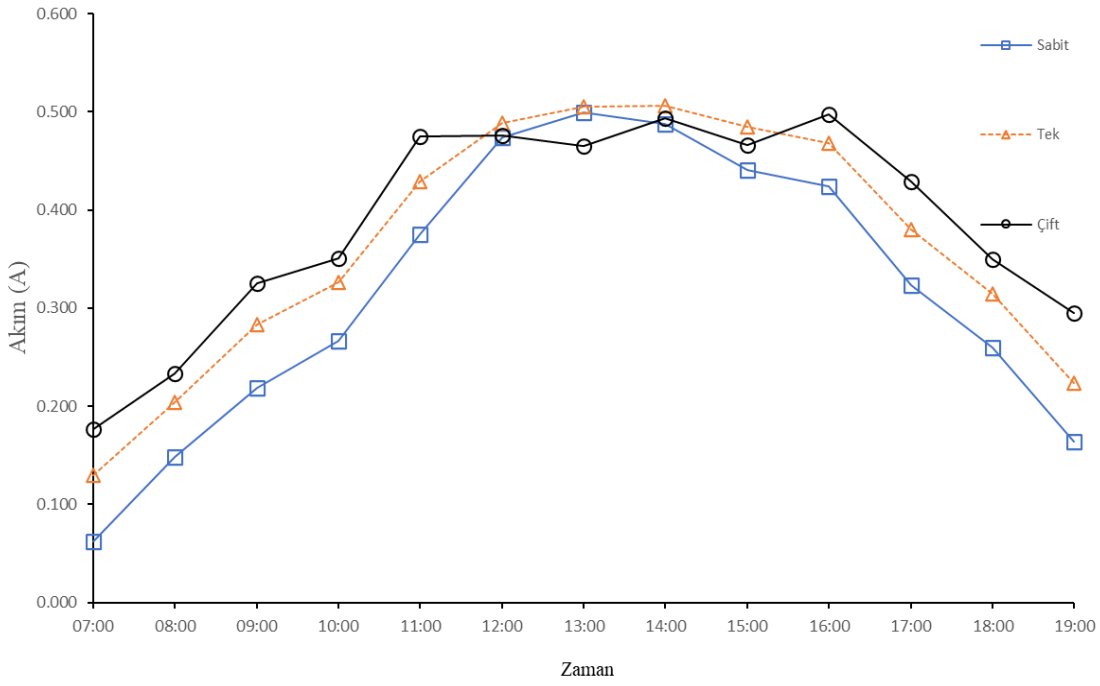
Şekil 4.6. 20 Temmuz güneşin yükseklik açısı grafiği (Spark, 2023)

Çizelge 4.2. 16 Ağustos arasında yapılan ölçümler

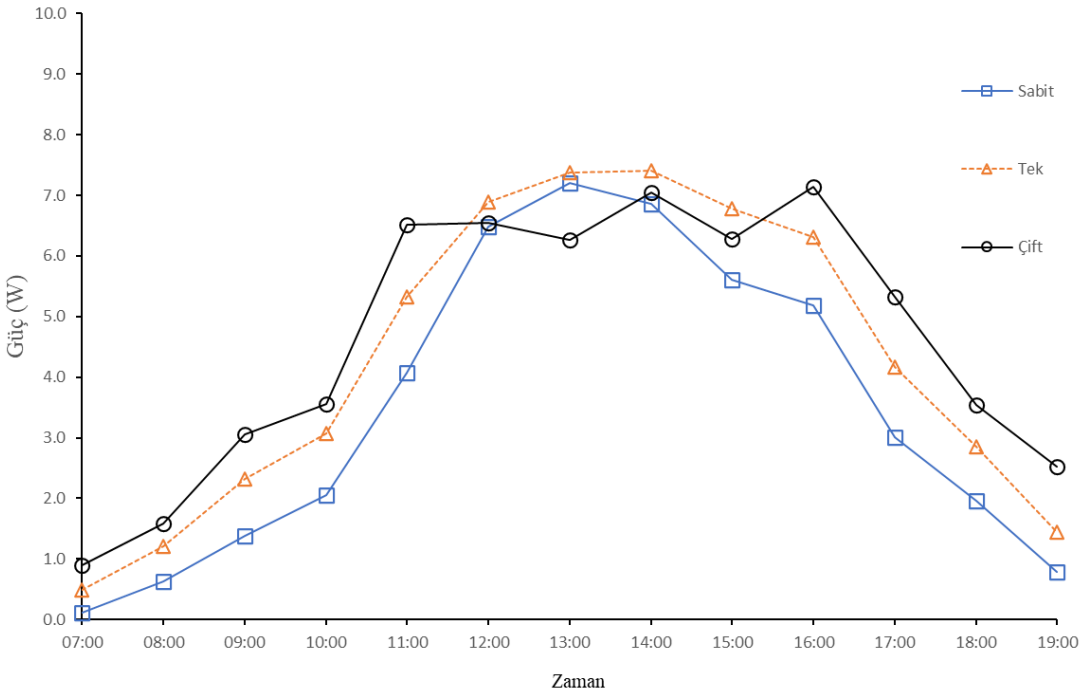
	Gerilim (V)			Akım (A)			Güç(W)		
Zaman	Sabit	Tek	Çift	Sabit	Tek	Çift	Sabit	Tek	Çift
07:00	1.801	3.746	5.097	0.062	0.130	0.176	0.112	0.486	0.899
08:00	4.278	5.896	6.755	0.148	0.204	0.234	0.633	1.203	1.579
09:00	6.305	8.188	9.396	0.218	0.283	0.325	1.376	2.320	3.055
10:00	7.697	9.417	10.133	0.266	0.326	0.351	2.050	3.069	3.553
11:00	10.850	12.406	13.716	0.375	0.429	0.475	4.073	5.326	6.510
12:00	13.695	14.105	13.757	0.474	0.488	0.476	6.490	6.884	6.549
13:00	14.432	14.596	13.450	0.499	0.505	0.465	7.207	7.372	6.260
14:00	14.084	14.637	14.268	0.487	0.506	0.494	6.864	7.413	7.044
15:00	12.733	14.002	13.470	0.441	0.484	0.466	5.610	6.784	6.278
16:00	12.242	13.511	14.371	0.424	0.468	0.497	5.186	6.317	7.146
17:00	9.335	10.973	12.406	0.323	0.380	0.429	3.015	4.166	5.326
18:00	7.513	9.089	10.113	0.260	0.314	0.350	1.953	2.858	3.539
19:00	4.749	6.448	8.536	0.164	0.223	0.295	0.780	1.439	2.521



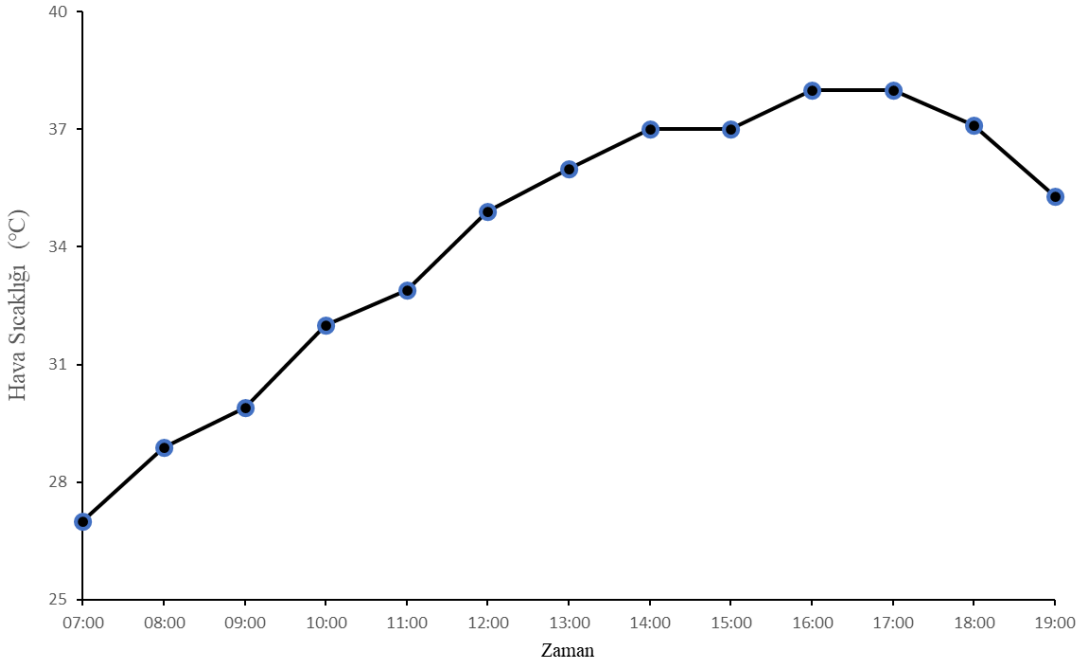
Şekil 4.7. 16 Ağustos deneysel verileri gerilim grafiği



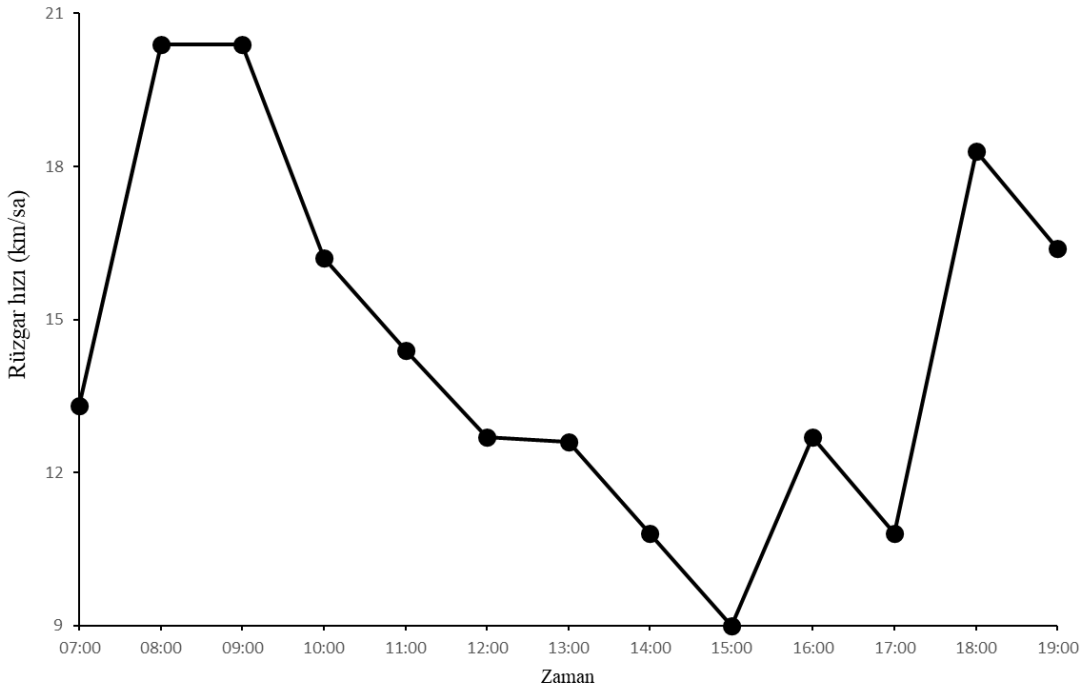
Şekil 4.8. 16 Ağustos deneysel verileri akım grafiği



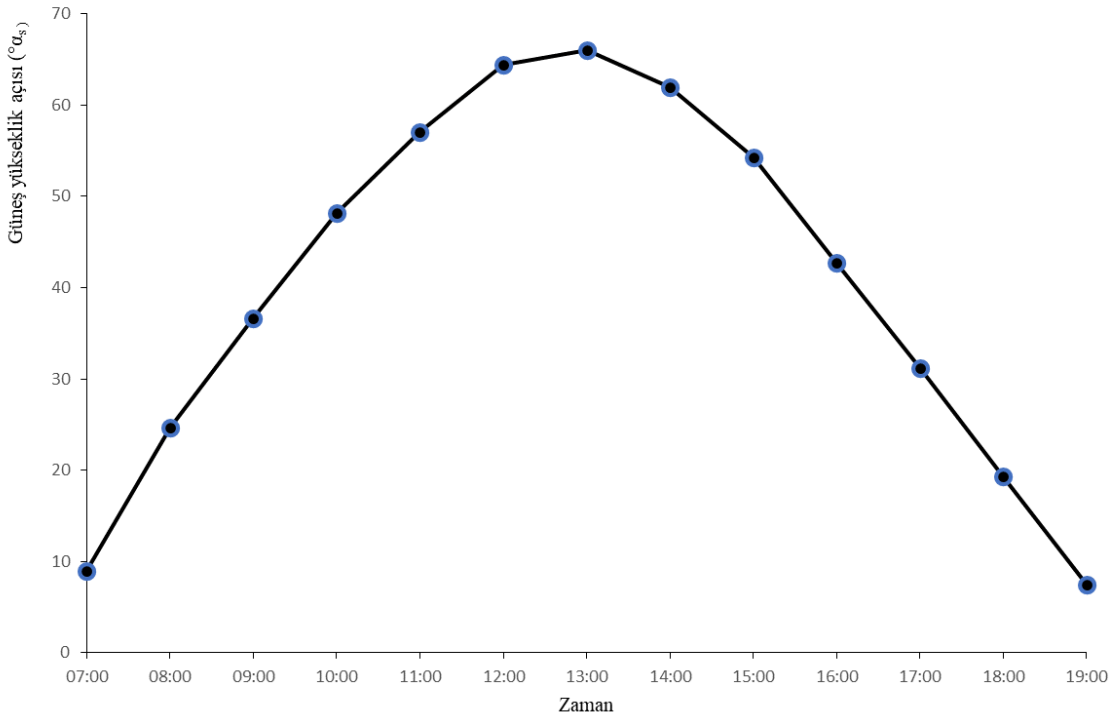
Şekil 4.9. 16 Ağustos deneysel verileri güç grafiği



Şekil 4.10. 16 Ağustos hava sıcaklığı grafiği (Spark, 2023)



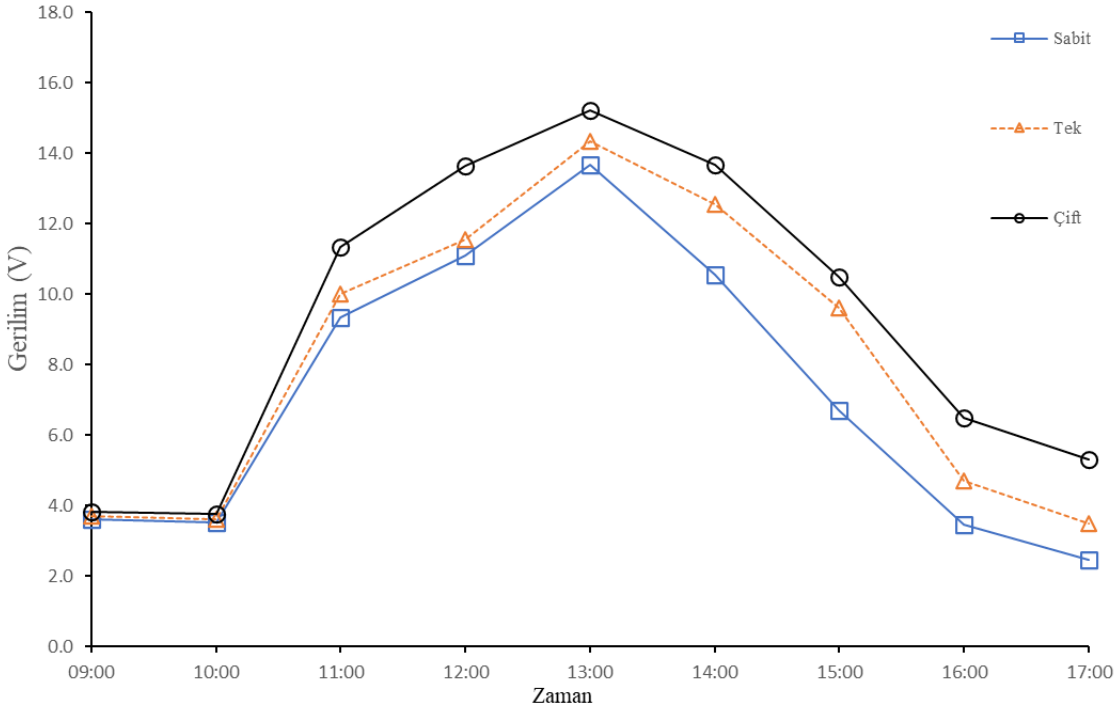
Şekil 4.11. 16 Ağustos rüzgâr hızı grafiği (Spark, 2023)



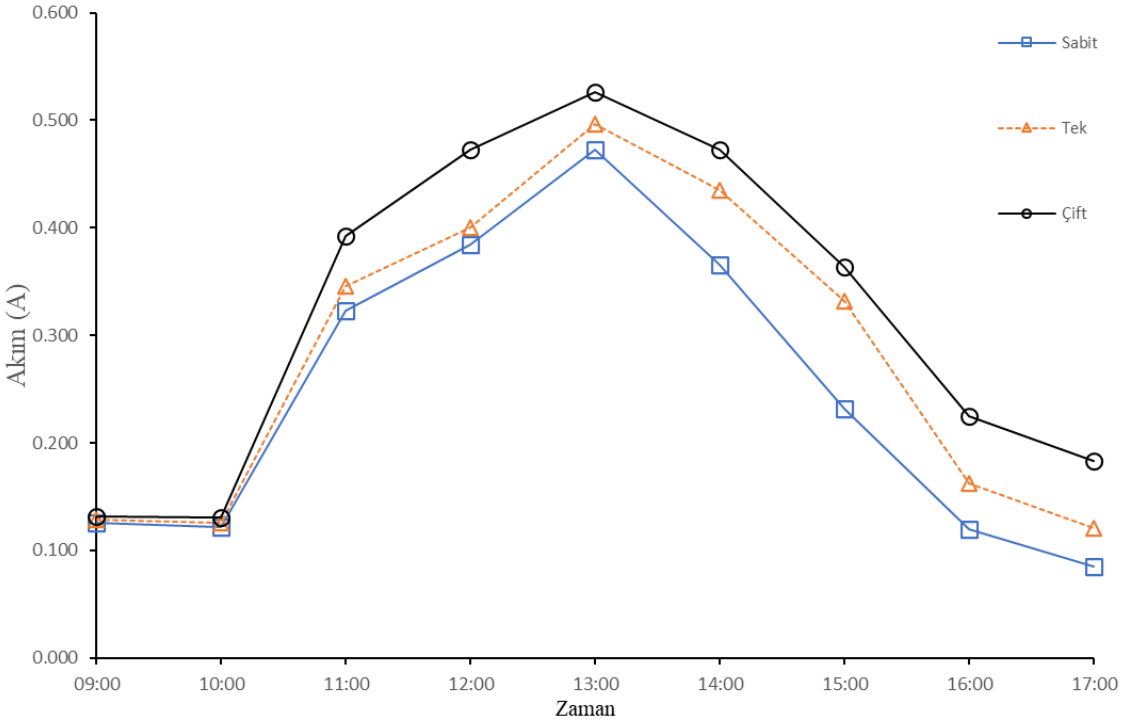
Şekil 4.12. 16 Ağustos güneşin yükseklik açısı grafiği (Spark, 2023)

Çizelge 4.3. 6 Ocak arasında yapılan ölçümler

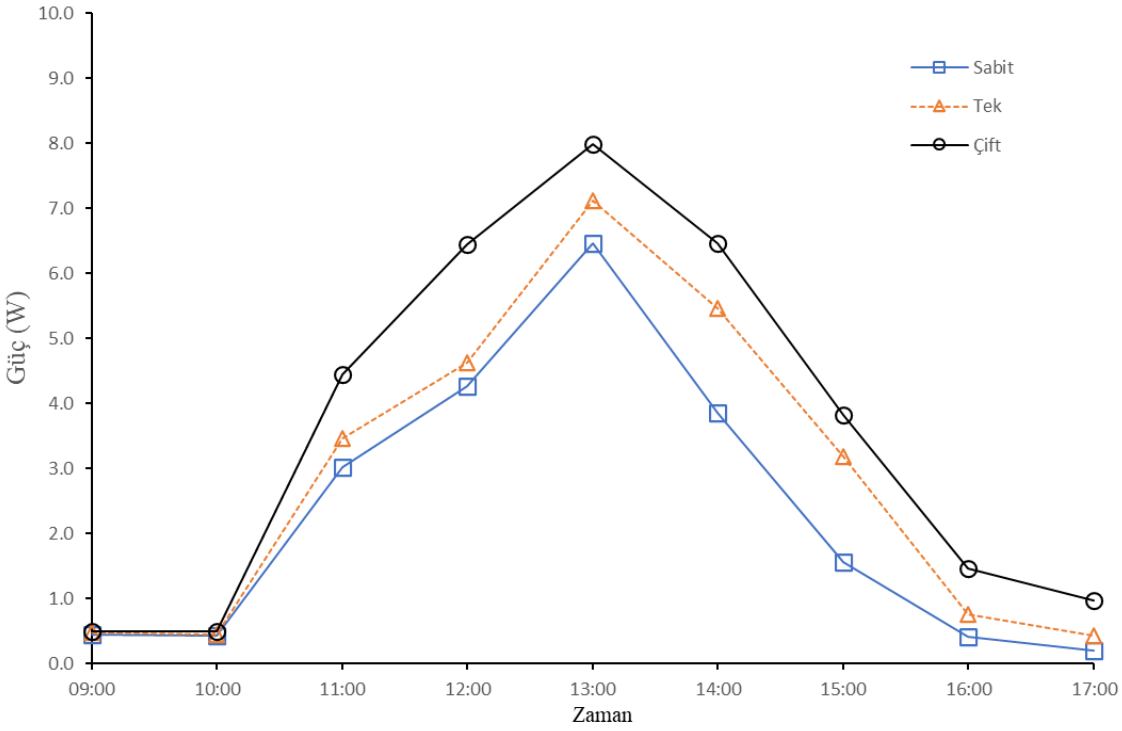
	Gerilim (V)			Akım (A)			Güç(W)		
Zaman	Sabit	Tek	Çift	Sabit	Tek	Çift	Sabit	Tek	Çift
09:00	3.62	3.71	3.812	0.125	0.128	0.132	0.453	0.476	0.503
10:00	3.52	3.62	3.77	0.122	0.125	0.130	0.429	0.453	0.492
11:00	9.34	10	11.34	0.323	0.346	0.392	3.019	3.460	4.450
12:00	11.1	11.56	13.65	0.384	0.400	0.472	4.263	4.624	6.447
13:00	13.67	14.34	15.2	0.473	0.496	0.526	6.466	7.115	7.994
14:00	10.56	12.56	13.67	0.365	0.435	0.473	3.859	5.459	6.466
15:00	6.7	9.6	10.5	0.232	0.332	0.363	1.553	3.189	3.815
16:00	3.45	4.7	6.5	0.119	0.163	0.225	0.412	0.764	1.462
17:00	2.45	3.5	5.3	0.085	0.121	0.183	0.208	0.424	0.972



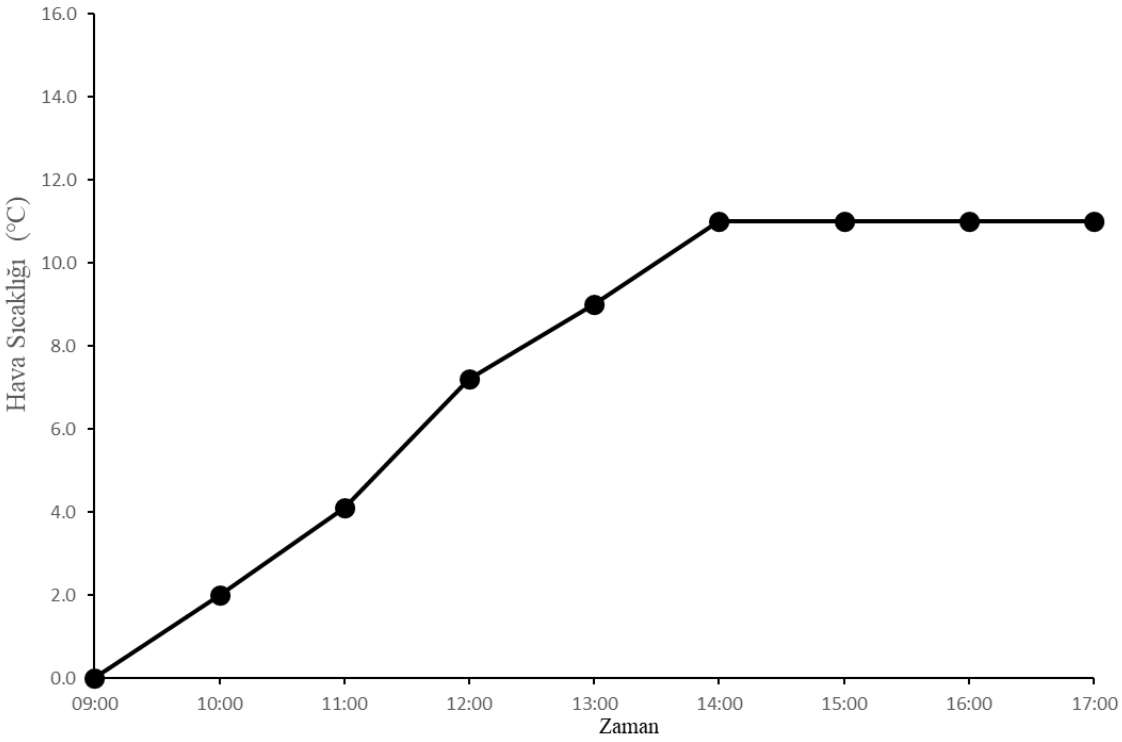
Şekil 4.13. 6 Ocak deneysel verileri gerilim grafiği



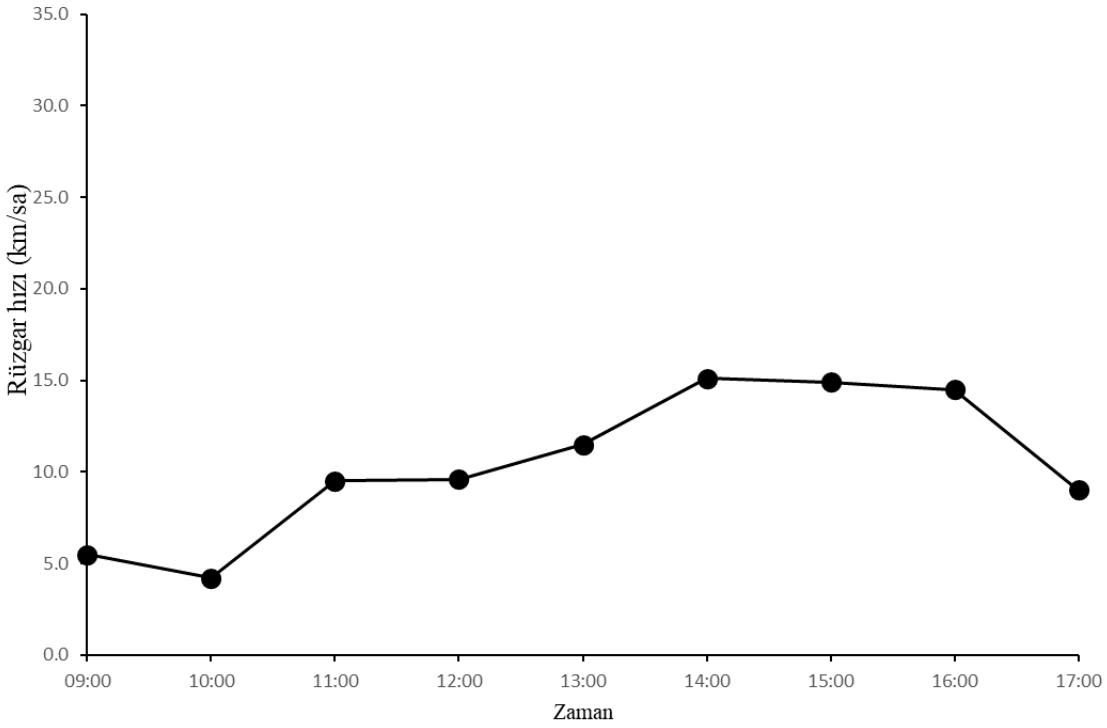
Şekil 4.14. 6 Ocak deneysel verileri akım grafiği



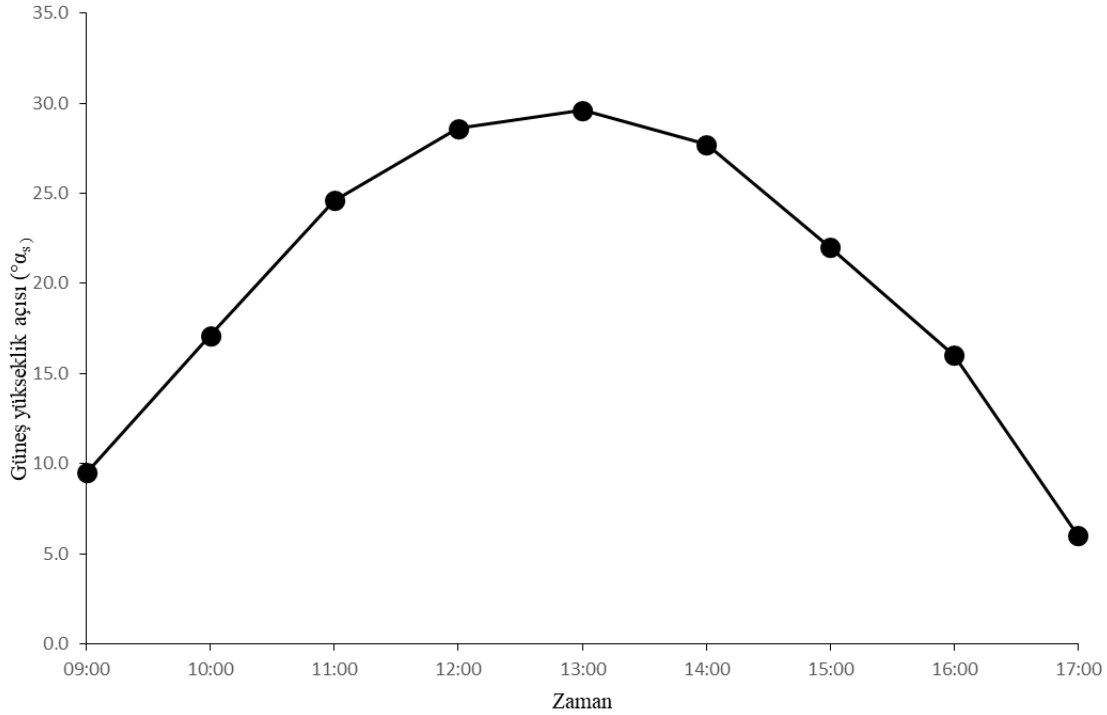
Şekil 4.15. 6 Ocak 2024 deneysel verileri güç grafiği



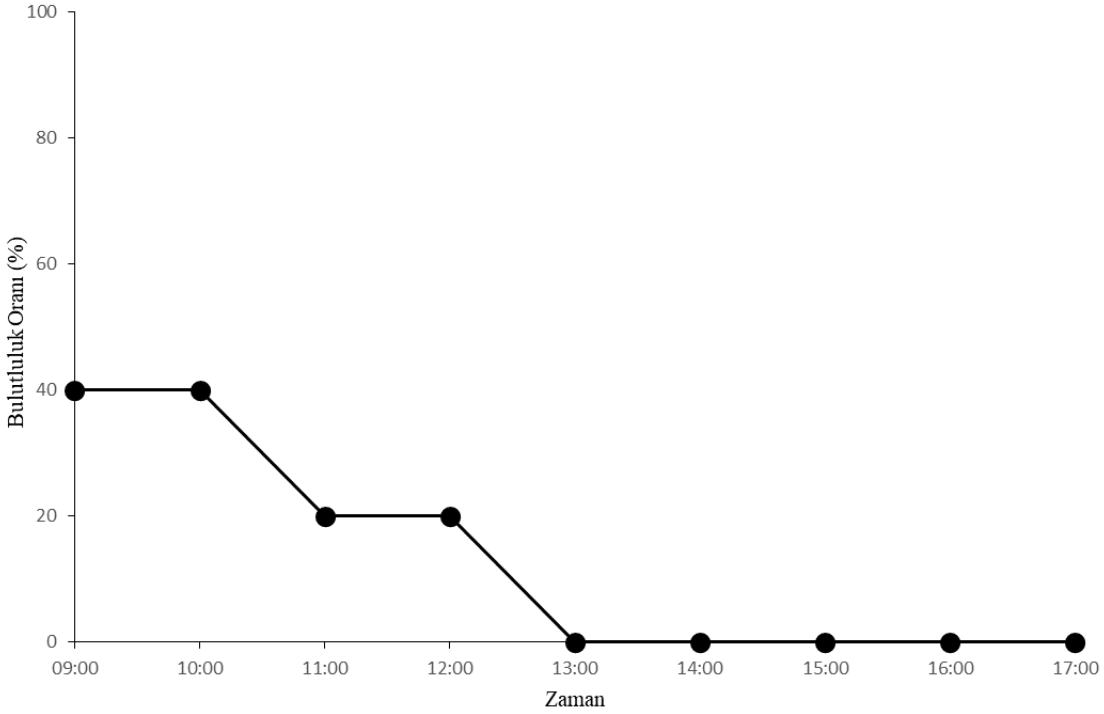
Şekil 4.16. 6 Ocak hava sıcaklığı grafiği (Spark, 2023)



Şekil 4.17. 6 Ocak rüzgâr hızı grafiği (Spark, 2023)



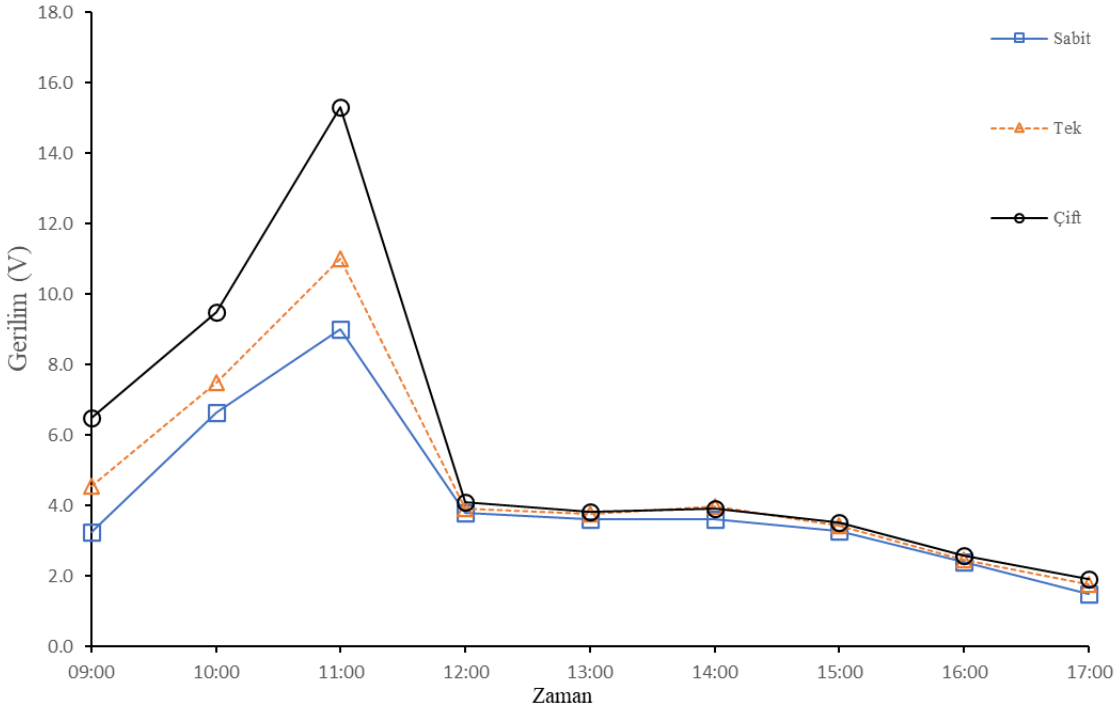
Şekil 4.18. 6 Ocak güneşin yükseklik açısı grafiği (Spark, 2023)



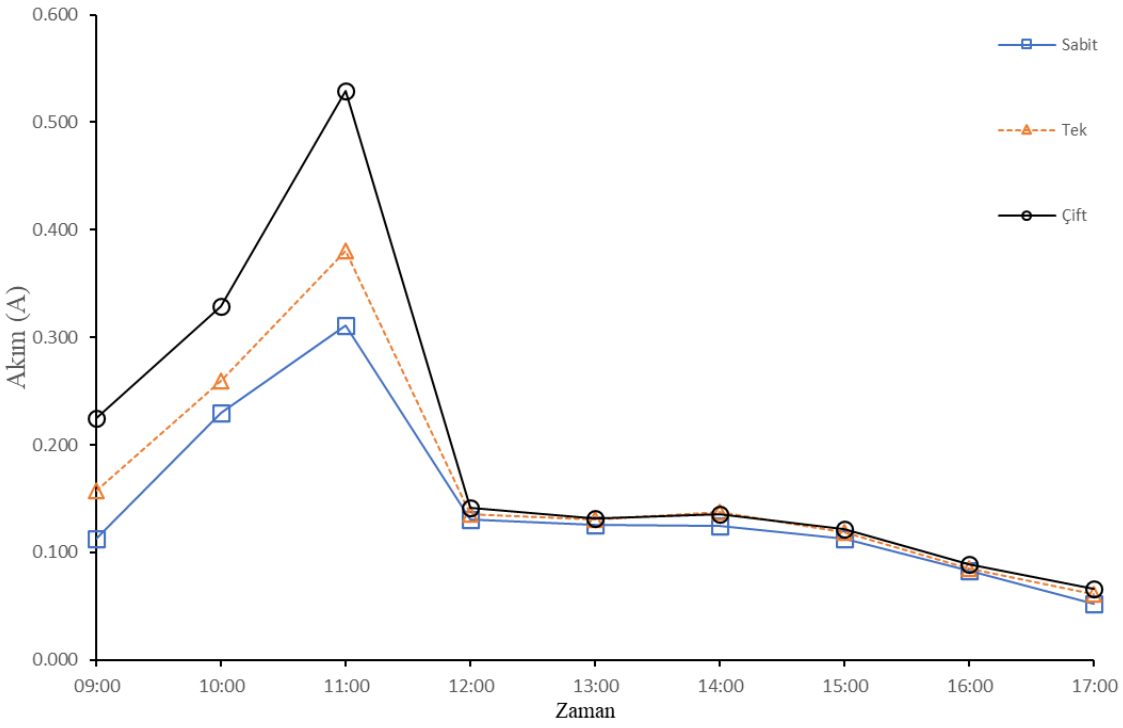
Şekil 4.19. 6 Ocak bulutluluk oranı grafiği (Spark, 2023)

Çizelge 4.4.7 Ocak arasında yapılan ölçümler

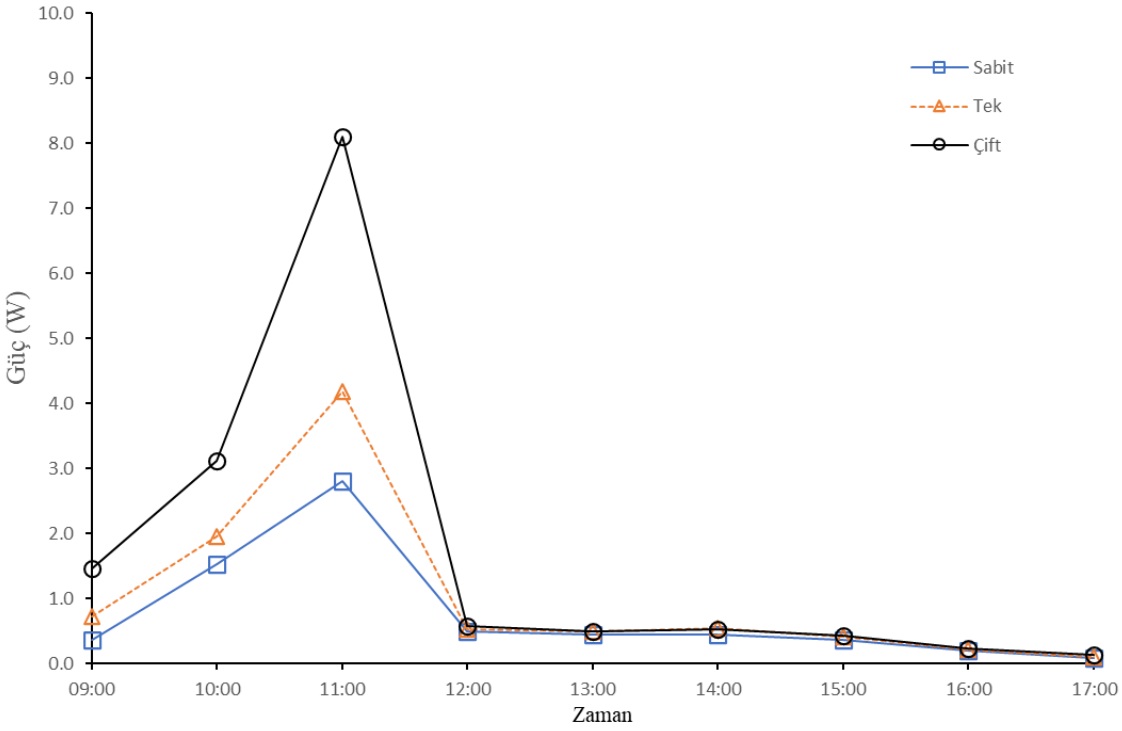
	Gerilim (V)			Akım (A)			Güç(W)		
Zaman	Sabit	Tek	Çift	Sabit	Tek	Çift	Sabit	Tek	Çift
09:00	3.25	4.56	6.5	0.112	0.158	0.225	0.365	0.720	1.462
10:00	6.64	7.5	9.5	0.230	0.260	0.329	1.526	1.946	3.123
11:00	9	11	15.3	0.311	0.381	0.529	2.803	4.187	8.100
12:00	3.78	3.91	4.1	0.131	0.135	0.142	0.494	0.529	0.582
13:00	3.62	3.77	3.81	0.125	0.130	0.132	0.453	0.492	0.502
14:00	3.61	3.98	3.92	0.125	0.138	0.136	0.451	0.548	0.532
15:00	3.26	3.43	3.51	0.113	0.119	0.121	0.368	0.407	0.426
16:00	2.4	2.46	2.58	0.083	0.085	0.089	0.199	0.209	0.230
17:00	1.5	1.76	1.92	0.052	0.061	0.066	0.078	0.107	0.128



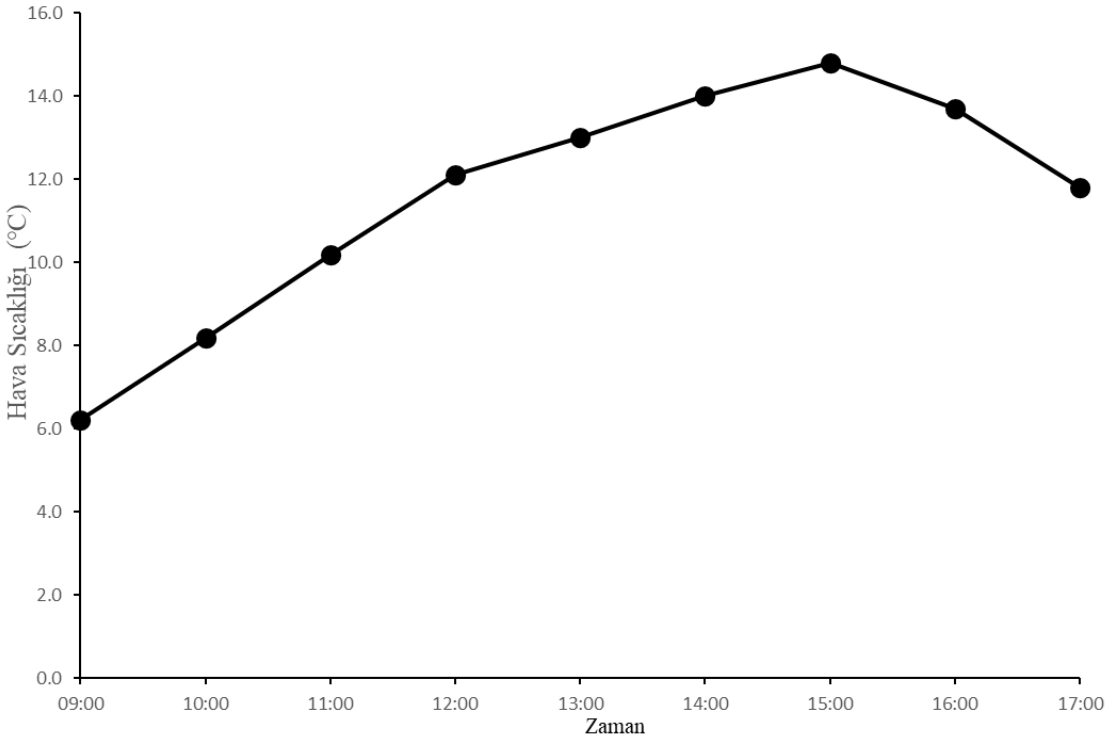
Şekil 4.20. 7 Ocak deneysel verileri Gerilim grafiği



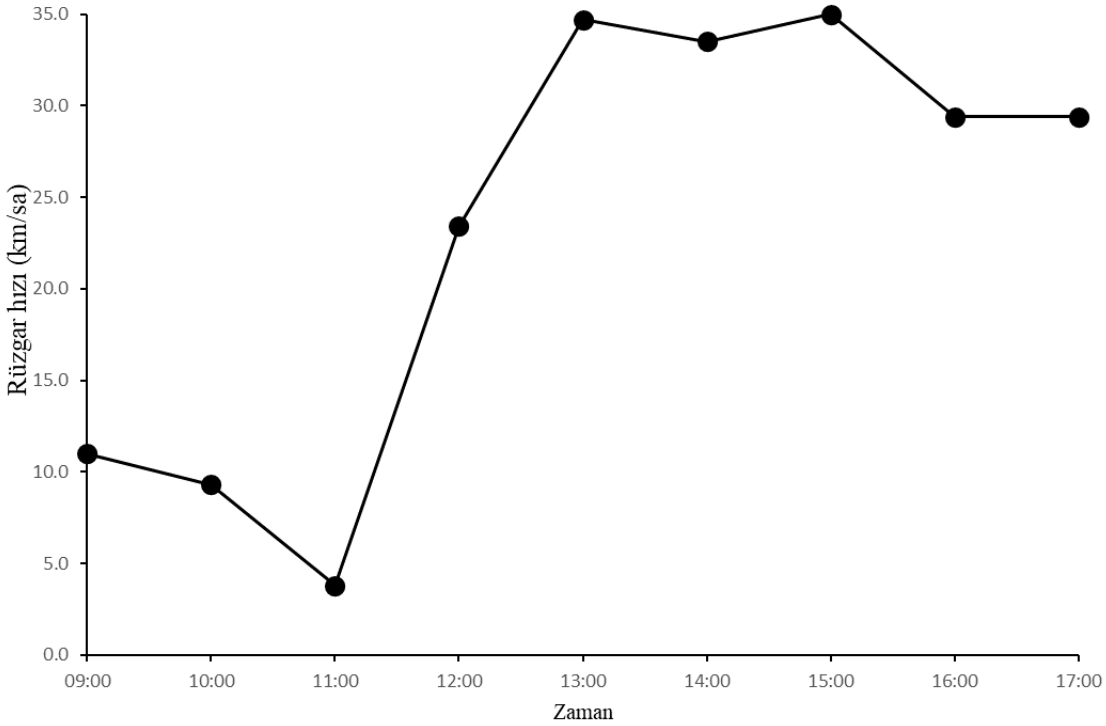
Şekil 4.21. 7 Ocak deneysel verileri akım grafiği



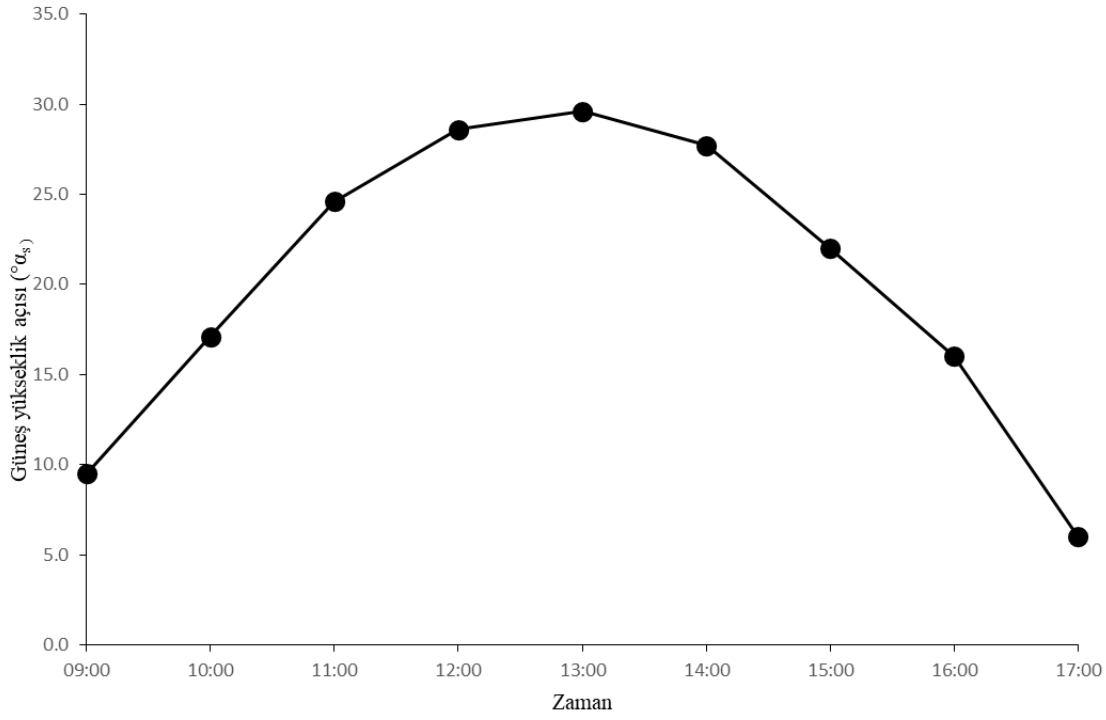
Şekil 4.22. 7 Ocak deneysel verileri güç grafiği



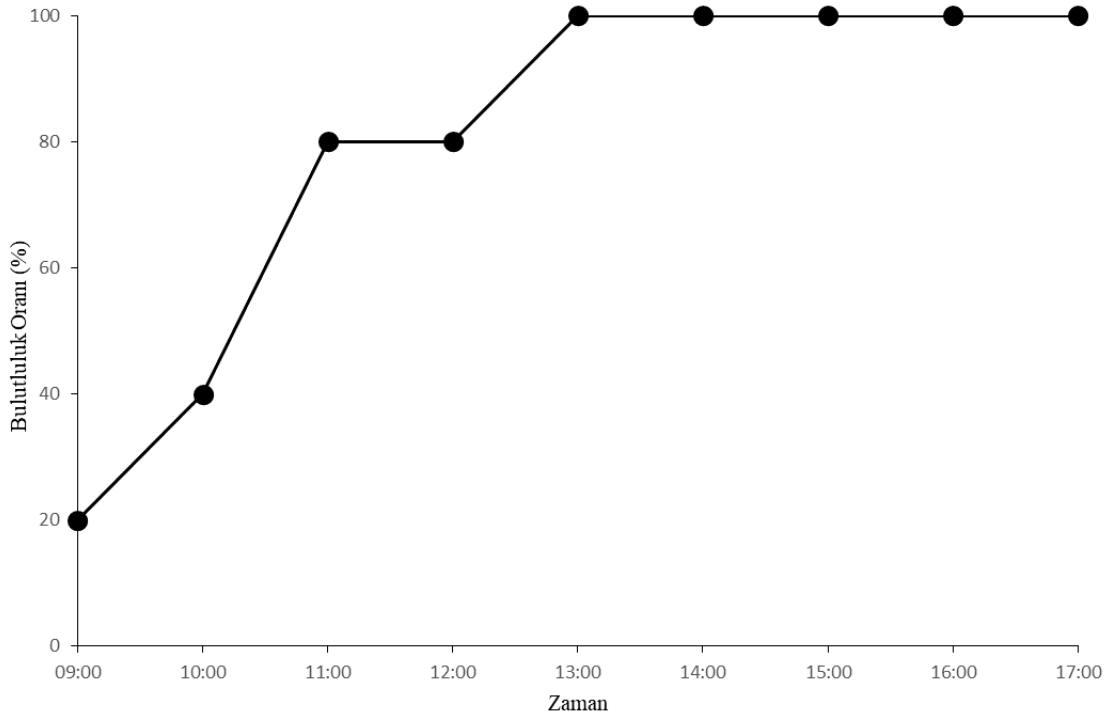
Şekil 4.23. 7 Ocak hava sıcaklığı grafiği (Spark, 2023)



Şekil 4.24. 7 Ocak rüzgâr hızı grafiği (Spark, 2023)



Şekil 4.25. 7 Ocak güneşin yükseklik açısı grafiği (Spark, 2023)



Şekil 4.26. 7 Ocak bulutluluk oranı grafiği (Spark, 2023)

4.2. Tartışma

Tek eksenli, çift eksenli ve sabit açılı güneş takip sistemlerinin verim karşılaştırılmasının yapılması için her bir panelin Temmuz ve Ağustos ayında günlük ortalama ürettikleri W saat enerji hesaplaması, sistemde kullanılan otomatik ölçüm sisteminin etkisinin görülmesi için 2 farklı şekilde hesaplanmıştır. Öncelikle ölçümlerin saatlik olduğu varsayıldığı taktirde elde edilecek olan günlük toplam enerji değeri hesaplanmıştır.

Temmuz ayında çift eksenli güneş takip sistemi, saatlik alınan ortalama ölçümlere bakıldığında toplam 68.042 W saat enerji ürettiği görülmüştür. Tek eksenli güneş takip sistemli panel 60.325 Wh ve sabit açılı panel 49.13 Wh enerji üretmiştir. Bu değerler karşılaştırıldığında, çift eksenli güneş takip sisteminin 20 Temmuz arasında ortalama enerji üretiminin sabit açılı panele oranla %38.5 daha yüksek olduğu görülmüştür. Tek

eksenli güneş takip sisteminin sabit açılı sisteme oranla %22.8 daha yüksek verime sahip olduğu görülmüştür. Çift eksenli güneş takip sistemi ile tek eksenli takip sistemi arasında ise, %12.8 verim farkı vardır.

Bu değerler ölçüm sistemi 1 saat arayla ölçüm yaptığı varsayıldığı zaman hesaplanan değerlerdir, şimdide sisteminin gerçekte olan 15 dakikalık aralıklarda yaptığı ölçümler hesaplandığında çift eksenli güneş takip sistemli panelin günlük toplam enerji üretiminin, 64 Wh olduğu görülmüştür. Tek eksenli güneş takip sistemli panelin 56.62 Wh ve sabit açılı panelin 45.87 Wh enerji üretimlerinin olduğu hesaplanmıştır.

Bu sonuçtan yola çıkıldığında, panellerin güç çıkışlarının ölçüm sıklığı artığında ölçülen değerin düştüğü anlaşabilir, fakat bu değer gerçekte olan değere çok daha yakındır. Ayrıca 15 dakikalık aralıklarla yapılan ölçümlerle hesaplanan günlük toplam enerji üretim değerleri, öncesinde yapıldığı gibi güneş takip sistemleri arasındaki verim farkının hesabı yapılırsa, çift eksenli güneş takip sisteminin 20 Temmuz tarihinde enerji üretiminin sabit açılı panele oranla %39.5 daha yüksek olduğu görülmüştür. Tek eksenli güneş takip sisteminin sabit açılı sisteme oranla %23.5 daha yüksek verime sahip olduğu görülmüştür. Çift eksenli güneş takip sistemi ile tek eksenli takip sistemi arasında ise, %13 verim farkı vardır. Yani elde edilen toplam enerji üretim değerleri daha düşük olsa bile, ölçüm sıklığı artığında güneş takip sisteminin sağladığı verim artışı daha belirgin hale gelmektedir.

Bu işlemler 16 Ağustos'ta yapılan ortalama panel güç ölçüm değerleri için yapıldığında, Ağustos ayında çift eksenli güneş takip sistemi, saatlik alınan ortalama ölçümlere bakıldığında toplam çift eksenli güneş takip sistemli panelin günlük toplam enerji üretiminin, 60.26 Wh olduğu görülmüştür. Tek eksenli güneş takip sistemli panelin 55.63 Wh ve sabit açılı panelin 45.35 Wh enerji üretimlerinin olduğu hesaplanmıştır. Bu değerler karşılaştırıldığında, çift eksenli güneş takip sisteminin 16 Ağustos tarihinde enerji üretiminin sabit açılı panele oranla %32.9 daha yüksek olduğu görülmüştür. Tek eksenli güneş takip sisteminin sabit açılı sisteme oranla %22.6 daha yüksek verime sahip olduğu görülmüştür. Çift eksenli güneş takip sistemi ile tek eksenli takip sistemi arasında ise, %8.3 verim farkı vardır.

Şimdide sisteminin gerçekte olan 15 dakikalık aralıklarda yaptığı ölçümler hesaplandığında çift eksenli güneş takip sistemli panelin günlük toplam enerji üretiminin, 57.68 Wh olduğu görülmüştür. Tek eksenli güneş takip sistemli panelin 51.68 Wh ve sabit açılı panelin 42.89 Wh enerji üretimlerinin olduğu görülmüştür. Bu değerler

karşılaştırıldığında, çift eksenli güneş takip sisteminin 16 Ağustos tarihinde enerji üretiminin sabit açılı panele oranla %34.5 daha yüksek olduğu görülmüştür. Tek eksenli güneş takip sisteminin sabit açılı sisteme oranla %21 daha yüksek verime sahip olduğu görülmüştür. Çift eksenli güneş takip sistemi ile tek eksenli takip sistemi arasında ise, %11.2 verim farkı vardır.

Temmuz ayında yapılan ölçümlerin ortalaması ile Ağustos ayında yapılan ölçümlerin ortalaması göz önüne alındığında, Ağustos ayında Temmuz ayına oranla üretilen toplam enerji ile birlikte, çift ve tek eksenli güneş takip sistemlerinin sabit açılı panele oranla sahip oldukları verim artışının da düştüğü gözlenmiştir. Fakat hem Temmuz hem de Ağustos ayı boyunca yapılan ölçümlerin farklılık göstermesine rağmen, iki durum içinde 1 saat aralıklı yapılan ölçümlerin 15 dakika aralıklı yapılan ölçümlere oranla daha yüksek günlük toplam enerji çıkışı görülmesine rağmen çift ve tek eksenli güneş takip sistemli panellerin sabit açılı panele oranla sahip oldukları verim artışının pozitif yönlü olması, Temmuz ayındaki hesaplar sonucunda düşünülen ölçüm sıklığının bu duruma etkisi fikrini desteklemektedir.

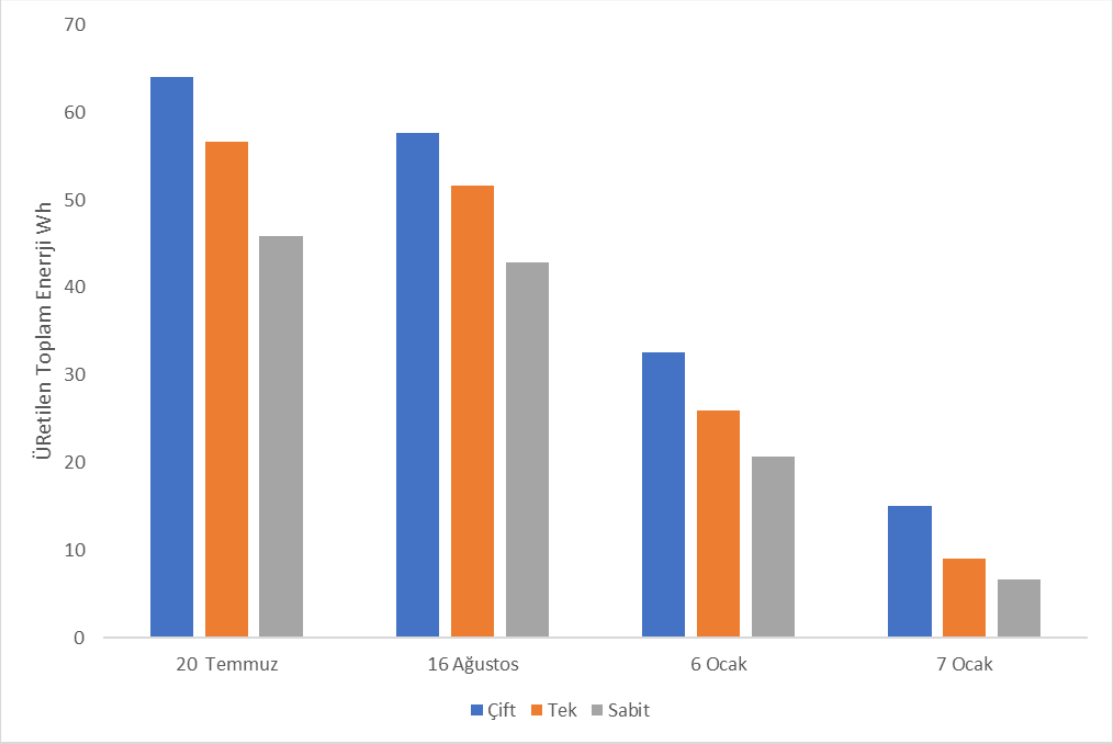
Temmuz ile Ağustos aylarında yapılan ölçümlerden elde edilen sonuçlara yola çıkarak, Temmuz ayında Ağustos ayına oranla daha yüksek bir günlük toplam enerji üretimi görülmesi, şekil 4.6 ile şekil 4.12'de görülen güneşin yükseklik açısı grafiklerine bakıldığında ölçümlerinin beklenen değerlerle uyuştuğu görülmektedir.

Temmuz ayındaki daha uzun güneşlenme süreleri ve daha yüksek güneş yükseklik açısı, panellerin daha yüksek enerji üretimi yapmalarını sağlamıştır. Ayrıca Temmuz ve Ağustos ayları arasındaki ölçümlerden elde edilen sonuçlar sonucunda güneş takip sistemli panellerin, Temmuz ayına oranla sabit açılı panele olan verim artışındaki azalmasını etkileyen faktörlerden biri, Ağustos ayının Temmuz ayına oranla daha yüksek hava sıcaklıklarına sahip olması söz konusu olabilir. Şayet Şekil 4.4 ve Şekil 4.10'da bakıldığında, saat 13:00 için 20 Temmuz'da 29 °C sıcaklık görülürken 16 Ağustos tarihinde aynı saatte sıcaklığın 37 °C olması, takip sistemli panellerin aşırı ısınmasına ve sabit sistemin kısa bir süre için en yüksek verime sahip olmasına neden olmuş olabilir.

6 Ocak yapılan ölçümlerde güneş takip sisteminin havanın bulutlu ve açık olması durumlarındaki güç çıkışları ve verim artışları incelemiştir. Yapılan ölçümlerin sonuçları hesaplandığında çift eksenli güneş takip sistemli panelin günlük toplam enerji üretiminin, 32.60 Wh olduğu görülmüştür. Tek eksenli güneş takip sistemli panelin 25.97 Wh ve sabit açılı panelin 20.66 Wh enerji üretimlerinin olduğu görülmüştür. Bu değerler

karşılaştırıldığında, çift eksenli güneş takip sisteminin 6 Ocak tarihinde enerji üretiminin sabit açılı panele oranla %36.6 daha yüksek olduğu görülmüştür. Tek eksenli güneş takip sisteminin sabit açılı sisteme oranla %20 daha yüksek verime sahip olduğu görülmüştür. Çift eksenli güneş takip sistemi ile tek eksenli takip sistemi arasında ise, %20 verim farkı vardır.

7 Ocak arasında yapılan ölçümlerin sonuçları hesaplandığında çift eksenli güneş takip sistemli panelin günlük toplam enerji üretiminin, 15.1 Wh olduğu görülmüştür. Tek eksenli güneş takip sistemli panelin 9.1 Wh ve sabit açılı panelin 6.74 Wh enerji üretimlerinin olduğu görülmüştür. Bu değerler karşılaştırıldığında, çift eksenli güneş takip sisteminin 7 Ocak tarihinde enerji üretiminin sabit açılı panele oranla %53.3 daha yüksek olduğu görülmüştür. Tek eksenli güneş takip sisteminin sabit açılı sisteme oranla %26.3 daha yüksek verime sahip olduğu görülmüştür. Çift eksenli güneş takip sistemi ile tek eksenli takip sistemi arasında ise, %39.4 verim farkı vardır.



Şekil 4.27. Panellerin Toplam Enerji Üretimleri

5. SONUÇLAR VE ÖNERİLER

5.1. Sonuçlar

20 Temmuz'da fotovoltaik panellerin güç çıkışı W olarak ölçülmüş ve Matlab programında güç saate dönüştürülmüştür. Hesapların sonucunda, 1 saatlik ölçüm aralıkları için çift eksenli sistemin 68.042 W saat, tek eksenli sistemin 60.325 Wh ve sabit açılı panel 49.13 Wh enerji üretmiştir. 15 dakikalık ara ile yapılan ölçümlerde ise bu değerler, çift eksenli sistemin 64 W saat, tek eksenli sistemin 56.62 Wh ve sabit açılı panel 45.87 Wh enerji üretmiştir.

Hesaplanan iki durum için güneş takip sistemlerinin verimleri, 1 saatlik ölçüm aralıkları için çift eksenli sistemin sabit açılı panele oranla %38.5 tek eksenli takip sistemi ile ise arasında ise %12.8 ayrıca tek eksenli güneş takip sisteminin sabit açılı sisteme oranla %22.8 fazla verime sahip olduğu görülmüştür. 15 dakikalık ara ile yapılan ölçümlerde ise bu değerler, çift eksenli sistemin sabit açılı panele oranla %39.5 tek eksenli takip sistemi ile ise arasında ise %13 ayrıca tek eksenli güneş takip sisteminin sabit açılı sisteme oranla %23.5 fazla verime sahip olduğu görülmüştür.

16 Ağustos'ta fotovoltaik panellerin güç çıkışı W olarak ölçülmüş ve Matlab programında güç saate dönüştürülmüştür. Hesapların sonucunda, 1 saatlik ölçüm aralıkları için çift eksenli sistemin 60.26 W saat, tek eksenli sistemin 55.63 Wh ve sabit açılı panel 45.33 Wh enerji üretmiştir. 15 dakikalık ara ile yapılan ölçümlerde ise bu değerler, çift eksenli sistemin 57.68 W saat, tek eksenli sistemin 51.68 Wh ve sabit açılı panel 42.89 Wh enerji üretmiştir.

Hesaplanan iki durum için güneş takip sistemlerinin verimleri, 1 saatlik ölçüm aralıkları için çift eksenli sistemin sabit açılı panele oranla %32.9 tek eksenli takip sistemi ile ise arasında ise %8.3 ayrıca tek eksenli güneş takip sisteminin sabit açılı sisteme oranla %22.6 fazla verime sahip olduğu görülmüştür. 15 dakikalık ara ile yapılan ölçümlerde ise bu değerler, çift eksenli sistemin sabit açılı panele oranla % 34.5 tek eksenli takip sistemi ile ise arasında ise %11.2 ayrıca tek eksenli güneş takip sisteminin sabit açılı sisteme oranla %21 fazla verime sahip olduğu görülmüştür.

20 Temmuz ve 16 Ağustos'ta yapılan ölçümlerin farklılık göstermesine rağmen, iki durum içinde 1 saat aralıklı yapılan ölçümlerin 15 dakika aralıklı yapılan ölçümlere oranla daha yüksek günlük toplam enerji çıkışı görülmesine rağmen çift ve tek eksenli

güneş takip sistemli panellerin sabit açılı panele oranla sahip oldukları verim artışının pozitif yönlü olması, 20 Temmuz'daki hesaplar sonucunda düşünülen ölçüm sıklığının bu duruma etkisi fikri ön sürülmüştür.

20 Temmuz'da yapılan ölçümlerin ortalaması ile 16 Ağustos'ta yapılan ölçümler göz önüne alındığında, Ağustos ayında Temmuz ayına oranla üretilen toplam enerji ile birlikte, çift ve tek eksenli güneş takip sistemlerinin sabit açılı panele oranla sahip oldukları verim artışının da düştüğü gözlenmiştir. Toplam günlük enerji üretiminde görülen bu düşüşün sebebinin Temmuz ayı ile Ağustos ayı arasındaki güneşlenme süreleri ve güneşin yükseklik açısının değişimi olduğu düşünülmüş ve bu fikir gösterilen güneş yükseklik açıları ile desteklenmiştir.

Temmuz ve Ağustos ayları arasındaki güneş takip sisteminin sabit sisteme olan verim artışının azalmasını etkileyen faktörlerden biri, Ağustos ayının Temmuz ayına oranla daha yüksek hava sıcaklıklarına sahip olduğu düşünülmüştür ve bu fikir verilen hava sıcakları grafiği ile desteklenmiştir.

Yapılan tüm sonuçlara bakıldığında, güneş takip sisteminin saat 12:00 ile 14:00 sırasında, sabit açılı sisteme oranla belirgin bir verim artışı sağlamadığı gözlenmiştir, bunun sebebi bu saatlerdeki azimut ve yükseklik açılarının, sabit açılı panelin açısına yakın olmasıdır. Fakat özellikle saat 07:00-10:00 ve 16:00-19:00 arasında çift eksenli güneş takip sisteminin sabit açılı sisteme oranla %200-250 fazla verime dahi sahip olduğu görülmüştür. Tek eksenli güneş takip sisteminin ise sabit açılı sisteme oranla %150-200 fazla verime sahip olduğu görülmüştür.

Güneş takip sistemlerinin özellikle bu saatler sırasında sahip olduğu bu verim artışı, oldukça önemlidir. Çünkü bilindiği üzere fotovoltaik güç santrallerinin bağlı oldukları şebekeye anlık verebilecekleri gücün miktarı oldukça sınırlıdır ve bu limitin aşılması durumda şebekenin zarar görmesi ve bu masrafların ödenebilmesi için limit aşımı yapan sistemlere yüklü bir ceza kesilmektedir. Güneş takip sistemi kullanılmasının bu konudaki avantajı ise santralin şebekeye sadece saat 11:00 ile 14:00 arasında yüksek güç vermesi ve limit aşımı riski alması yerine, kullanılan fotovoltaik panellerin sayısı azaltılabilir ve yerine kullanılan güneş takip sistemi, şebekeye daha düzenli ve geniş saatler arasında güç sağlayabilir. Bu sayede hem kurulu şebeke gereksiz ani güç artışlarına maruz kalmaz ve aynı zamanda şebekeye satılan toplam enerji miktarının miktarı artırılmış olur.

Daha önce yapılmış olan bazı araştırmaların sonucunda düşünülen bir sorun güneş takip sistemlerinde kullanılan motorun gereksiz güç harcamasıdır. Yani panelin güç çıkışında bir artış gözlenmiş olsa bile bazı araştırma sonuçlarında motorların güneş takip sistemi için kullandığı enerji bu verim artışını önemli ölçüde etkilediği gözlenmiştir.

Yaptığım bu çalışmada özellikle bu konudaki araştırmalarda kullanılan adım motorların yüksek güç tüketimleri olduğunu kanıtlamak amacıyla oldukça yaygın ve ucuz bir motor tercih edilmiştir. Kullanılan bu motorla beraber eklenen yüksek çevrim oranlı bir dişli sistemi kullanılmıştır, bu sayede motordan elde edilen tork önemli bir düzeyde artmıştır.

Yapılan hesaplamalarda, tek eksenli sabit sisteminde kullanılan 6 V 0.2 A redüktörlü DC motorun sadece günde 10-15 dakika çalıştığı gözlenmiştir. Yani sadece 10 W gücünde bir güneş panelinin günlük toplam enerji üretimini sabit açılı panele oranla 8-10 Wh artıran bu sistem sadece 0.2-0.5 Wh enerjiyi motor için kullanmaktadır. İki eksenli güneş takip sisteminde ise motorların günlük ortalama 0.6-1 Wh enerji harcadığı hesaplanmıştır. Fakat bunun karşılında çift eksenli güneş takip sistemli panel, sabit açılı sisteme oranla günlük ortalama 12-15 Wh fazla enerji ürettiği görülmüştür. Özetle hem çift eksenli hem de tek eksenli güneş takip sistemlerinde motorların tükettiği enerji, panelde elde edilen ekstra enerjinin %8-10'nunun üzerine çıkmadığı gözlemlenmiştir. Ayrıca bu araştırmada kullanılan dişli ve motor sisteminin, büyük çaplı güneş takip sistemi projelerine oranla oldukça basit olduğu düşünüldüğünde, motorların tükettiği enerji neredeyse ihmal edilebilecek hale gelmektedir.

6 Ocak'ta gerçekleştirilen ölçümler, güneş takip sistemlerinin bulutlu ve açık hava koşullarındaki güç çıkışlarını ve verim artışlarını inceledi. Yapılan ölçümlerin analizi sonucunda, çift eksenli güneş takip sistemine sahip panellerin günlük toplam enerji üretiminin 32.60 Wh olduğu belirlendi. Tek eksenli güneş takip sistemine sahip panellerin enerji üretimi 25.97 Wh iken, sabit açılı panellerin enerji üretimi 20.66 Wh olarak ölçüldü. Bu sonuçlar gösteriyor ki, 6 Ocak tarihinde çift eksenli güneş takip sistemi, enerji üretimi açısından sabit açılı panele göre %36.6 daha yüksek performans sergiledi. Aynı şekilde, tek eksenli güneş takip sistemi, sabit açılı sistemle karşılaştırıldığında %20 daha yüksek bir verime sahipti. Çift eksenli ve tek eksenli güneş takip sistemleri arasında ise %20'lik bir verim farkı gözlemlendi.

7 Ocak tarihinde gerçekleştirilen ölçümler sonucunda, çift eksenli güneş takip sistemine sahip panellerin günlük toplam enerji üretimi 15.1 Wh olarak belirlendi. Tek

eksenli güneş takip sistemine sahip panellerin enerji üretimi 9.1 Wh iken, sabit açılı panellerin enerji üretimi 6.74 Wh olarak ölçüldü. Bu sonuçlar incelendiğinde, çift eksenli güneş takip sisteminin 7 Ocak tarihinde enerji üretiminin sabit açılı panele göre %53.3 daha yüksek olduğu gözlendi. Aynı şekilde, tek eksenli güneş takip sistemi, sabit açılı sisteme oranla %26.3 daha yüksek bir verime sahipti. Çift eksenli ve tek eksenli güneş takip sistemleri arasında ise %39.4'lük bir verim farkı tespit edildi.

6 Ocak ile 7 Ocak tarihleri arasında yapılan ölçümler incelendiğinde havanın bulutluluk durumuna göre çift eksenli güneş panelinin havanın tam kapalı olduğu günlerde tam açık günlere oranla %46.3 enerji ürettiği görülmüştür, tek eksenli güneş takip sistemi ise %35.2 enerji ürettiği görülmüştür. Sabit açılı güneş panelinin ise %32.6 enerji ürettiği görülmüştür. Fakat burada önemli bir nokta, havanın %100 kapalı olduğu saatlerde güneş takip sistemli güneş panellerinin, sabit açılı sisteme oranla çok az bir enerji artışı sağladığı gözlenmiştir. Daha önce bahsedilen veriler saat 07:00 ile 11:00 arasında görülmüştür ve bu saatler arasında havanın bulutluluk oranı %0-40 arasındadır fakat havanın %100 kapalı olduğu saatlerde ise güneş takip sistemli panellerin sabit açılı güneş paneline oranla belirgin bir güç farkı gözlenmemiştir. Temmuz ile Ağustos aylarında yapılan ölçümlerin, 6 Ocak ile 7 Ocak arasında yapılan ölçümlerden farklı olmasının ana sebebi kış aylarındaki güneşlenme sürelerindeki düşüştür yani temmuz ayında görülen 12-14 saatlik güneşlenme süreleri, kış aylarında 8-10 saate kadar düşmektedir.

5.2. Öneriler

Bu araştırma projesinde görülen sonuçlardan yola çıkıldığında, güneş takip sisteminin verim analizi yapılırken, elde edilen ölçümlerin daha sık yapılmasının ölçüm sonuçlarını değiştirdiği gözlenmiştir. Bu yüzden 15 dakikalık aralık yerine daha sık ölçüm yapılması önerilebilir. Özellikle Arduino mega yerine daha gelişmiş bir mikro kontroller kullanılırsa, ölçümler belli bir aralıklarla kayıt alınmak yerine, internet bağlantısı aracılığı ile bir sisteme aktarılabilir ve anlık ölçüm sonuçları kaydedilebilir.

Ayrıca Ağustos ayında özellikle saat 12:00 ile 14:00 arasında güneş takip sistemli panellerin sabit açılı panelden daha az güç üretmesinin, Ağustos ayında yaşanan yüksek sıcaklıklar ile bağlantısının olup olmadığını anlamak için, 2 adet çift eksenli güneş takip sistemi kullanılabilir ve bunlardan birisine bir soğutma sistemi eklenebilir. Bu sayede

eğer soğutma sistemi eklenen çift eksenli güneş takip sistemli panel Ağustos ayında öğle vakitlerinde yüksek hava sıcakları yaşanmasına rağmen, sabit açılı panelden fazla güç üretirse, bu fikir doğrulanabilir.

Bu projede kullanılan Arduino mega mikro kontrol cihazının bir çok giriş ve çıkış pinlerine sahip olması ve bu sayede güneş takip sistemi projesinde gereken kontrol görevini yapabiliyor olmasına rağmen, cihazın kullandığı ATmega tabanlı işlemcinin yetersiz olduğu görülmüştür. Bu nedenle Arduino mega aynı anda birçok cihazı kontrol edebilmesine rağmen, karmaşık hesaplamalarda veya mikro hafıza kartı kaydı sırasında sorun oluşturduğu görülmüştür. Bu nedenle gelecekte yapılacak olan araştırmalarda daha yüksek işlemci hızına sahip olan ve Wifi özellikli esp8266 veya Raspberry pi tercih edilmesi önerilir.

Bu çalışmada kullanılan lityum iyon bataryaların ve diğer hassas elektronik parçaların yaz aylarındaki yüksek hava sıcakları sırasında aşırı ısınmaları ve hatta yapmış olduğum araştırmanın 24 Ağustos tarihinde sonlanmasına sebep olması nedeniyle, ileride yapılacak benzer çalışmalarda bu konuda gerekli tedbirlerin ve sistemlerin kullanılması önemle öneririm.

Çalışmada kullanılan dişli kutusu sistemi yerine gezegen dişli kutusu kullanılabilir, bu sayede daha hassas açı değişimleri sağlanabilir.

6. KAYNAKLAR

Abdallah, S. (2004). "The effect of using sun tracking systems on the voltage-current characteristics and power generation of flat plate photovoltaics", Amman Department of Mechanical and Industrial Engineering, Applied Science University, 1671-1679.

Adom, P. K. (2011). "Electricity Consumption-Economic Growth Nexus: The Ghanaian Case", International Journal of Energy Economics and Policy, 17-19.

Ahmed, R., ve Khademul, I. M. (2010). "Improvement of Efficiency for Solar Photovoltaic Cell Application", Department of Electrical and Electronic Engineering, BRAC University, Dhaka, Bangladesh, 3-5.

Aksoy, T. C. ve Yavuz, C. (2018). "Determining Optimum Tilt Angles Of Solar Surfaces in Sakarya, Turkey", Faculty ofElectrical and Electronics Engineering Department, Sakarya University Engineering, 15-22.

Armakan, E. (2003). "Analysis Of Two-Axis Sun Tracking System", Izmir lnstitute of Technology, 57-68.

Arsalan, S. (2013). "Sun Tracking System with Microcontroller 8051". International Journal of Scientific & Engineering Research. International Journal of Scientific & Engineering Research, 4-6.

Aydın, B. ve Kobya, H. İ. (2015)." Güneş Takip Sistemi", Karabük Mühendislik Fakültesi Mekantronik Mühendisliği, 34-36.

Bajpai, P. ve Subhash, K. (2011). "Design, Development And Performance Test Of An Automatic Two-Axis Solar Tracker System". Annual IEEE India Conference: Engineering Sustainable Solutions, 26-37.

Balghouthi, M. vd. (2016). "Potential Of Concentrating Solar Power (CSP) Technology In Tunisia And The Possibility Of Interconnection With Europe". Renewable and Sustainable Energy Reviews, 28973-28996.

Bangali, J. ve Arvind, S. (2013). “Design And Implementation Of Security Systems For Smart Home Based On Gsm Technology”. International Journal of Smart Home, 201-208.

Barsoumi, N. (2011). “Fabrication of Dual-Axis Solar Tracking Controller Project”. Intelligent Control and Automation, 57-68.

Başak, S. ve Abdullah, Ş. (2017). “PID Denetimli Güneş Takip Sistemi”, Sakarya Üniversitesi Elektrik Elektronik Mühendisliği, 47-57.

Bavel, J. (2013). “The World Population Explosion: Causes, Backgrounds And Projections For The Future”, 2011 India Conference: Engineering Sustainable Solutions, 270-281.

Bayraç, H. N. (2009). “Küresel Enerji Politikalari ve Türkiye: Petrol ve Doğal Gaz Kaynaklari Açisindan Bir Karşilaştirma”. Eskişehir Osmangazi Üniversitesi Sosyal Bilimler Dergisi, 115-130.

Beyoğlu, M. F. (2011). “Balıkesir İlinde Çift Eksenli Güneş Takip Sistemi ile Sabit Eksenli PV Sistemin Verimlerinin Karşılaştırılması”, Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Mühendisliği, 40-67.

Bilgin, Z. (2006). “Güneş Takip Sistemi Tasarımı ve Gerçekleştirilmesi”, Elektrik Elektronik Mühendisliği Gazi Üniversitesi Fen Bilimleri Enstitüsü, 65-71.

Çalışkan, H. ve Harun, K. Ö. (2008). “Güneş Takip Sistemlerinin İncelenmesi”, Uşak Üniversitesi, Müh. Fak., Makina Müh. Böl., 1-5.

Chowdhury, K. I. ve Mosaddequr, R. (2017). “Performance Comparison Between Fixed Panel, Single-axis and Dual-axis Sun Tracking Solar Panel System”, Department Of Electrical And Electronic Engineering Brac Unıversıty, 4-10.

Dennis, N. (2000). “The Origin and Evolution of the Solar System”. Monthly Notices of the Royal Astronomical Society, 600-609.

EBay, (2013). “Micro SD Card Card Adapter Reader Module for Arduino”, 1-2.

Electronics, Rajguru (2017). “Lm393 Motor Speed Measuring Sensor Module For

Arduino",1-3.

Gensed, (2021). "Türkiye'de Güneş Enerjisine Olan Yatırımlar", https://www.gensed.org.

Gerden, T. (2018). "The Adoption Of The Kyoto Protocol Of T He United Nations Framework Convention On Climate Change",United Nations, 58-62.

Ghassoul, M. (2009). "Design of an Automatic Solar Tracking System to Maximize Energy Extraction", International Journal of Emerging Technology and Advanced Engineering, 453-482.

Gill, K. , Shuang, H. Y. , Fang, Y. , ve Xin, L. (2009). "A ZigBee-Based Home Automation System". IEEE Transactions on Consumer Electronics, 422-452.

Hellmich, M. ve Rüdiger, K. 2021. "Carbon Finance", Indicon-2011,270-281.

Kansal, R. (2008). "PIC Based Automatic Solar Radiation Tracker", Electronics Instrumentation & Control Engineering Thapar Unıversıty, 21-31.

Kumar, S. (2018). "Innovative Intravenous Fluid Control and Emergency Monitoring System". International Journal of Engineering and Manufacturing Science, 41-51.

Kürklü, Y. (2017). "Güneş Kaynaklı Farklı Enerji Üretim Sistemlerinde Çevresel Etkilerin Kıyaslanması ve Çözüm Önerileri", Over The Rim ,190-300.

Kusriyanto, M. ve Bambang, D. P. (2017). "Smart Home Usıng Local Area Network (LAN) Based Arduıno Mega 2560". ICWT 2016: 2nd International Conference on Wireless and Telematics, 127-158.

Marliyani, B. O. (2009). "Low Cost Solar Tracker". Faculty of Electrical & Electronics Engineering Universiti Malaysia Pahang, 1689-1790.

Maxim Integrated, (2015). "DS1302 Datasheet",1-13.

Demirtaş, M. (2006). "Bilgisayar Kontrollü Güneş Takip Sisteminin Tasarımı ve Uygulaması", Politenik Dergisi 9(4), 247-290.

Menak, R. (2018). "Çift Eksenli Güneş Takip Sistemi Tasarımı ve Modellemesi", Siirt

Üniversitesi Fen Bilimleri Enstitüsü Elektrik-Elektronik Mühendisliği,71-91.

Moussa, A. (2018). "Clımate Change Success Storıes Morocco's Noor: Largest Concentrated Solar Power Plant In Afrıca", AJC Journal 7(2), 14-28.

Musa, Y. (2017). "İki Eksenli Güneş Takip Sistemlerinde Takip Verimliliğin Arttırılması", Batman Üniversitesi Yaşam Bilimleri Dergisi 7 ,56-118.

Öztürk, H. (2004). "Güneş Enerjisinden Fotovoltaik Yöntemle Elektrik Üretiminde Güç Dönüşü Verimi ve Etkili Etmenler", Pamukkale Üniversitesi, Müh. Fak., Makina Müh. Böl, 1-16.

Piyare, R. (2013). "Internet of Things: Ubiquitous Home Control and Monitoring System using Android based Smart Phone", International Journal of Internet of Things 2013, 5–11.

Praveen, R. P. (2020). "Performance Analysis And Optimization Of Central Receiver Solar Thermal Power Plants For Utility Scale Power Generation", Sustainability,1-12.

Rittenberry, R. (2005). "Hands-on technology: L298N Dual H-Bridge Motor Driver", Occup Health, 24-74.

Rüstemli, S. Dincer, F. Çelik, M. ve Cengiz, M. S. (2013). "Fotovoltaik Paneller: Güneş Takip Sistemleri ve İklimlendirme Sistemleri", BEU Journal of Science 2, 141–147.

Sadeque, F. (2020). "Desıgn And Implementatıon Of Automated Solar Trackıng System", Bangladesh Unıversuty Of Engıneerıng And Technology,47-54.

Sensors, Dark. (2008). "Light / Dark Sensors", Technology: 2008,1-10.

Solargis. (2021). "Solar Resource Map".

Sparavigna A. C. (2018). "Exercises in Archaeoastronomy -2 -The passage mound of Newgrange", HAL (2), 3-12.

Steiner T. (2008). "ACCENT Global Change Magazine", Climate 12, 1-2.

Turan A. (2015). “Assessment of Geothermal and Solar Hybrid Power Generation Technologies in Turkey and Its Application to Menderes Graben”, Assessment 19, 1-25.

Uzunok, S. (2007). “Fotovoltaik Modüllerin Elektrik Enerjisi Üretiminde Güneş Takip Sisteminin Etkisinin İncelenmesi”, Mustafa Kemal Üniversitesi Fen Bilimleri Enstitüsü Makine Mühendisliği, 3-6.

Weather Spark. (2023) “Konya Ağustos 2023 Tarihi Hava Durumu Verileri (Türkiye) - ”.https://tr.weatherspark.com/h/m/97310/2023/8/Ağustos-2023-tarihinde-in-Konya-Türkiye-Tarihi-Hava-Durumu#Figures-ColorTemperature.

Yavuz, A. H. (2020). “Isı Borulu Vakum Tüplü Termoelektrik Güneş Jeneratörü Tasarımı ve Uygulaması”. Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi (9), 180-265.

Yousif, C I. (2002). “Comparison Study Between The Performance Of Tracking And Stationary Solar Photovoltaic Systems in Malta”, Institute for Energy Technology, University of Malta, 720-723.

Yousif, E. ve Raghad, H. (2013). “Photodegradation And Photostabilization Of Polymers, Especially Polystyrene: Review”. SpringerPlus 2(1): 1–32.

Yuksekkaya, B. vd. (2006). “A GSM, Internet And Speech Controlled Wireless Interactive Home Automation System”. IEEE Transactions on Consumer Electronics, 837-843.

7. EKLER

EK-1 Arduino Mega yazılımı setup_loop.ino dosyası

```
#define DEBUG false

#include <Arduino.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
#include <RtcDS1302.h>
#include "ldr_sensor.h"
#include "SdCardHandler.h"
LdrSensor ldr_dual_C;
#include "Motor_Control.h"
MotorControl tracker;
const int button3 = 26;
const int button2 = 27;  //arduino mega sensör shield üzerinde sd card kısmında
const int button1 = 28;  // Etiketleme hata var: 25,34,23,22 olarak belirtilmiş ama doğrusu: 25 24 23 22
const int togglesw1 = 29;
bool buttonarray[3][3][3];  // ix  ,0 , 0 1 2 button1,,,,

int panelA[3] = { 0, 0, 0 };  // panel analog ölçüm dizisi
// RTC DEĞİŞKENLERİ
int myyear;
int mymonth;
int myday;
int myday_previous;  // panelleri ne zaman sıfırlayacağımızı bilmek için kullanılır
int myhour;
int mymin;
int mysec;
String date;
String date_dmy;
String date_hms;
bool timer15m = false;
bool timer15s = false;
bool timerx1 = false;
// DS 1302 RTC AYARLARI BAŞLANGIÇ
//  BAĞLANTILAR:
//  DS1302 RST/CE --> 2 mavi 6
//  DS1302 DAT/IO --> 4 kırmızı 5
//  DS1302 CLK/SCLK --> 5 siyah  4
//  DS1302 GND --> GND yeşil
//  DS1302 VCC --> 3.3v - 5v  sarı
ThreeWire myWire(5, 4, 6);  // DAT, CLK, RST
RtcDS1302<ThreeWire> Rtc(myWire);
#define countof(a) (sizeof(a) / sizeof(a[0]))
// DS 1302 RTC AYARLARI SON

String test = "REAL1";
String reset = "REAL1";
String Rtc_Test_File = "RTC";
String Rtc_Reset_File = "RTC";
String storage_test = "storage.txt";
String storage_reset = "storage.txt";
String event_recorder_test = "datalogger";
String event_recorder_reset = "datalogger";
String Last_Rtc_File;
String holderS = "start";
const int motorin3a = 10;
const int motorin4a = 11;
const int dual_eswReed = 40;
bool yaw_lastdirection = false;
int yaw_last_move = 0;
const int encoderPin = 19;

volatile int encoderValue = 0;
unsigned long mytime1a;
unsigned long mytime1b;
long mytime1c;
unsigned long mytime2a;
unsigned long mytime2b;
long mytime2c;
unsigned long mytime3a;
unsigned long mytime3b;
long mytime3c;
int encoder_mintime = 200;
int encoder_steps = 3;
long int yaw_full_step;
long int yaw_current_location;

bool timer_fin5 = false;
bool timer_fin15 = false;
bool timer_fin60 = false;

//////////////////////////////FONKSİYON TANIMLAMALARI BAŞLANGIÇ//////////////////////////

long int move_Yaw_RESETTER();
bool single_axis_ldr_move();
bool dual_axis_ldr_move();
bool motor_move_YAW(bool clockwise, int slotCount);
bool move_Yaw_EswEscape(unsigned long EswEscape_timeout);
bool motor_move_FREE_yaw(bool clockwise, int slotCount);

void SD_system_event_recorder(String event, bool x);
```

```
void sd_card_panel_write();

void rtc_update();
void printDateTime(const RtcDateTime &dt);
void rtc_test();
bool minutebetweenEvent(bool x, bool y, int
z);

void pc_testmode(int x);
void pc_testmode_sd(int x);
void pc_testmode_motor(int x);
void pc_testmode_analogRead(int x);
void pc_testmode_final(int x);
bool single_axis_ldr_move_test();
bool dual_axis_ldr_move_test();
void test_functions(int x);
void userinput();
void handleEncoder();

void resetArray();
void battery_voltage();
void toggleswchecker();
void timed(bool x, int y);
int freeMemory();
//////////////////////////////FONKSİYON
TANIMLAMALARI
SON//////////////////////////


enum PanelState {
  NORMAL,
  NIGHTTIME,
  USER_OVERRIDE_OFFLINE

};
// Global variables
PanelState S_Panel = NORMAL;
PanelState S_PanelHolder = NORMAL;

void setup() {
  Serial.begin(9600);
  pinMode(LED_BUILTIN, OUTPUT);  //
hızlı yanıp sönme ve yanık kalma = sd kart
hatası anlamına gelir
  ldr_dual_C.init();              // yavaş yanıp
sönme ve ardından kapanma = sd kart
çalışıyor
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Hello World");
  Serial.println("Hello World");
  delay(1000);
  lcd.clear();
  lcd.noBacklight();
  initializeSDCard();
  pinMode(encoderPin, INPUT_PULLUP);

attachInterrupt(digitalPinToInterrupt(encoder
Pin), handleEncoder, CHANGE);
  tracker.init();
  pinMode(button3, INPUT);
  pinMode(button2, INPUT);
  pinMode(button1, INPUT);
  pinMode(togglesw1, INPUT);

  Serial.print(__DATE__);
  Serial.println(__TIME__);
  Rtc.Begin();
  rtc_test();  // rtc'yi test et
  timer_fin5 = minutebetweenEvent(true, true,
0);
  timer_fin15 = minutebetweenEvent(true,
true, 1);
  timer_fin60 = minutebetweenEvent(true,
true, 2);
  holderS = "Arduino start at setup";
//Arduino başlatıldı sd kart olay günlüğü
kaydı
  SD_system_event_recorder(holderS, false);
}

void loop() {

  if (Serial.available() > 0) {  // Herhangi bir
giriş olup olmadığını kontrol et
    int x = Serial.parseInt();   // Girişi bir
tamsayı olarak oku
    Serial.print("User input : ");
   Serial.println(x);  // Kullanıcı girişini yazdır

    pc_testmode(x);
    x = 0;

  } else {
    toggleswchecker();
    // Eğer bugün ilk kez saat 19 ise, bundan
sonra saat 6:00 olana kadar saatin önemi yok
    if ((myhour == 19) && (myday_previous
!= myday)) {
      myday_previous = myday;
      S_Panel = NIGHTTIME;  //sistemin
durumu (state) gece modunda,
      //Gece modunda güç tasaruffu için saat
dışında tüm sistemler sabah 06:00 a kadar
devre dışı

      // Panel resetleme olaylarını kaydet
     holderS = " 19:00 night resetting panels;";
      SD_system_event_recorder(holderS,
false);
```

```
      // Tek eksenli güneş takip sisteminin doğu batı motorunu resetle

tracker.move_Pitch_RESETTER(tracker.motorin1b);
      holderS = "tracker.move_Pitch_RESETTER(tracker.motorin1b);";
      SD_system_event_recorder(holderS, false);
      // Çift eksenli güneş takip sisteminin kuzey güney motorunu resetle
      yaw_full_step = move_Yaw_RESETTER();
      holderS = "move_Yaw_RESETTER";
      SD_system_event_recorder(holderS, false);
     // Çift eksenli güneş takip sisteminin doğu batı motorunu resetle

tracker.move_Pitch_RESETTER(tracker.motorin1a);
      holderS = "tracker.move_Pitch_RESETTER(tracker.motorin1a);";
      SD_system_event_recorder(holderS, false);

      // Eğer sistem gece modu durumunda ise ve saat 6 ise sistemi tekrar başlat
    } else if ((S_Panel == NIGHTTIME) && (myhour == 6)) {
      S_Panel = NORMAL;
      timer_fin5 = minutebetweenEvent(true, true, 0);
      timer_fin15 = minutebetweenEvent(true, true, 1);
      // Sistemi açma olayını kaydet
     holderS = " 06:00 morning turning on the system";
      SD_system_event_recorder(holderS, false);
    }
    // Eğer Sistem  NORMAL durumunda ise (saat 06:00 ile 19:00 arası)
    if (S_Panel == NORMAL) {
      //5 dakika zamanlayıcıyı başlat
      timer_fin5 = minutebetweenEvent(false, false, 0);
      //5 dakika zamanlayıcı foto direnç sensörlerini ölç ve gerekli ise motorları hareket ettir
      if (timer_fin5 == true) {
        holderS = "5 min timer ldr moving;";
        SD_system_event_recorder(holderS, false);
        single_axis_ldr_move();
        dual_axis_ldr_move();
        Serial.println("5 Min timer  start");
        //5 dakika zamanlayıcıyı sıfırla
        timer_fin5 = minutebetweenEvent(true, true, 0);
      }
      //15 dakika zamanlayıcıyı başlat
      timer_fin15 = minutebetweenEvent(false, false, 1);
      if (timer_fin15 == true) {
        //15 dakika zamanlayıcı panel ölçüm ve kaydı
        holderS = "15 min timer panel record;";
        SD_system_event_recorder(holderS, false);
        ldr_dual_C.panel_measure();
        sd_card_panel_write();
        Serial.println("15 Min timer  start");

        //15 dakika zamanlayıcıyı sıfırla
       timer_fin15 = minutebetweenEvent(true, true, 1);
      }
      //60 dakika zamanlayıcıyı başlat
      timer_fin60 = minutebetweenEvent(false, false, 2);
      if (timer_fin60 == true) {
        //60 dakika zamanlayıcı pil şarj durumu ölçüm ve kaydı
        holderS = "60 min timer battery record;";
        battery_voltage();
        //60 dakika zamanlayıcıyı sıfırla
        timer_fin60 = minutebetweenEvent(true, true, 2);
      }
    }
  }
}
```

EK-2 Arduino Mega yazılımı motor_Control.cpp dosyası

```
#include "HardwareSerial.h"
#include "Arduino.h"
#include "Motor_Control.h"
MotorControl::MotorControl() {
// Kurucu fonksiyon
  totalDistance = 0.0;
  currentLocation = 0.0;
  lastMovement = 0;
  const int dual_eswReed = 40;
  const int motorin3a = 10;
  const int motorin4a = 11;
}


void MotorControl::init() {
  pinMode(motorin1a, OUTPUT);
  pinMode(motorin2a, OUTPUT);
  pinMode(motorin3a, OUTPUT);
  pinMode(motorin4a, OUTPUT);
  pinMode(motorin1b, OUTPUT);
  pinMode(motorin2b, OUTPUT);
  digitalWrite(motorin1a, LOW);
  digitalWrite(motorin2a, LOW);
  digitalWrite(motorin3a, LOW);
  digitalWrite(motorin4a, LOW);
  digitalWrite(motorin1b, LOW);
  digitalWrite(motorin2b, LOW);
  pinMode(dual_eswL, INPUT);
    pinMode(dual_eswR, INPUT);
      pinMode(single_eswR, INPUT);
        pinMode(single_eswL, INPUT);
          pinMode(dual_eswReed, INPUT);
  Serial.println("Motors ready");
  return;
}

bool MotorControl::motor_move_PITCH(bool
clockwise, unsigned long Mseconds, int
motorPin) {  //dual pitch (motorin1a)pitch single
(motorin1b)
  bool r = false;
// esw trip return
  int z1 = (motorPin + 1);
  bool zw = true;  // while
  //Serial.print("direction ==
");Serial.println(clockwise);
Serial.print("motorPin ==
");Serial.println(motorPin);
  int eswxpin = 0;
  if ((motorPin == motorin1a) && (clockwise
== true)) {
    eswxpin = 0;
    Serial.print("DUAL  AXIS PITCH 1a WEST
");
  }
  if ((motorPin == motorin1a) && (clockwise
== false)) {
    eswxpin = 1;
    Serial.print("DUAL  AXIS PITCH 1a EAST
");
  }
  if ((motorPin == motorin1b) && (clockwise
== true)) {
    eswxpin = 3;
    Serial.print("SINGLE  AXIS PITCH 1b
EAST ");
  }
  if ((motorPin == motorin1b) && (clockwise
== false)) {
    eswxpin = 4;
    Serial.print("SINGLE  AXIS PITCH 1b
WEST ");
  }


  if (motorpanic == true) {
    Serial.println("MOTOR  PANIC MODE ON,
MOTOR MOVE FUNCTION BREAK ");
    MotorKillAll();
    return false;
  } else if ((motorpanic_dualPitch == true) &&
(motorPin == motorin1a)) {
    Serial.println("dualPitch  MOTOR PANIC
MODE ON,  MOTOR MOVE FUNCTION
BREAK ");
    MotorKillAll();
    return false;
  } else if ((motorpanic_SinglePitch == true)
&& (motorPin == motorin1b)) {
    Serial.println("SinglePitch  MOTOR PANIC
MODE ON,  MOTOR MOVE FUNCTION
BREAK ");
    MotorKillAll();
    return false;
  }
  Esw_check();

  if ((esw_array[eswxpin]) == false) {
    Serial.print("eswx  ");
    Serial.print(eswxpin);
    Serial.println("  tripped in pitch motor move
function before motor move started  ");
    Serial.println("kill  the motors");
    zw = false;
    r = false;
    MotorKillAll();
  }
  timed(true);

  while (zw == true) {
    timed(false);
    digitalWrite(motorPin, clockwise);
    digitalWrite(z1, !clockwise);
    Esw_check();
```

```
    if ((esw_array[eswxpin]) == false) {
      Serial.print("eswx  ");
      Serial.print(eswxpin);
      Serial.println("   tripped during pitch motor
move function ");
      Serial.println("kill  the motors");
      MotorKillAll();

      zw = false;
      r = false;
    } else if (mytimeMc > Mseconds) {
      Serial.println();
      Serial.print("motor stopped  ");
      switch (motorPin) {
        case 8:
          Serial.print("1a");
          break;
        case 30:
          Serial.print("1b");
          break;
      }
      zw = false;
      r = true;
    }
  }

  digitalWrite(motorPin, LOW);
  digitalWrite(z1, LOW);
  timed(false); delay(250);
  Serial.print(" stopped "); //Serial.print(r);
  return r;
}



bool
MotorControl::move_Pitch_EswEscape(unsigne
d long EswEscape_timeout) {
  int eswxpin = 0;
  int motorPin = 0;
  bool clockwise = false;
  Esw_check();
  //Esw_display();
  Serial.print("motor panic = ");
  Serial.println(motorpanic);
  if (((esw_array[0] == 0 && esw_array[1] ==
0) || (esw_array[3] == 0 && esw_array[4] ==
0)) || motorpanic == true) {
    Serial.println("IMPOSSIBLE  CONDITION
WIRING ISSUE ");
    Serial.println("OR  ESW ESCAPE CALLED
WHEN MOTOR PANIC ON EXITING ");
    motorpanic = true;
    return false;
  } else if ((!esw_array[0]) &&
(motorpanic_dualPitch == false)) {
    motorPin = motorin1a;
    clockwise = false;
    eswxpin = 0;
  } else if ((!esw_array[1]) &&
(motorpanic_dualPitch == false)) {
    motorPin = motorin1a;
    clockwise = true;
    eswxpin = 1;
  } else if ((!esw_array[3]) &&
(motorpanic_SinglePitch == false)) {
    motorPin = motorin1b;
    clockwise = false;
    eswxpin = 3;
  } else if ((!esw_array[4]) &&
(motorpanic_SinglePitch == false)) {
    motorPin = motorin1b;
    clockwise = true;
    eswxpin = 4;
  } else {
    Serial.println("ESW  ESCAPE CALLED
WITHOUT NEED");
    Serial.println("OR  WHEN MOTORPANIC
ON EXITING");
    return false;
  }

  int z1 = (motorPin + 1);
  bool r = false;  // Return value
  bool zw = true;  // while loop control variable

  // Start moving the motor in the specified
direction

  timed(true);
  while (zw == true) {
    digitalWrite(motorPin, clockwise);
    digitalWrite(z1, !clockwise);
    timed(false);
    Esw_check();

    if (esw_array[eswxpin] == true) {
      Serial.print("eswx  ");
      Serial.print(eswxpin);
      //Serial.println("  stopped tripping during
escape motor function ");

      zw = false;
      r = true;
    } else if (mytimeMc > EswEscape_timeout) {
      zw = false;
      r = false;
      switch (motorPin) {
        case 8:
          Serial.print("DualPitch");
          motorpanic_dualPitch = true;
          break;
        case 30:
          Serial.print("SinglePitch");
          motorpanic_SinglePitch = true;
          break;
      }
```

```
    Serial.println("ESW ESCAPE FUNCTION FAILED DUE TO TIMEOUT ACTIVATING MOTOR PANIC MODE");
   }
  }
  //Serial.println("giving some extra time to motors 250 ms add tolarance and 250 ms for end ");
  delay(250);

  // Stop the motor
  digitalWrite(motorPin, LOW);
  digitalWrite(z1, LOW);
  delay(250);
  Esw_check();
  if (esw_array[eswxpin] == true) {
   r = 1;
   Serial.println("ESW ESCAPE SUCCESFULL");
  } else {
   r = 0;
   Serial.println("Something happened after motors stop and esw tripping again ");
  }
  return r;



  // Implementation of move_Pitch_EswEscape
}

unsigned long MotorControl::move_Pitch_RESETTER(int motorPin) {
  long false_to_true = 0;
  long true_to_false = 0;
  long temporary_holder = 0;
  long Total_Distance = 0;
  bool esw_trip_bool[10];

  esw_trip_bool[0] = motor_move_PITCH(true, 40000, motorPin);
  if (esw_trip_bool[0]) {
   Serial.println("Resetter function timeout ");
   return 0;
  } else {

   esw_trip_bool[1] = move_Pitch_EswEscape(5000);
  }
  if (esw_trip_bool[1]) {

   Serial.println("RESETTER AT EAST MEASURING ");
   timed(true);
   esw_trip_bool[2] = motor_move_PITCH(false, 40000, motorPin);
   if (!esw_trip_bool[3]) {
    timed(false);
    false_to_true = mytimeMc;
    Serial.print("TIME BETWEEN EAST TO WEST = ");
    Serial.println(mytimeMc);
    esw_trip_bool[4] = move_Pitch_EswEscape(5000);
    if (esw_trip_bool[4]) {
     Serial.println("RESETTER AT WEST MEASURING ");
     timed(true);
     esw_trip_bool[5] = motor_move_PITCH(true, 40000, motorPin);
     if (!esw_trip_bool[5]) {
      timed(false);
      true_to_false = mytimeMc;
      temporary_holder = (false_to_true / 2);
      Serial.print("TIME BETWEEN WEST TO EAST = ");
      Serial.println(mytimeMc);
      esw_trip_bool[6] = move_Pitch_EswEscape(5000);
      timed(true);
      if (esw_trip_bool[6]) {
       Serial.print("Moving for middle for reset ");
       Serial.println(temporary_holder);
       esw_trip_bool[7] = motor_move_PITCH(false, temporary_holder, motorPin);
       if (esw_trip_bool[7]) {
        Serial.println("Resetter function Finished ");
        return ((false_to_true + true_to_false) / 2);
       }
      }
     }
    }
   }
  }
  ///HERE///
}



void MotorControl::Esw_display() {
  Esw_check();
  Serial.print(" DUAL PITCH LEFT = ");
  Serial.println(esw_array[0]);
  Serial.print(" DUAL PITCH RIGHT = ");
  Serial.println(esw_array[1]);
  Serial.print(" YAW = ");
  Serial.println(esw_array[2]);
  Serial.print(" SINGLE PITCH LEFT = ");
  Serial.println(esw_array[3]);
  Serial.print(" SINGLE PITCH RIGHT = ");
  Serial.println(esw_array[4]);
}
```

```
void MotorControl::Esw_check() {
  //0 =DUAL PITCH LEFT , 1=RIGHT ,
2=YAW , 3 =SINGLE PITCH LEFT ,
4=SINGLE PITCH RIGHT
  esw_array[0] = (digitalRead(dual_eswL));
  esw_array[1] = (digitalRead(dual_eswR));
  esw_array[2] = (digitalRead(dual_eswReed));
  esw_array[3] = (digitalRead(single_eswL));
  esw_array[4] = (digitalRead(single_eswR));
}

void MotorControl::timed(bool start) {
  if (start == true) {
    mytimeMa = millis();
  }
  mytimeMb = millis();
  mytimeMc = (mytimeMb - mytimeMa);
  // Implementation of timed function
}

void MotorControl::MotorKillAll()  {
  digitalWrite(motorin1a, LOW);
  digitalWrite(motorin2a, LOW);
  digitalWrite(motorin3a, LOW);
  digitalWrite(motorin4a, LOW);
  digitalWrite(motorin1b, LOW);
  digitalWrite(motorin2b, LOW);
}
void MotorControl::Motor_identifier() {

  motor_move_FREE(true, 1000, motorin1a);
  motor_move_FREE(false, 1000, motorin1a);
  motor_move_FREE(true, 1000, motorin3a);
  motor_move_FREE(false, 1000, motorin3a);
  motor_move_FREE(true, 1000, motorin1b);
  motor_move_FREE(false, 1000, motorin1b);
}

bool MotorControl::motor_move_FREE(bool
clockwise, unsigned long Mseconds, int
motorPin) {
  bool r = false;  // esw trip return
  int z1 = (motorPin + 1);
  bool zw = true;  // while
  // end switch check
  int eswxpin = 0;
  if ((motorPin == motorin1a) && (clockwise
== true)) {
    eswxpin = 0;
    Serial.print("DUAL  AXIS PITCH 1a EAST
");
  }
  if ((motorPin == motorin1a) && (clockwise
== false)) {
    eswxpin = 1;
    Serial.print("DUAL  AXIS PITCH 1a WEST
");
  }

  if ((motorPin == motorin3a) && (clockwise
== false)) {
    eswxpin = 2;
    Serial.print("DUAL  AXIS YAW 3a WEST
");
  }
  if ((motorPin == motorin3a) && (clockwise
== true)) {
    eswxpin = 2;
    Serial.print("DUAL  AXIS YAW 3a WEST
");
  }

  if ((motorPin == motorin1b) && (clockwise
== true)) {
    eswxpin = 3;
    Serial.print("SINGLE  AXIS PITCH 1b
EAST ");
  }
  if ((motorPin == motorin1b) && (clockwise
== false)) {
    eswxpin = 4;
    Serial.print("SINGLE  AXIS PITCH 1b
WEST ");
  }

  timed(true);

  while (zw == true) {
    timed(false);
    digitalWrite(motorPin,  clockwise);
    digitalWrite(z1,  !clockwise);

    if (mytimeMc > Mseconds) {
      Serial.println();
      Serial.print("motor ");
      switch (motorPin) {
        case 8:
          Serial.print("1a");
          break;
        case 10:
          Serial.print("3a");
          break;
        case 30:
          Serial.print("1b");
          break;
      }
      zw = false;
      r = true;
    }
  }

  digitalWrite(motorPin,  LOW);
  digitalWrite(z1,  LOW);
  timed(false);
  Serial.print(" stopped");

  return r;
}
```

EK-3 Arduino Mega yazılımı

motor_Control.h dosyası

Boşlukları ayarla

```
#ifndef MOTOR_CONTROL_H
#define MOTOR_CONTROL_H
#include <Arduino.h>
class MotorControl {
public:
  MotorControl();
  void init();
  void Motor_identifier();
  bool motor_move_PITCH(bool clockwise,
unsigned long Mseconds, int motorPin);
  bool move_Pitch_EswEscape(unsigned long
EswEscape_timeout);
  bool Esw_Escape();
  bool motor_move_FREE(bool clockwise,
unsigned long Mseconds, int motorPin);
  void Esw_check();
  void Esw_display();
  unsigned long move_Pitch_RESETTER(int
motorPin);
  void MotorKillAll();
  bool esw_array[5] = { 0, 0, 0, 0, 0 };  //0
=DUAL  PITCH LEFT , 1=RIGHT , 2=YAW , 3
=SINGLE  PITCH LEFT , 4=SINGLE  PITCH
RIGHT
  const int motorin3a = 10;
  const int motorin4a = 11;
  const int dual_eswReed = 40;
  bool motorpanic = false;
  bool motorpanic_dualPitch = false;
  bool motorpanic_Yaw = false;
  bool motorpanic_SinglePitch = false;
  const int motorin1a = 8;
  const int motorin2a = 9;
  const int motorin1b = 30;
  const int motorin2b = 31;

private:
  const int dual_eswL = 42;
  const int dual_eswR = 38;
  const int single_eswR  = 44;  // LOW when sw
tripped, HIGH when it's  not tripped
  const int single_eswL  = 39;
  const int minMotorSpeed = 50;
  const int maxMotorSpeed = 200;
  unsigned long mytimeMa;
  unsigned long mytimeMb;
  long mytimeMc;
  bool motorset = false;
  float totalDistance;
  float currentLocation;
  int lastMovement;
  void timed(bool start);
};

#endif
```

EK-4 Arduino Mega yazılımı SdCardHandler.cpp dosyası

```
#include "Arduino.h"
#include "SdCardHandler.h"
#include <SdFat.h>
#include <SPI.h>
#include "sdios.h"  //VERY CRTICIAL CODE
SdFat SD;
File MyFile;
const int SD_CS_PIN = 53;
void initializeSDCard() {
  Serial.print("Initializing SD card...");

  if (!SD.begin(SD_CS_PIN)) {
    Serial.println("initialization failed!");
    for (int i = 0; i < 10; i++) {
      digitalWrite(LED_BUILTIN, LOW);
      delay(100);
      digitalWrite(LED_BUILTIN, HIGH);
      delay(100);
    }
    return;
  }
  for (int i = 0; i < 5; i++) {
    digitalWrite(LED_BUILTIN, HIGH);
    delay(100);
    digitalWrite(LED_BUILTIN, LOW);
    delay(1000);
  }
  Serial.println("initialization done.");
}

void sd_write(const char *fileName, const char
*message) {
  Serial.print("MESSAGE ");
  Serial.println(message);
  MyFile = SD.open(fileName, FILE_WRITE);

  // if the file opened okay, write to it:
  if (MyFile) {
    Serial.print("Writing ");
    Serial.print(fileName);
    Serial.println(" file: ");
    Serial.println(message);
    MyFile.println(message);
    // close the file:
    MyFile.close();
    Serial.println("done.");
  } else {
    // if the file didn't open, print an error:
    Serial.print("error opening ");
    Serial.println(fileName);
  }
  return;
}

void sd_read(const char *fileName) {
  // re-open the file for reading:

  MyFile = SD.open(fileName);
  if (MyFile) {
    Serial.print("Reading ");
    Serial.print(fileName);
    Serial.print(" file. ");
    Serial.print(fileName);
    Serial.println(" :");

    // read from the file until there's nothing else
in it:
    while (MyFile.available()) {
      Serial.write(MyFile.read());
    }
    // close the file:
    MyFile.close();
  } else {
    // if the file didn't open, print an error:
    Serial.print("error opening ");
    Serial.print(test);
    Serial.println(" file ");
  }
  return;
}

void SD_event_addrow(const char *fileName,
String date, String event) {

  char temporaryBuffer1[200];
  Serial.print("FILENAME ");
  Serial.println(fileName);
  sprintf(temporaryBuffer1, "%s,%s ",
date.c_str(), event.c_str());
  MyFile = SD.open(fileName, FILE_WRITE);

  // if the file opened okay, write to it:
  if (MyFile) {
    Serial.print("Writing ");
    Serial.print(fileName);
    Serial.println(" file: ");
    Serial.println(temporaryBuffer1);
    MyFile.println(temporaryBuffer1);
    // close the file:
    MyFile.close();
    Serial.println("done.");
  } else {
    // if the file didn't open, print an error:
    Serial.print("error opening ");
    Serial.println(fileName);
  }
  return;
}
void sd_addrow(const char *fileName, String
date, int int1, int int2, int int3) {
  char temporaryBuffer[100];
  Serial.print("FILENAME ");
  Serial.println(fileName);
```

```
  sprintf(temporaryBuffer, "%s,%d ,%d ,%d ",
date.c_str(), int1, int2, int3);
  MyFile = SD.open(fileName, FILE_WRITE);

  // if the file opened okay, write to it:
  if (MyFile) {
    Serial.print("Writing   ");
    Serial.print(fileName);
    Serial.println("  file: ");
    Serial.println(temporaryBuffer);
    MyFile.println(temporaryBuffer);
    // close the file:
    MyFile.close();
    Serial.println("done.");
  } else {
    // if the file didn't open, print an error:
    Serial.print("error  opening ");
    Serial.println(fileName);
  }
  return;
}

bool sd_filecheck(char *fileName) {

  if (SD.exists(fileName))  {
    Serial.print(fileName);
    Serial.println("exists.");
    return 1;
  } else {
    Serial.print(fileName);
    Serial.println("  doesn't exist.");
    return 0;
  }
}

void sd_filedelete(char *fileName) {
  // delete the file:
  if (SD.exists(fileName))  {
    Serial.print("Removing  ");
    Serial.println(fileName);
    SD.remove(fileName);
    if (SD.exists(fileName))  {
      Serial.print(fileName);
      Serial.println("  exists failed to remove.");
    } else {
      Serial.println("File  removed done");
    }
  } else {
    Serial.print(fileName);
    Serial.println("  doesn't exist failed to
remove.");
  }
}

int sd_card_file_storage(bool y, int x = 0) {  // x
new value stored in file, y =true write mode,
false = read only  x=0 is, if you just going to
read no point sending value to fnuction
  const char *fileName = "storage.txt";
  int fileContent;
  if (y == true) {
    // deleting previous recording
    if (SD.exists(fileName))  {
      Serial.print("Removing  ");
      Serial.println(fileName);
      SD.remove(fileName);
      if (SD.exists(fileName))  {
        Serial.print(fileName);
        Serial.println("  exists failed to remove.");
      } else {
        Serial.println("File  removed done");
      }
    }
    MyFile = SD.open(fileName, FILE_WRITE);
    if (MyFile) {
      Serial.print("Writing   ");
      Serial.print(x);
      Serial.print("  to storage.txt");
      MyFile.println(x);
      // close the file:
      MyFile.close();
      Serial.println("done.");
    } else {  // if the file didn't open, print an
error:
      Serial.println("error  opening storage.txt");
    }
  }

  // re-open the file for reading:
  MyFile = SD.open(fileName);
  if (MyFile) {
    Serial.println("storage.tx:");

    // read from the file until there's nothing else
in it:

    if (MyFile.available()) {
      fileContent = MyFile.parseInt();
    }

    // close the file:
    MyFile.close();
    Serial.print("File  Content: ");
    Serial.println(fileContent);
  } else {
    // if the file didn't open, print an error:
    Serial.println("error  opening storage.txt");
  }
  return fileContent;
}
```

EK-5 Arduino Mega yazılımı SdCardHandler.h dosyası

```
#ifndef SDCARDHANDLER_H
#define SDCARDHANDLER_H
#include <SdFat.h>
#include <SPI.h>
#include "sdios.h" //VERY CRTICIAL CODE
extern SdFat SD; // Declare SD as an external
variable
extern File MyFile;
extern const int SD_CS_PIN;
extern int sd_addrow_array[3];
extern String date_hms;
extern String test;
extern String reset;
void initializeSDCard();
bool sd_filecheck( char *fileName);
void sd_filedelete( char *fileName);
void sd_write( const char *fileName, const char
*message);
void sd_read( const char *fileName);
void sd_addrow(const char *fileName, String
date, int int1, int int2, int int3);
void SD_event_addrow( const char *fileName,
String date,String event );
int sd_card_file_storage(bool y, int x = 0);
#endif
```

EK-6 Arduino Mega yazılımı ldr_sensor.cpp dosyası

```
#include "binary.h"
#include "HardwareSerial.h"
#include "ldr_sensor.h"

LdrSensor::LdrSensor() {
}

void LdrSensor::init() {
  Serial.begin(9600);

  pinMode(MosfetRed, OUTPUT);
  pinMode(MosfetGreen, OUTPUT);
  pinMode(MosfetBlue, OUTPUT);
  pinMode(MosfetMotor, OUTPUT);
  digitalWrite(MosfetRed, LOW);
  digitalWrite(MosfetGreen, LOW);
  digitalWrite(MosfetBlue, LOW);
  digitalWrite(MosfetMotor, LOW);
  Serial.println("LDR  INIT");
  pinMode(relayHON, OUTPUT);
  digitalWrite(relayHON, LOW);

  pinMode(panelA1, INPUT);
  pinMode(panelA2, INPUT);
  pinMode(panelA3, INPUT);

  pinMode(ldr1D, INPUT);
  pinMode(ldr2D, INPUT);
  pinMode(ldr3D, INPUT);
  pinMode(ldr4D, INPUT);

  pinMode(ldr1S, INPUT);
  pinMode(ldr2S, INPUT);
}
void LdrSensor::battery_control(int x, bool y) {
  switch (x) {
    case 1:
      {
        Serial.print("Mosfet Red ");
        Serial.println(y);
        digitalWrite(MosfetRed, y);
      }
      break;
    case 2:
      {
        Serial.print("Mosfet Green ");
        Serial.println(y);
        digitalWrite(MosfetGreen, y);
      }
      break;
    case 3:
      {
        Serial.print("Mosfet Blue ");
        Serial.println(y);
        digitalWrite(MosfetBlue, y);
      }
      break;
    case 4:
      {
        Serial.print("Mosfet Motor, ");
        Serial.println(y);
        digitalWrite(MosfetMotor, y);
      }
      break;
    case 5:
      {
        Serial.println("ALL  Mosfet OFF, ");
        digitalWrite(MosfetRed, HIGH);
        digitalWrite(MosfetGreen, HIGH);
        digitalWrite(MosfetBlue, HIGH);
        digitalWrite(MosfetMotor, HIGH);
      }
      break;
    case 6:
      {
        Serial.println("MOSFET  CHECK ");
        Serial.print("Mosfet Red ");
        Serial.println(digitalRead(MosfetRed));
        Serial.print("Mosfet Green ");
        Serial.println(digitalRead(MosfetGreen));
        Serial.print("Mosfet Blue ");
        Serial.println(digitalRead(MosfetBlue));
        Serial.print("Mosfet Motor, ");
        Serial.println(digitalRead(MosfetMotor));
      }
      break;
    case 0:
      {
        Serial.println("ALL  Mosfet ON, ");
        digitalWrite(MosfetRed, LOW);
        digitalWrite(MosfetGreen, LOW);
        digitalWrite(MosfetBlue, LOW);
        digitalWrite(MosfetMotor, LOW);
      }
      break;
  }
}

int LdrSensor::ldr_dual_compare_V2() {
  sensor_analog(numReadings,
numInitialReadingsToDiscard, ldr1D, ldr2D,
ldr3D, ldr4D);
  int TR, BR, TL, BL = 0;
  TR = sensoranalogA[0];
  BR = sensoranalogA[1];
  TL = sensoranalogA[2];
  BL = sensoranalogA[3];
  Serial.println("");
  int r = 4;
  int avgT = (TR + TL) / 2;
  int avgB = (BR + BL) / 2;
  int avgR = (TR + BR) / 2;
  int avgL = (BL + TL) / 2;
  int dif_TB = (avgT - avgB);
  int dif_RL = (avgR - avgL);
```

```
  Serial.print("TR  = ");
  Serial.print(TR);
  Serial.print("   BR = ");
  Serial.print(BR);
  Serial.print("   TL = ");
  Serial.print(TL);
  Serial.print("   BL = ");
  Serial.println(BL);

  Serial.print("avgT  = ");
  Serial.print(avgT);
  Serial.print("   avgB = ");
  Serial.print(avgB);
  Serial.print("   avgR = ");
  Serial.print(avgR);
  Serial.print("   avgL = ");
  Serial.println(avgL);

  Serial.print("dif_TB  = ");
  Serial.print(dif_TB);
  Serial.print("   dif_RL = ");
  Serial.println(dif_RL);
  int tol = 10;

  if ((-1 * tol > dif_TB) || (dif_TB > tol)) {

    if (avgT > avgB) {
      Serial.println("TOP");
      r = 0;
    } else if (avgT < avgB) {
      Serial.println("  BOTTOM");
      r = 1;
    }


  } else if ((-1 * tol > dif_RL) || (dif_RL > tol)) {

    if (avgR > avgL) {
      Serial.println("RIGHT");
      r = 2;
    } else if (avgR < avgL) {
      Serial.println("  LEFT");
      r = 3;
    }
  }
  return r;
}

bool LdrSensor::battery_measure() {
  Serial.println("start reading");
  digitalWrite(relayHON,  HIGH);
  delay(100);
  sensor_analog(numReadings,
numInitialReadingsToDiscard, BatRed,
BatGreen, BatBlue, BatMotor);
  delay(100);
  digitalWrite(relayHON,  LOW);
  for (int i = 0; i < 4; i++) {
    battery_voltage_array_A[i] = (1023 -
(sensoranalogA[i]));
    battery_voltage_array_V[i] =
(((battery_voltage_array_A[i]) * 5) / 1023);
  }
  /*
  Serial.println("");
  Serial.print("RED Battery  A =   ");
  Serial.print(battery_voltage_array_A[0]);
  Serial.print("  V  =  ");
  Serial.println(battery_voltage_array_V[0]);
  Serial.print("GREEN  Battery A =  ");
  Serial.print(battery_voltage_array_A[1]);
  Serial.print("  V  =  ");
  Serial.println(battery_voltage_array_V[1]);
  Serial.print("BLUE  Battery A =  ");
  Serial.print(battery_voltage_array_A[2]);
  Serial.print("  V  =  ");
  Serial.println(battery_voltage_array_V[2]);
  Serial.print("MOTOR  Battery A =  ");
  Serial.print(battery_voltage_array_A[3]);
  Serial.print("  V  =  ");
  Serial.println(battery_voltage_array_V[3]);
  */
}
int LdrSensor::ldr_single_compare() {  // return
0 = error , 1=top, 2=bottom ,3 all equal

  sensor_analog(numReadings,
numInitialReadingsToDiscard, ldr1S, ldr2S);
  int hold = 0;
  int tol = 10;
  int S_top = sensoranalogA[0];
  int S_bottom = sensoranalogA[1];
  Serial.print("S_top  = ");
  Serial.println(S_top);
  Serial.print("S_bottom = ");
  Serial.println(S_bottom);
  if (S_top > (S_bottom + tol)) {
    Serial.println("TOP");
    hold = 1;
    return hold;
  } else if (S_bottom > (S_top + tol)) {
    Serial.println("BOTTOM");
    hold = 2;
    return hold;
  } else {
    hold = 3;
    return hold;
  }
}
void LdrSensor::timed(bool x, int y) {
  switch (y) {
    case 1:
      if (x == true) {
        mytime1a = millis();
      }
      mytime1b = millis();
      mytime1c = (mytime1b - mytime1a);
```

```
    break;
   case 2:
    if (x == true) {
     mytime2a = millis();
    }
    mytime2b = millis();
    mytime2c = (mytime2b - mytime2a);
    break;
   case 3:
    if (x == true) {
     mytime3a = millis();
    }
    mytime3b = millis();
    mytime3c = (mytime3b - mytime3a);
    break;
  }
  return;
}

void LdrSensor::panel_measure() {
  Serial.println("start reading");
  digitalWrite(relayHON, HIGH);
  delay(100);
  sensor_analog(numReadingspanel,
numInitialReadingsToDiscard, panelA1,
panelA2, panelA3);
  delay(100);
  digitalWrite(relayHON, LOW);
  Serial.println("end reading");
  for (int i = 0; i < 3; i++) {
   panelA[i] = (1023 - (sensoranalogA[i]));
  }
  for (int i = 0; i < 3; i++) {
   Serial.print("Panel");
   Serial.print(i + 1);
   Serial.print(" = ");
   Serial.println(panelA[i]);
  }
  Serial.println("end function");
  return;
}


void LdrSensor::sensor_analog(int x, int y, int
z1 = 0, int z2 = 0, int z3 = 0, int z4 = 0) {
  int analogInputPins[] = { z1, z2, z3, z4 };
  int numPins = 0;
  // Aktif pin sayısını belirle
  for (int i = 0; i < sizeof(analogInputPins) /
sizeof(analogInputPins[0]); i++) {
   if (analogInputPins[i] != 0) {
    numPins++;
   } else {
    break;
   }
  }
  int readings[numPins][x];
  int index[numPins] = { 0 };
  unsigned long total[numPins] = { 0 };

  // İlk okumaları atla
  while (index[0] < y) {
   for (int i = 0; i < numPins; i++) {
    analogRead(analogInputPins[i]);
    delay(5);
    index[i]++;
   }
  }

  // Analog girişleri birden çok kez oku ve
ortalamayı hesapla
  if (!(index[0] < y)) {
   for (int i = 0; i < x; i++) {
    for (int j = 0; j < numPins; j++) {
     readings[j][index[j]] =
analogRead(analogInputPins[j]);
     delay(5);
     total[j] += readings[j][index[j]];
     index[j] = (index[j] + 1) % x;
    }
   }

   // Her pin için stabil okumaların ortalamasını
hesapla
   int average[numPins];
   for (int i = 0; i < numPins; i++) {
    average[i] = total[i] / x;
    sensoranalogA[i] = (1023 - (average[i]));
   }

   // Değişkenleri bir sonraki döngü için sıfırla
   for (int i = 0; i < numPins; i++) {
    total[i] = 0;
    index[i] = 0;
   }
  }
  return;
}
```

EK-7 Arduino Mega yazılımı ldr_sensor.h dosyası

```
#ifndef LDR_SENSOR_H
#define LDR_SENSOR_H
#include <Arduino.h>
class LdrSensor {
public:
  LdrSensor();
  void init();
  void timed(bool x, int y);
  void sensor_analog(int x, int y, int z1 = 0, int
z2 = 0, int z3 = 0, int z4 = 0);
  //  void sensor_analogV2(int x, int y, int z1 =
0, int z2 = 0, int z3 = 0, int z4 = 0);
  void battery_control(int x, bool y);
  void panel_measure();
  bool battery_measure();
  int ldr_dual_compare_V2();
  int ldr_single_compare();
  unsigned long mytime1a;
  unsigned long mytime1b;
  long mytime1c;
  unsigned long mytime2a;
  unsigned long mytime2b;
  long mytime2c;
  unsigned long mytime3a;
  unsigned long mytime3b;
  long mytime3c;
  int panelA[3] = { 0, 0, 0 };
  int batteryA[4] = { 0, 0, 0, 0 };            //battery
analog read value
  float batteryV[4] = { 0, 0, 0, 0 };
//battery voltage readings
  bool batteryMosfetState[4] = { 0, 0, 0, 0 };
//////////////////!!!!! WHY BAT V A 2 ?
  float battery_voltage_array_A[4] = { 0, 0, 0, 0
};
  float battery_voltage_array_V[4] = { 0, 0, 0, 0
};
  const int MosfetRed = 46;
  const int MosfetGreen = 47;
  const int MosfetBlue = 48;
  const int MosfetMotor = 49;
  const int relayHON = 45;
  const int panelA1 = A13;
  const int panelA2 = A14;
  const int panelA3 = A15;
  const int BatRed = A8;
  const int BatGreen = A9;
  const int BatBlue = A10;
  const int BatMotor = A11;
private:
  const int ldr1D = A0;  // WEST NORTH BL
  const int ldr2D = A1;  // EAST NORTH TL
  const int ldr3D = A2;  // WEST SOUTH BR
  const int ldr4D = A3;  // EAST SOUTH TR
  const int ldr1S = A4;  // east
  const int ldr2S = A5;  // west
  const float R1 = 6.9;
  const float R2 = 22;
  const float RT = 28.9;
  int singleldrtol  = 10;
  int dualldrtol = 10;
  int tolnew = 50;  //user entered tol
  //MOTOR
  int sensoranalogA[4] = { 0, 0, 0, 0 };  // analog
read function can measure 8 pins at once tops
  const int numReadings = 50;  // keep this 500-
1000 for better results
  const int numInitialReadingsToDiscard = 3;
  const int numReadingspanel = 501;  // keep
this 500-1000 for better results
  int ldrSA[2]  = { 0, 0 };        // analog read
  int ldrDA[4]  = { 0, 0, 0, 0 };  // analog read

  // analog read
};

#endif
```

EK-8 Arduino Mega yazılımı functions.ino dosyası

```
//////////////////////////////MAIN FUNCTION LIST START//////////////////////////

long int move_Yaw_RESETTER();
bool single_axis_ldr_move();
bool dual_axis_ldr_move();
bool motor_move_YAW(bool clockwise, int slotCount);
bool move_Yaw_EswEscape(unsigned long EswEscape_timeout);
bool motor_move_FREE_yaw(bool clockwise, int slotCount);

void SD_system_event_recorder(String event, bool x);
void sd_card_panel_write();

void rtc_update();
void printDateTime(const RtcDateTime &dt);
void rtc_test();
bool minutebetweenEvent(bool x, bool y, int z);

void pc_testmode(int x);
void pc_testmode_sd(int x);
void pc_testmode_motor(int x);
void pc_testmode_analogRead(int x);
void pc_testmode_final(int x);
bool single_axis_ldr_move_test();
bool dual_axis_ldr_move_test();
void test_functions(int x);
void userinput();
void handleEncoder();

void resetArray();
void battery_voltage();
void toggleswchecker();
void timed(bool x, int y);
int freeMemory();

//////////////////////////////MAIN FUNCTION LIST END//////////////////////////


//////////////////////////////MOTOR FUNCTIONS START //////////////////////////

long int move_Yaw_RESETTER();
bool single_axis_ldr_move();
bool dual_axis_ldr_move();
bool motor_move_YAW(bool clockwise, int slotCount);
bool move_Yaw_EswEscape(unsigned long EswEscape_timeout);
bool motor_move_FREE_yaw(bool clockwise, int slotCount);

long int move_Yaw_RESETTER() {
  int yaw_false_to_true = 0;
  int yaw_true_to_false = 0;
  int temporary_holder = 0;
  long int Total_Distance = 0;
  bool esw_trip_bool[10];
  int motorPin = 10;

  esw_trip_bool[0] = motor_move_YAW(true, 10000);
  if (esw_trip_bool[0]) {
    Serial.println("Resetter  function timeout ");
    return 0;
  } else {

    esw_trip_bool[1] = move_Yaw_EswEscape(5000);
  }
  if (esw_trip_bool[1]) {

    Serial.println("RESETTER   AT EAST MEASURING   ");

    esw_trip_bool[2] = motor_move_YAW(false, 10000);
    if (!esw_trip_bool[3]) {

      yaw_false_to_true = encoderValue;
      Serial.print("STEP  BETWEEN EAST  TO WEST  = ");
      Serial.println(yaw_false_to_true);
      esw_trip_bool[4] = move_Yaw_EswEscape(5000);
      if (esw_trip_bool[4]) {
        Serial.println("RESETTER  AT WEST MEASURING   ");

        esw_trip_bool[5] = motor_move_YAW(true, 10000);
        if (!esw_trip_bool[5]) {

          yaw_true_to_false = encoderValue;

          temporary_holder = (yaw_false_to_true / 2);
          Serial.print("STEP  BETWEEN WEST TO EAST  = ");
          Serial.println(yaw_true_to_false);
          esw_trip_bool[6] = move_Yaw_EswEscape(5000);

          if (esw_trip_bool[6]) {
            Serial.print("Moving  for middle for reset ");
            Serial.println(temporary_holder);
            // esw_trip_bool[7] = motor_move_YAW(false, temporary_holder);
            if (esw_trip_bool[6]) {
```

```
            return ((yaw_false_to_true +
yaw_true_to_false) / 2);
          }
         }
        }
       }
     }
  }
  ///HERE///
}
bool single_axis_ldr_move() {
  if (tracker.motorpanic_SinglePitch) {
    holderS = "tracker.motorpanic_SinglePitch
TRUE ";
    SD_system_event_recorder(holderS, false);
  }
  bool fin = false;
  int single_axisR = 0;
  bool fin_ldr_fine = 0;
  bool eswx = false;        //false is issue
  int eswx_row_pitchT = 0;  //row stuck checker
  int eswx_row_pitchF = 0;  //row stuck checker
  bool r = false;
  timed(true, 3);
  while (!fin) {
    timed(false, 3);
    single_axisR =
ldr_dual_C.ldr_single_compare();
    switch (single_axisR) {
      case 0:
        {
         Serial.println("single_axis_ldr_move
ERROR RETURN 0");
          fin = true;
          r = false;
        }
        break;
      case 1:
        {
         eswx = tracker.motor_move_PITCH(true,
50, tracker.motorin1b);
          fin_ldr_fine = false;
          if (!eswx) {
            eswx_row_pitchT++;
            holderS = "motorin1b eswx  tripped ";
            SD_system_event_recorder(holderS,
false);
            tracker.move_Pitch_EswEscape(5000);
          }
        }
        break;
      case 2:
        {
         eswx =
tracker.motor_move_PITCH(false, 50,
tracker.motorin1b);
          fin_ldr_fine = false;
          if (!eswx) {
            eswx_row_pitchT++;
            holderS = "motorin1b eswx tripped ";
            SD_system_event_recorder(holderS,
false);
            tracker.move_Pitch_EswEscape(5000);
          }
        }
        break;
      case 3:
        {
         Serial.println("single_axis_ldr_move
LDR fine waiting again");
          delay(1000);
          if (fin_ldr_fine) {
            fin = true;
            r = true;
          }
          fin_ldr_fine = true;
        }
        break;
    }
    if (mytime3c > 30000) {
      fin = true;
      r = false;
      holderS = "single_axis_ldr_move timeout ";
      SD_system_event_recorder(holderS, false);
      Serial.println("single_axis_ldr_move
timeout ");
    } else if ((eswx_row_pitchT > 2) ||
(eswx_row_pitchF > 2)) {
      Serial.println("single_axis_ldr_move  esw
tripping in rows ");
      fin = true;
      r = false;
      holderS = "dual_axis_ldr_move esw trips in
rows";
      SD_system_event_recorder(holderS, false);


tracker.move_Pitch_RESETTER(tracker.motori
n1b);
      holderS = "
tracker.move_Pitch_RESETTER(tracker.motori
n1b);";
      SD_system_event_recorder(holderS, false);
    }
  }
  if (r == true) {
    Serial.println("single_axis_ldr_move  LDR
fine done");
    holderS = " single_axis_ldr_move LDR fine
done;";
    SD_system_event_recorder(holderS, false);
  }

  return r;
}
```

```
bool dual_axis_ldr_move() {  //0 = WEST, 1 =EAST , 2 =NORTH, 3=SOUTH,4=all fine
                        // yaw_full_step = move_Yaw_RESETTER();
                        // tracker.move_Pitch_RESETTER(tracker.motorin1a);
  if (tracker.motorpanic_dualPitch) {
    holderS = "tracker.motorpanic_dualPitch TRUE ";
    SD_system_event_recorder(holderS, false);
  }
  bool fin = false;
  int dual_axisR = 0;
  bool fin_ldr_fine = 0;
  bool r = false;
  bool eswx = false;        //false is issue
  int eswx_row_pitchT = 0;  //row stuck checker
  int eswx_row_pitchF = 0;  //row stuck checker
  int eswx_row_yawT = 0;    //row stuck checker
  int eswx_row_yawF = 0;    //row stuck checker

  timed(true, 3);
  while (!fin) {
    timed(false, 3);
    dual_axisR = ldr_dual_C.ldr_dual_compare_V2();
    switch (dual_axisR) {
      case 0:
        {
          eswx = tracker.motor_move_PITCH(true, 50, tracker.motorin1a);
          fin_ldr_fine = false;

          if (!eswx) {
            eswx_row_pitchF++;
            holderS = "motorin1a eswx 1 tripped ";
            SD_system_event_recorder(holderS, false);
            tracker.move_Pitch_EswEscape(5000);
          }
        }
        break;
      case 1:
        {
          eswx = tracker.motor_move_PITCH(false, 50, tracker.motorin1a);
          fin_ldr_fine = false;
          if (!eswx) {
            eswx_row_pitchT++;
            holderS = "motorin1a eswx 0 tripped ";
            SD_system_event_recorder(holderS, false);
            tracker.move_Pitch_EswEscape(5000);
          }
        }
        break;
      case 2:
        {
          eswx = motor_move_YAW(true, 5);
          fin_ldr_fine = false;

          if (!eswx) {
            eswx_row_yawT++;
            holderS = "yaw motor reed tripped at true yaw move ";
            SD_system_event_recorder(holderS, false);
            move_Yaw_EswEscape(5000);
          }
        }
        break;

      case 3:
        {
          eswx = motor_move_YAW(false, 5);
          fin_ldr_fine = false;

          if (!eswx) {
            eswx_row_yawF++;
            holderS = "yaw motor reed tripped at false yaw move ";
            SD_system_event_recorder(holderS, false);
            move_Yaw_EswEscape(5000);
          }
        }
        break;
      case 4:
        {
          Serial.println("dual_axis_ldr_move LDR fine waiting again");
          delay(1000);
          if (fin_ldr_fine) {
            fin = true;
            r = true;
          }
          fin_ldr_fine = true;
        }
        break;
    }
    if (mytime3c > 60000) {
      fin = true;
      r = false;
      holderS = "dual_axis_ldr_move timeout ";
      SD_system_event_recorder(holderS, false);
      Serial.println("dual_axis_ldr_move timeout ");
    } else if ((eswx_row_pitchT > 2) || (eswx_row_pitchF > 2) || (eswx_row_yawT > 2) || (eswx_row_yawF > 2)) {
      Serial.println("dual_axis_ldr_move esw tripping in rows ");
      fin = true;
      r = false;
      holderS = "dual_axis_ldr_move esw trips in rows";
```

```
      SD_system_event_recorder(holderS, false);
      yaw_full_step = move_Yaw_RESETTER();
      holderS = "move_Yaw_RESETTER";
      SD_system_event_recorder(holderS, false);

tracker.move_Pitch_RESETTER(tracker.motori
n1a);
      holderS = "
tracker.move_Pitch_RESETTER(tracker.motori
n1a);";
      SD_system_event_recorder(holderS, false);
    }
  }
  if (r == true) {
    Serial.println("dual_axis_ldr_move LDR fine
done");
    holderS = " dual_axis_ldr_move LDR fine
done;";
    SD_system_event_recorder(holderS, false);
  }

  return r;
}

bool motor_move_YAW(bool clockwise, int
slotCount) {  //dual pitch (motorin1a)pitch
single  (motorin1b)
  bool r = false;                              // esw trip
return
  int motorPin = 10;
  int z1 = (motorPin + 1);
  bool eswx = true;  // end switch check
  bool zw = true;
  // while
  //Serial.print("direction  ==
");Serial.println(clockwise);
Serial.print("motorPin  ==
");Serial.println(motorPin);
  int eswxpin = 2;
  encoderValue = 0;
  if (clockwise)  {

    Serial.print("DUAL  AXIS YAW 3a north ");
  }
  if (!clockwise)  {

    Serial.print("DUAL  AXIS YAW 3a south ");
  }
  if (tracker.motorpanic == true) {
    Serial.println("MOTOR  PANIC MODE ON,
MOTOR MOVE FUNCTION BREAK ");
    tracker.MotorKillAll();
    return false;
  } else if (tracker.motorpanic_Yaw == true) {
    Serial.println("YAW  MOTOR PANIC
MODE ON,  MOTOR MOVE FUNCTION
BREAK ");
    tracker.MotorKillAll();
    return false;
  }
  tracker.Esw_check();
  tracker.esw_array[2];

  if (!tracker.esw_array[2]) {
    Serial.print("eswx  ");
    Serial.print(eswxpin);
    Serial.println("   tripped in pitch motor move
function before motor move started  ");
    Serial.println("kill  the motors");
    tracker.MotorKillAll();
    zw = false;
    r = false;
  }
  timed(true, 1);

  while  (zw == true) {
    timed(false, 1);
    digitalWrite(motorPin,  clockwise);
    digitalWrite(z1,  !clockwise);
    tracker.Esw_check();
    if  (encoderValue >= slotCount) {
      digitalWrite(motorin3a, LOW);
      digitalWrite(motorin4a, LOW);
      Serial.print("encoder value = ");
      Serial.println(encoderValue);
      zw = false;
      r = true;
    }
    if  (!tracker.esw_array[2]) {
      Serial.print("eswx  ");
      Serial.print(eswxpin);
      Serial.println("   tripped during pitch motor
move function ");
      Serial.println("kill  the motors");
      tracker.MotorKillAll();

      zw = false;
      r = false;
    }
  }

  digitalWrite(motorPin,  LOW);
  digitalWrite(z1,  LOW);
  timed(true, 1);
  int encodervalue_fullstop = encoderValue;
  while  (mytime1c < 1000) {
    if  (encoderValue != encodervalue_fullstop) {
      timed(true, 1);
      encodervalue_fullstop = encoderValue;
    } else {
      timed(false, 1);
    }
  }
  Serial.print("encoder value full stop = ");
  Serial.println(encoderValue);
  Serial.println("  stopped  "); //Serial.print(r);
  yaw_lastdirection = clockwise;
  yaw_last_move = encoderValue;
```

```
  if ((clockwise  == true) && (r == false)) {

    yaw_current_location = 0;
  } else if (clockwise)  {
    yaw_current_location =
(yaw_current_location - encoderValue);
  } else if ((clockwise  == false) && (r == false))
{

    yaw_current_location = yaw_full_step;
  } else if (!clockwise)  {
    yaw_current_location =
(yaw_current_location + encoderValue);
  }
  Serial.print("current location  ");
  Serial.println(yaw_current_location);
  return r;
}

bool move_Yaw_EswEscape(unsigned long
EswEscape_timeout) {
  int eswxpin = 2;
  int motorPin = 10;
  int z1 = (motorPin + 1);
  bool yaw_eswEscape_stage1 = false;
  bool yaw_eswEscape_stage2 = false;
  bool yaw_eswEscape_stage3 = false;
  bool yaw_eswEscape_stage4 = false;
  encoderValue = 0;
  int encodervalue_stuck = encoderValue;
  tracker.Esw_check();
  tracker.Esw_display();
  if (tracker.motorpanic == true) {
    Serial.println("MOTOR  PANIC MODE ON,
YAW ESW ESCAPE FUNCTION BREAK ");
    tracker.MotorKillAll();
    return false;
  } else if (tracker.motorpanic_Yaw == true) {
    Serial.println("YAW  MOTOR PANIC
MODE ON, YAW ESW ESCAPE FUNCTION
BREAK ");
    tracker.MotorKillAll();
    return false;
  }
  if (tracker.esw_array[2]) {
    Serial.println("YAW  ESW ESCAPE
CALLED WITHOUT NEED");
  } else {
    Serial.println("YAW  ESW ESCAPE START
LAST KNOWN SYSTEM");
    yaw_eswEscape_stage1 = true;
  }
  timed(true, 1);
  timed(true, 2);
  while (yaw_eswEscape_stage1) {
    timed(false, 1);
    timed(false, 2);
    if ((encoderValue) > (encodervalue_stuck +
encoder_steps)) {
      timed(true, 2);
      encodervalue_stuck = encoderValue;
    } else if (mytime2c > encoder_mintime) {
      Serial.println("DUAL  YAW MOTOR
STUCK ");
      tracker.MotorKillAll();
      yaw_eswEscape_stage1 = false;
    }

    digitalWrite(motorPin,  !yaw_lastdirection);
    digitalWrite(z1,  yaw_lastdirection);
    tracker.Esw_check();
    if (tracker.esw_array[2]) {
      Serial.print("eswx  ");
      Serial.print(eswxpin);
      Serial.println("stopped  tripping ");
      Serial.println("kill  the motors");

      //Serial.println("giving  some extra time to
motors 250 ms add tolarance and 250 ms for
end ");
      delay(250);
      yaw_eswEscape_stage1 = false;
      digitalWrite(motorPin,  LOW);
      digitalWrite(z1,  LOW);
      delay(250);

      return true;

    } else if (mytime1c > EswEscape_timeout) {
      Serial.println("DUAL  YAW MOTOR STOP
TIMEOUT");

      tracker.MotorKillAll();
      yaw_eswEscape_stage1 = false;
      tracker.motorpanic_Yaw = true;
      holderS = "tracker.motorpanic_Yaw = true;
after escape fail ";
      SD_system_event_recorder(holderS, false);
    }
  }


  // Implementation of move_Yaw_EswEscape
}

bool motor_move_FREE_yaw(bool clockwise,
int slotCount) {  //dual pitch (motorin1a)pitch
single (motorin1b)
  bool r = false;                                  // esw
trip return
  int motorPin = 10;
  int z1 = (motorPin + 1);
  bool zw = true;
  //Serial.print("direction  ==
");Serial.println(clockwise);
Serial.print("motorPin  ==
");Serial.println(motorPin);
```

```
  encoderValue = 0;

  if (clockwise) {

   Serial.print("DUAL  AXIS YAW 3a north ");
  }
  if (!clockwise) {

   Serial.print("DUAL  AXIS YAW 3a south ");
  }


  int encodervalue_stuck = encoderValue;

  timed(true, 1);
  timed(true, 2);
  while (zw == true) {
   timed(false, 1);
   digitalWrite(motorPin,  clockwise);
   digitalWrite(z1,  !clockwise);
   timed(false, 2);
   if ((encoderValue) > (encodervalue_stuck +
encoder_steps)) {
     timed(true, 2);
     encodervalue_stuck = encoderValue;
   } else if (mytime2c > encoder_mintime) {
     Serial.println("DUAL  YAW MOTOR
STUCK ");
     tracker.MotorKillAll();
     zw = false;
   }
   if (encoderValue >= slotCount) {
     digitalWrite(motorin3a, LOW);
     digitalWrite(motorin4a, LOW);
     Serial.print("encoder value = ");
     Serial.println(encoderValue);
     zw = false;
     r = true;
   } else if (mytime1c > 5000) {
     Serial.println("DUAL  YAW MOTOR STOP
TIMEOUT");



     zw = false;
     r = false;
   }
  }

  digitalWrite(motorPin,  LOW);
  digitalWrite(z1,  LOW);
  timed(true, 1);
  int encodervalue_fullstop = encoderValue;
  while (mytime1c < 1000) {
   if (encoderValue != encodervalue_fullstop) {
     timed(true, 1);
     encodervalue_fullstop = encoderValue;
   } else {
     timed(false, 1);
   }
  }
  Serial.print("encoder value full stop = ");
  Serial.println(encoderValue);
  Serial.print(" stopped  "); //Serial.print(r);
  return r;
}

//////////////////////////////MOTOR FUNCTIONS
END ////////////////////////////

//////////////////////////////SD CARD FUNCTIONS
START ////////////////////////////

void SD_system_event_recorder(String event,
bool x);
void sd_card_panel_write();


void SD_system_event_recorder(String event,
bool x) {

  rtc_update();

  Serial.println("event recorder");

  event_recorder_test = event_recorder_reset;
  event_recorder_test += " ";
  event_recorder_test += myday;
  event_recorder_test += " ";
  event_recorder_test += mymonth;
  event_recorder_test += " ";
  event_recorder_test += myyear;

  event_recorder_test += ".csv";
  const char *fileName =
event_recorder_test.c_str();



  SD_event_addrow(event_recorder_test.c_str(),
date, event);
  if (x) {
    sd_read(event_recorder_test.c_str());
  }


  return;
}

void sd_card_panel_write() {

  rtc_update();
  bool countr = false;  // true means file is new
  Serial.println("Start  sd_card_panel_write
function start");
  int counti;
  test = reset;
  test += " ";
  test += myday;
```

```
  test += " ";
  test += mymonth;
  test += " ";
  test += myyear;

  test += ".csv";
  const char *fileName = test.c_str();
  Serial.println("Start sd_card_panel_write
function start1");
  if (SD.exists(fileName)) {
    Serial.println("File exists!");
    countr = false;
  } else {
    Serial.println("File does not exist!");
    countr = true;
  }

  Last_Rtc_File = test;

  if (countr == true) {
    counti = sd_card_file_storage(true, 0);
  } else {
    counti = sd_card_file_storage(false);
    Serial.print("previous counter =");
    Serial.println(counti);
    counti++;
    counti = sd_card_file_storage(true, counti);
    Serial.print("current counter =");
    Serial.println(counti);
  }

  Serial.println(test);
  if (counti == 0) {
    sd_write(test.c_str(),
"Date,Panel1ADC,Panel2ADC,Panel3ADC");
  }


  Serial.println("Start sd_card_panel_write
function start end while");
  if (counti == 10) {
    Serial.println("read file time");
    sd_read(test.c_str());
  }
  sd_addrow(test.c_str(), date_hms,
ldr_dual_C.panelA[0], ldr_dual_C.panelA[1],
ldr_dual_C.panelA[2]);
  Serial.println("End sd_card_panel_write
function end");
  return;
}

//////////////////////////////SD CARD FUNCTIONS
END ////////////////////////////


//////////////////////////////RTC FUNCTIONS
START ////////////////////////////
```

```
void rtc_update();
void printDateTime(const RtcDateTime &dt);
void rtc_test();
bool minutebetweenEvent(bool x, bool y, int z);


void rtc_update() {
  RtcDateTime now = Rtc.GetDateTime();
  printDateTime(now);
  if (!now.IsValid()) {
    // Common Causes:
    //    1) the battery on the device is low or even
missing and the power line was disconnected
    Serial.println("RTC lost confidence in the
DateTime!");
  }
  return;
}
void printDateTime(const RtcDateTime &dt) {
  char datestring[20];
  char date_dmy_char[20];  // it seperates the
date to day month year
  char date_hms_char[20];  // it seperates the
date to hour minute second
  snprintf_P(date_dmy_char,
             countof(date_dmy_char),
             PSTR("%02u/%02u/%04u"),
             dt.Day(),
             dt.Month(),
             dt.Year());
  snprintf_P(date_hms_char,
             countof(date_hms_char),
             PSTR("%02u:%02u:%02u"),
             dt.Hour(),
             dt.Minute(),
             dt.Second());

  snprintf_P(datestring,
             countof(datestring),
             PSTR("%02u/%02u/%04u
%02u:%02u:%02u"),
             dt.Day(),
             dt.Month(),
             dt.Year(),
             dt.Hour(),
             dt.Minute(),
             dt.Second());
  myyear = dt.Year();
  mymonth = dt.Month();
  myday = dt.Day();
  myhour = dt.Hour();
  mymin = dt.Minute();
  mysec = dt.Second();
  date = String(datestring);
  date_dmy = String(date_dmy_char);
  date_hms = String(date_hms_char);
  // Serial.print(datestring);   Serial.println();
  return;
}
```

```
void rtc_test() {

  RtcDateTime compiled =
RtcDateTime(__DATE__, __TIME__);
  printDateTime(compiled);
  Serial.println();

  if (!Rtc.IsDateTimeValid()) {
    // Common Causes:
    //    1) first time you ran and the device wasn't
running yet
    //    2) the battery on the device is low or even
missing

    Serial.println("RTC  lost confidence in the
DateTime!");
    Rtc.SetDateTime(compiled);
  }

  if (Rtc.GetIsWriteProtected()) {
    Serial.println("RTC  was write protected,
enabling writing  now");
    Rtc.SetIsWriteProtected(false);
  }

  if (!Rtc.GetIsRunning()) {
    Serial.println("RTC  was not actively running,
starting now");
    Rtc.SetIsRunning(true);
  }

  RtcDateTime now = Rtc.GetDateTime();
  if (now < compiled) {
    Serial.println("RTC  is older than compile
time!  (Updating DateTime)");
    Rtc.SetDateTime(compiled);
  } else if (now > compiled) {
    Serial.println("RTC  is newer than compile
time.updating (this is expected)");
    Rtc.SetDateTime(compiled);
  } else if (now == compiled) {
    Serial.println("RTC  is the same as compile
time! (not expected but all is fine)");
  }
}

int timetrackerh[4][3];
int timetrackerm[4][3];  // 5 10 15 30 min

bool minutebetweenEvent(bool x, bool y, int z)
{  // bool x =true, start timer , false=check the
timer
  // I added 5 10 15 30 minutes versions, each
one has own array 0  1 2 3 , 0 1 2
  ///SUMMARY : X =true, start the timer ,
x=false check the timer
  //y == true it prints if timer finished or if it's
not prints the remaining time, y=false, only
prints if it's  finished or missed
  int hourmin = 0;
  int hourmin1 = 0;
  int hourmin2 = 0;
  int hourmin3 = 0;
  int hourmin4 = 0;
  int hourmin5 = 0;
  bool timef = false;
  int o1 = 0;  // return true if timer fin
  rtc_update();
  switch (z) {
    case 0:
      o1 = 5;
      break;
    case 1:
      o1 = 15;
      break;
    case 2:
      o1 = 60;
      break;
    case 3:
      o1 = 60;
      break;
  }
  if (x == true) {

    timetrackerm[z][0] = mymin;
    timetrackerh[z][0] = myhour;

    hourmin = ((myhour * 60) + (mymin + o1));

    hourmin1 = (hourmin / 60);  // 22
    hourmin2 = (hourmin % 60);  // 44
    timetrackerh[z][1] = hourmin1;
    timetrackerm[z][1] = hourmin2;
  } else {
    timetrackerm[z][2] = mymin;
    timetrackerh[z][2] = myhour;
    hourmin3 = ((myhour * 60) + (mymin));
    hourmin4 = ((timetrackerh[z][1] * 60) +
(timetrackerm[z][1]));
    hourmin5 = (hourmin4 - hourmin3);
    if (hourmin5 < 0) {
      Serial.print(o1);
      Serial.println("  minutes timer MISSED
MORE THAN: ");
      timef = true;
      return timef;
    }
    if (hourmin5 == 0) {
      Serial.print(o1);
      Serial.println("  minutes timer fin: ");
      timef = true;
      return timef;
    } else {
      if (y == true) {
        Serial.print(o1);
        Serial.print("  minutes timer remaining =:
");
        Serial.print(hourmin5);
```

```
      Serial.println("  minutes ");
     }
     timef = false;
     return timef;
    }
  }
  // if(myhour>=23&&mymin>44){//this means
15 min later is  new day not important since it'll
be sleeping}
  return timef;
}


//////////////////////////////RTC FUNCTIONS END
//////////////////////////

//////////////////////////////TEST FUNCTIONS
START //////////////////////////


void pc_testmode(int x);
void pc_testmode_sd(int x);
void pc_testmode_motor(int x);
void pc_testmode_analogRead(int x);
void pc_testmode_final(int x);
bool single_axis_ldr_move_test();
bool dual_axis_ldr_move_test();
void test_functions(int x);
void userinput();
void handleEncoder();



void pc_testmode(int x) {
  Serial.println("pc_testmode start");




  if (x > 99 && x < 200) {
    pc_testmode_sd(x);
  }
  if (x > 199 && x < 300) {
    pc_testmode_motor(x);
  }
  if (x > 299 && x < 400) {
    pc_testmode_analogRead(x);
  }
  if (x > 399 && x < 500) {
    pc_testmode_final(x);
  }
  Serial.println("pc_testmode end");
}

void pc_testmode_sd(int x) {
  bool file_check_bool = false;
  bool rtc_file_check_bool = false;
  int counti = 0;
  bool wx = true;  // while loop

  int y = 0;
  // Serial.println("sd  card test functions start");
  switch (x) {
    case 101:  // sd write
      {
        Serial.println("sd_card_panel_write();");
        test = reset;
        test += ".csv";

        sd_write(test.c_str(),
"Date,Panel1ADC,Panel2ADC,Panel3ADC");
      }
      break;

    case 102:  //  sd_read
      {
        Serial.println("sd_read();");
        test = reset;
        test += ".csv";
        sd_read(test.c_str());
      }
      break;
    case 103:  //  add row
      {
        Serial.println("add row;");
        test = reset;
        test += ".csv";
        sd_addrow(test.c_str(), date_hms,
panelA[0], panelA[1], panelA[2]);
      }
      break;
    case 104:  //  sd_filecheck
      {
        test = reset;
        test += ".csv";
        file_check_bool =
sd_filecheck(test.c_str());

Serial.print("sd_filename_checker_return_bool
=  ");
        Serial.println(file_check_bool);
      }
      break;
    case 105:  // sd_filedelete(test.c_str());
      {
        Serial.println("sd_filedelete(test.c_str());");

        // sd_filedelete( test.c_str());
      }
      break;
    case 106:  // sd write
      {
        sd_card_panel_write();
      }
      break;
    case 107:  // sd write
      {
```

```
    }
    break;

  case 108:  // sd write
    {
    }
    break;

  case 109:  // sd write
    {
      sd_card_file_storage(true, 5);
    }
    break;
  case 110:  // sd write
    {
      sd_card_file_storage(false, 0);
    }
    break;
  case 111:  // sd write
    {
    }
    break;
 }
}

void pc_testmode_motor(int x) {
 bool wx = true;  // while loop

 int y = 0;
 bool file_check_bool = false;
 bool eswtest = true;  // if this returns false
something went wrong
 int counti = 0;
 // Serial.println("sd  card test functions start");

 switch (x) {
  case 201:  // Serial.println("
tracker.Motor_identifier();");
    {
      tracker.Motor_identifier();
    }
    break;
  case 202:  // Serial.println(" Esw_check();");
    {
      tracker.move_Pitch_EswEscape(5000);
    }
    break;
  case 203:  // Serial.println(" Esw_display();");
    {
      tracker.Esw_display();
    }
    break;
  case 204:  // Serial.println("
motor_move_PITCH(true,1000,30);");
    {
      tracker.motor_move_PITCH(true, 1000,
30);
    }
    break;
  case 205:  // Serial.println("
motor_move_PITCH(false,1000,30);");
    {
      tracker.motor_move_PITCH(false, 1000,
30);
    }
    break;
  case 206:  // Serial.println("
motor_move_PITCH(true,20000,30);");
    {
      tracker.motor_move_PITCH(true, 40000,
30);
    }
    break;
  case 207:  // Serial.println("
motor_move_PITCH(false,20000,30);");
    {
      tracker.motor_move_PITCH(false, 40000,
30);
    }
    break;
  case 208:  // Serial.println("
tracker.move_Pitch_Esw(5000);");
    {
      eswtest = move_Yaw_EswEscape(5000);
    }
    break;


  case 209:  //
tracker.move_Pitch_RESETTER(8);
    {
      tracker.move_Pitch_RESETTER(8);
    }
    break;
  case 210:  // sd write
    {
      Serial.println("  motor_move_YAW(true,
20);");
      motor_move_YAW(true, 20);

      //tracker.motor_move_YAW(true,20);
    }
    break;
  case 211:  // sd write
    {
      Serial.println("  motor_move_YAW(false,
20);");
      motor_move_YAW(false, 20);
    }
    break;
  case 212:  // sd write
    {
      tracker.move_Pitch_RESETTER(8);
    }
    break;

  case 213:  //Serial.println("
motor_move_FREE_yaw(true, 20);");
```

```
    {

      Serial.print(" YAW              = ");
      Serial.println(digitalRead(dual_eswReed));
      Serial.println("
motor_move_FREE_yaw(true, 20);");
      motor_move_FREE_yaw(true, 100);
      Serial.print(" YAW              = ");
      Serial.println(digitalRead(dual_eswReed));
    }
    break;
  case 214: //   motor_move_FREE_yaw(false,
100);
    {
      Serial.print(" YAW              = ");
      Serial.println(digitalRead(dual_eswReed));
      Serial.println("
motor_move_FREE_yaw(false, 20);");
      motor_move_FREE_yaw(false, 100);
      Serial.print(" YAW              = ");
      Serial.println(digitalRead(dual_eswReed));
    }
    break;
  case 215:  // move_Yaw_EswEscape(5000);
    {
      move_Yaw_EswEscape(5000);
    }
    break;
  case 216:  //    yaw_full_step =
move_Yaw_RESETTER();
    {
      yaw_full_step =
move_Yaw_RESETTER();
    }
    break;
  case 217:  //
tracker.motor_move_PITCH(true, 100,
tracker.motorin1b);
    {
      tracker.motor_move_PITCH(true, 100,
tracker.motorin1b);
    }
    break;
  case 218:  //
tracker.motor_move_PITCH(false, 100,
tracker.motorin1b);
    {
      tracker.motor_move_PITCH(false, 100,
tracker.motorin1b);
    }
    break;
  case 219:  // sd write
    {
      tracker.motor_move_PITCH(true, 1000,
tracker.motorin1a);
    }
    break;
  case 220:  // sd write
    {
      tracker.motor_move_PITCH(false, 1000,
tracker.motorin1a);
    }
    break;
  case 221:  // sd write
    {
      Serial.println("  motor_move_YAW(true,
2000);");
      motor_move_YAW(true, 2000);
    }
    break;
  case 222:  // sd write
    {
      Serial.println("  motor_move_YAW(false,
2000);");
      motor_move_YAW(false, 2000);
    }
    break;
  case 223:  // sd write
    {
      tracker.move_Pitch_RESETTER(30);
    }
    break;
  case 224:  // sd write
    {
    }
    break;
  case 225:  // sd write
    {
    }
    break;
  case 226:  // sd write
    {
    }
    break;
 }
}

void pc_testmode_analogRead(int x) {
 bool file_check_bool = false;
 bool eswtest = true;  // if this returns false
something went wrong
 int counti = 0;
 bool wx = true;  // while loop

 int y = 0;
 // Serial.println("sd  card test functions start");
 switch (x) {
  case 301:  //DUAL LDR ONCE
    {
      int ldr_dual =
ldr_dual_C.ldr_dual_compare_V2();
    }

    break;
  case 302:  //DUAL LDR LOOP
    {
      int ldr_dual;
      ldr_dual_C.timed(true, 1);
```

```
      while (wx == true) {
        ldr_dual_C.timed(false, 1);
        if (Serial.available() > 0) {  // Check if
there is any input available
          int x = Serial.parseInt();   // Read the
user input as an integer
          Serial.print("User  input: ");
          Serial.println(x);   // Print the user input
          x = 0;
          wx = false;
        }
        if (ldr_dual_C.mytime1c >= 1000) {
          ldr_dual_C.timed(true, 1);
          ldr_dual =
ldr_dual_C.ldr_dual_compare_V2();
        }
      }
    }

    break;
  case 303:  //SINGLE LDR ONCE
    {
      int ldr_single  =
ldr_dual_C.ldr_single_compare();
    }

    break;
  case 304:  //SINGLE LDR LOOP
    {
      int ldr_single;
      ldr_dual_C.timed(true, 1);
      while (wx == true) {
        ldr_dual_C.timed(false, 1);
        if (Serial.available() > 0) {  // Check if
there is any input available
          int x = Serial.parseInt();   // Read the
user input as an integer
          Serial.print("User  input: ");
          Serial.println(x);   // Print the user input
          x = 0;
          wx = false;
        }
        if (ldr_dual_C.mytime1c >= 1000) {
          ldr_dual_C.timed(true, 1);
          ldr_single  =
ldr_dual_C.ldr_single_compare();
        }
      }
    }

    break;
  case 305:
    {
      ldr_dual_C.panel_measure();
      for (int i = 0; i < 3; i++) {
        Serial.print("Panel");
        Serial.print(i  + 1);
        Serial.print("  = ");
        Serial.println(ldr_dual_C.panelA[i]);
      }
    }
    break;
  case 306:  //Battery_control
    {
      ldr_dual_C.battery_control(1, true);
    }

    break;
  case 307:  //Battery_control
    {
      ldr_dual_C.battery_control(2, true);
    }

    break;
  case 308:  //Battery_control
    {
      ldr_dual_C.battery_control(3, true);
    }

    break;
  case 309:  //Battery_control
    {
      ldr_dual_C.battery_control(4, true);
    }

    break;
  case 310:  //Battery_control
    {
      ldr_dual_C.battery_control(5, true);
    }

    break;
  case 311:  //Battery_control
    {
      ldr_dual_C.battery_control(0, true);
    }

    break;
  case 312:  //Battery_control
    {
      ldr_dual_C.battery_control(6, true);
    }

    break;
  case 313:  //Battery_control
    {
      single_axis_ldr_move_test();
    }

    break;
  case 314:  //Battery_control
    {
      dual_axis_ldr_move_test();
    }

    break;
  case 315:  //Battery_control
    {
```

```
      single_axis_ldr_move_test();
      dual_axis_ldr_move_test();
     }

    break;
  }
}
void pc_testmode_final(int x) {
 switch (x) {
   case 401:  //bool singleaxis_final =
single_axis_ldr_move();
    {
     bool singleaxis_final =
single_axis_ldr_move();
    }
    break;
   case 402:  //bool singleaxis_final =
single_axis_ldr_move();
    {
     bool dualaxis_final =
dual_axis_ldr_move();
    }
    break;
   case 403:  //bool singleaxis_final =
single_axis_ldr_move();
    {
     rtc_update();
     Serial.print("myhour = ");
     Serial.println(myhour);
    }
    break;
   case 404:  //bool singleaxis_final =
single_axis_ldr_move();

    {
     rtc_update();
     Serial.print("date = ");
     Serial.println(date);
     Serial.print("date_dmy = ");
     Serial.println(date_dmy);
     Serial.print("date_hms = ");
     Serial.println(date_hms);
    }
    break;
   case 405:  //bool singleaxis_final =
single_axis_ldr_move();

    {
     digitalWrite(ldr_dual_C.relayHON,
HIGH);
    }
    break;
   case 406:  //bool singleaxis_final =
single_axis_ldr_move();

    {
     digitalWrite(ldr_dual_C.relayHON,  LOW);
    }
    break;
   case 407:  //bool singleaxis_final =
single_axis_ldr_move();

    {
     int availableMemory = freeMemory();
     Serial.print("Free  memory: ");
     Serial.print(availableMemory);
     Serial.println("  bytes");
    }
    break;
   case 408:  //bool singleaxis_final =
single_axis_ldr_move();

    {
     holderS = "arduino started";
     SD_system_event_recorder(holderS,
false);
    }
    break;

   case 409:  //bool singleaxis_final =
single_axis_ldr_move();

    {

     holderS = "LDR  check";
     SD_system_event_recorder(holderS, true);
    }
    break;
   case 410:  //bool singleaxis_final =
single_axis_ldr_move();

    {
     ldr_dual_C.battery_measure();
    }
    break;
   case 411:  //bool singleaxis_final =
single_axis_ldr_move();

    {
     battery_voltage();
    }
    break;
 }
}

bool single_axis_ldr_move_test() {

 bool fin = false;

 bool eswx = false;  //false is issue
 int single_axisR = 0;
 bool fin_ldr_fine = 0;
 bool r = false;

 while (!fin) {
  single_axisR =
ldr_dual_C.ldr_single_compare();
  switch (single_axisR) {
```

```
    case 0:
      {
       Serial.println("single_axis_ldr_move
ERROR RETURN 0");
       fin = true;
       r = false;
      }
      break;
    case 1:
      {
       eswx = tracker.motor_move_PITCH(true,
50, tracker.motorin1b);
       fin_ldr_fine = false;
       if (!eswx) {

         tracker.move_Pitch_EswEscape(5000);
       }
      }
      break;
    case 2:
      {
       eswx =
tracker.motor_move_PITCH(false, 50,
tracker.motorin1b);
       fin_ldr_fine = false;
       if (!eswx) {

         tracker.move_Pitch_EswEscape(5000);
       }
      }
      break;
    case 3:
      {
       Serial.println("single_axis_ldr_move
LDR fine waiting again");
       delay(1000);
       if (fin_ldr_fine) {
         fin = true;
         r = true;
       }
       fin_ldr_fine = true;
      }
      break;
  }
 }
 Serial.println("single_axis_ldr_move LDR fine
done");

 return r;
}

bool dual_axis_ldr_move_test() {
//CERTAINLY WORKING WITH SIMPLIER
SYSTEM
                        // yaw_full_step =
move_Yaw_RESETTER();
                        //
tracker.move_Pitch_RESETTER(tracker.motori
n1a);
 bool fin = false;
 int dual_axisR = 0;
 bool fin_ldr_fine = 0;
 bool r = false;
 bool eswx = false;


 while (!fin) {

  dual_axisR =
ldr_dual_C.ldr_dual_compare_V2();
  switch (dual_axisR) {
    case 0:
      {
       eswx = tracker.motor_move_PITCH(true,
200, tracker.motorin1a);
       fin_ldr_fine = false;

       if (!eswx) {

         tracker.move_Pitch_EswEscape(5000);
       }
      }
      break;
    case 1:
      {
       eswx =
tracker.motor_move_PITCH(false, 200,
tracker.motorin1a);
       fin_ldr_fine = false;
       if (!eswx) {


         tracker.move_Pitch_EswEscape(5000);
       }
      }
      break;
    case 2:
      {
       eswx = motor_move_YAW(true, 20);
       fin_ldr_fine = false;

       if (!eswx) {

         move_Yaw_EswEscape(5000);
       }
      }
      break;

    case 3:
      {
       eswx = motor_move_YAW(false, 20);
       fin_ldr_fine = false;

       if (!eswx) {

         move_Yaw_EswEscape(5000);
       }
      }
```

```
      break;
     case 4:
      {
       Serial.println("dual_axis_ldr_move LDR
fine waiting again");
       delay(1000);
       if (fin_ldr_fine) {
         fin = true;
         r = true;
       }
       fin_ldr_fine = true;
      }
      break;
   }
  }
  if (r == true) {
   Serial.println("dual_axis_ldr_move LDR fine
done");
   holderS = " dual_axis_ldr_move LDR fine
done;";
   SD_system_event_recorder(holderS, false);
  }

  return r;
}

void test_functions(int x) {  // x from user input
function
  switch (x) {
   case 111:
     {
      Serial.println("BUTTON  LIST ");
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("BUTTON  LIST");
      delay(2000);
      lcd.clear();
      lcd.setCursor(7, 0);
      lcd.print("112");
      lcd.setCursor(0, 1);
      lcd.print("MOTOR  PANIC ON");
      delay(2000);
      lcd.clear();
      lcd.setCursor(7, 0);
      lcd.print("113");
      lcd.setCursor(0, 1);
      lcd.print("MOTOR  PANIC OFF");
      delay(2000);
      lcd.clear();
      lcd.setCursor(7, 0);
      lcd.print("211");
      lcd.setCursor(0, 1);
      lcd.print("M. S. CW 1 SEC");
      delay(2000);
      lcd.clear();

      lcd.setCursor(7, 0);
      lcd.print("311");  //"FIRST VALUE  2 OR
3  CLCOKWISE  OR  COUNTER
CLOCKWISE"  SECOND  VALUE  "1 2 3 "
SINGLE,  DUAL  PITCH  AND  YAW
      lcd.setCursor(0, 1);
      lcd.print("M.S.  C-CW 1 SEC");
      delay(2000);
      lcd.clear();

      lcd.setCursor(7, 0);
      lcd.print("221");
      lcd.setCursor(0, 1);
      lcd.print("D.PITCH  C 1 SEC");
      delay(2000);
      lcd.clear();

      lcd.setCursor(7, 0);
      lcd.print("321");
      lcd.setCursor(0, 1);
      lcd.print("D.P  C-CW 1 SEC");
      delay(2000);
      lcd.clear();

      lcd.setCursor(7, 0);
      lcd.print("231");
      lcd.setCursor(0, 1);
      lcd.print("D.YAW  C 1 SEC");
      delay(2000);
      lcd.clear();

      lcd.setCursor(7, 0);
      lcd.print("331");
      lcd.setCursor(0, 1);
      lcd.print("D.YAW  C-CW 1SEC");
      delay(2000);
      lcd.clear();

      lcd.setCursor(7, 0);
      lcd.print("222");
      lcd.setCursor(0, 1);
      lcd.print("LDR & ESW CHECK");
      delay(2000);
      lcd.clear();

      lcd.setCursor(7, 0);
      lcd.print("333");
      lcd.setCursor(0, 1);
      lcd.print("PANEL  & BAT V");
      delay(2000);
      lcd.clear();
      lcd.noBacklight();
     }
     break;

   case 112:
     {
      Serial.println("MOTOR  PANIC ON");
      lcd.backlight();
      lcd.clear();
```

```
      lcd.setCursor(0, 0);
      lcd.print("MOTOR  PANIC ON");
      tracker.motorpanic = true;
      delay(2000);
      lcd.clear();
      lcd.noBacklight();
     }

     break;
   case 113:
     {
      Serial.println("MOTOR  PANIC OFF");
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("MOTOR  PANIC OFF");
      tracker.motorpanic = false;
      tracker.motorpanic_dualPitch = false;
      tracker.motorpanic_SinglePitch = false;
      tracker.motorpanic_Yaw = false;
      delay(2000);
      lcd.clear();
      lcd.noBacklight();
     }

     break;

   case 211:
     {
      Serial.println("MOTOR  PANIC OFF");
      tracker.motorpanic = false;
      tracker.motorpanic_dualPitch = false;
      tracker.motorpanic_SinglePitch = false;
      tracker.motorpanic_Yaw = false;
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("SINGLE  AXIS MOVE");
      bool tgbuttonsingle =
single_axis_ldr_move();
      if (tgbuttonsingle) {
       lcd.setCursor(0, 1);
       lcd.print("DONE  FINE ");
      } else {
       lcd.setCursor(0, 1);
       lcd.print("FAILED  ");
      }
      delay(2000);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("DUAL  AXIS MOVE");
      tgbuttonsingle = dual_axis_ldr_move();
      if (tgbuttonsingle) {
       lcd.setCursor(0, 1);
       lcd.print("DONE  FINE ");
      } else {
       lcd.setCursor(0, 1);
       lcd.print("FAILED  ");
      }
      delay(2000);
      lcd.clear();
      lcd.noBacklight();
     }
     break;
   case 212:
     {
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("BATTERY  SD");
      battery_voltage();
      delay(2000);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("RED = ");
      lcd.setCursor(8, 0);

lcd.print((ldr_dual_C.battery_voltage_array_V[
0]));
      lcd.setCursor(0, 1);
      lcd.print("GREEN = ");
      lcd.setCursor(10, 1);

lcd.print((ldr_dual_C.battery_voltage_array_V[
1]));
      delay(3000);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("BLUE  = ");
      lcd.setCursor(8, 0);

lcd.print((ldr_dual_C.battery_voltage_array_V[
2]));
      lcd.setCursor(0, 1);
      lcd.print("MOTOR  = ");
      lcd.setCursor(10, 1);

lcd.print((ldr_dual_C.battery_voltage_array_V[
3]));
      delay(3000);

      lcd.clear();
      lcd.noBacklight();
     }
     break;
   case 213:
     {
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("PANEL  SD");
      ldr_dual_C.panel_measure();
      sd_card_panel_write();
      delay(2000);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("TIME = ");
      lcd.setCursor(5, 0);
```

```
      lcd.print(date_hms);
      lcd.setCursor(0, 1);
      lcd.print("PANEL1  = ");
      lcd.setCursor(10, 1);
      lcd.print(ldr_dual_C.panelA[0]);
      delay(3000);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("PANEL2  = ");
      lcd.setCursor(10, 0);
      lcd.print(ldr_dual_C.panelA[1]);
      lcd.setCursor(0, 1);
      lcd.print("PANEL3  = ");
      lcd.setCursor(10, 1);
      lcd.print(ldr_dual_C.panelA[2]);
      delay(3000);
      lcd.clear();
      lcd.noBacklight();
     }
     break;
   case 311:
     {
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("DUAL  PITCH RESET");

      tracker.move_Pitch_RESETTER(8);
      delay(2000);

      lcd.clear();
      lcd.noBacklight();
     }
     break;
   case 312:
     {
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("DUAL  YAW RESET");

      yaw_full_step =
move_Yaw_RESETTER();
      delay(2000);
      lcd.clear();
      lcd.noBacklight();
     }
     break;
   case 313:
     {
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("SIN PITCH RESET");

      tracker.move_Pitch_RESETTER(30);
      delay(2000);
      lcd.clear();
      lcd.noBacklight();
     }
     break;
     /*
   case 221:
     {
      Serial.println("MOTOR  DUAL  AXIS
PITCH MOVE TRUE  1 SECONDS  ");
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("MOT. DUAL  PITCH");
      lcd.setCursor(0, 1);
      lcd.print("MOVE  TRUE 1 SEC");
      delay(2000);
      bool eswx = motor_move_PITCH(true,
1000, motorin1a);
      motor_kill_all();
      lcd.clear();
      lcd.setCursor(0, 0);
      Serial.println("MOTOR  MOVE FIN");
      lcd.print("MOTOR  MOVE FIN");
      if (eswx == false) {

       lcd.setCursor(0, 1);
       Serial.println("FAILED  ESW TRIP");
       lcd.print("FAILED  ESW TRIP");
       delay(2000);
      } else {
       lcd.setCursor(0, 1);
       lcd.print("OK.  NO ESW TRIP");
       delay(2000);
      }

      lcd.clear();
      lcd.noBacklight();
     }
     break;
   case 321:
     {
      Serial.println("MOTOR  DUAL  AXIS
PITCH MOVE FALSE  1 SECONDS  ");
      lcd.backlight();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("MOT. DUAL  PITCH");
      lcd.setCursor(0, 1);
      lcd.print("MOVE  FALSE 1SEC");
      delay(2000);
      bool eswx = motor_move_PITCH(false,
1000, motorin1a);
      motor_kill_all();
      lcd.clear();
      lcd.setCursor(0, 0);
      Serial.println("MOTOR  MOVE FIN");
      lcd.print("MOTOR  MOVE FIN");
      if (eswx == false) {

       lcd.setCursor(0, 1);
       Serial.println("FAILED  ESW TRIP");
```

```
        lcd.print("FAILED  ESW TRIP");
        delay(2000);
       } else {
        lcd.setCursor(0, 1);
        lcd.print("OK.  NO ESW TRIP");
        delay(2000);
       }

       lcd.clear();
       lcd.noBacklight();
      }
      break;
    case 231:

      {
       Serial.println("MOTOR  DUAL  YAW
MOVE  TRUE 1 SECONDS  ");
       lcd.backlight();
       lcd.clear();
       lcd.setCursor(0, 0);
       lcd.print("MOTOR  DUAL  YAW");
       lcd.setCursor(0, 1);
       lcd.print("MOVE  TRUE 1 SEC");
       delay(2000);
       bool eswx = motor_move_PITCH(true,
1000, motorin3a);
       motor_kill_all();
       lcd.clear();
       lcd.setCursor(0, 0);
       Serial.println("MOTOR  MOVE FIN");
       Serial.println("MOTOR  MOVE FIN");
       lcd.print("MOTOR  MOVE FIN");
       if (eswx == false) {

        lcd.setCursor(0, 1);
        Serial.println("FAILED  ESW TRIP");
        lcd.print("FAILED  ESW TRIP");
        delay(2000);
       } else {
        lcd.setCursor(0, 1);
        lcd.print("OK.  NO ESW TRIP");
        delay(2000);
       }

       lcd.clear();
       lcd.noBacklight();
      }
      break;
    case 331:


      {
       Serial.println("MOTOR  DUAL  YAW
MOVE  FALSE 1 SECONDS  ");
       lcd.backlight();
       lcd.clear();
       lcd.setCursor(0, 0);
       lcd.print("MOTOR  DUAL  YAW");
       lcd.setCursor(0, 1);
       lcd.print("MOVE  TRUE 1 SEC");
       delay(2000);
       bool eswx = motor_move_PITCH(false,
1000, motorin3a);
       motor_kill_all();
       lcd.clear();
       lcd.setCursor(0, 0);
       Serial.println("MOTOR  MOVE FIN");
       lcd.print("MOTOR  MOVE FIN");
       if (eswx == false) {

        lcd.setCursor(0, 1);

        Serial.println("FAILED  ESW TRIP");
        lcd.print("FAILED  ESW TRIP");
        delay(2000);
       } else {
        lcd.setCursor(0, 1);
        lcd.print("OK.  NO ESW TRIP");
        delay(2000);
       }

       lcd.clear();
       lcd.noBacklight();
      }
      break;



    case 222:
      {
       Serial.println("LDR  TEST START");
       lcd.backlight();
       lcd.clear();
       lcd.setCursor(0, 0);
       lcd.print("LDR TEST START");
       sensor_analog(numReadings,
numInitialReadingsToDiscard, ldr1S, ldr2S,
ldr1D, ldr2D, ldr3D, ldr4D);
       lcd.setCursor(0, 1);
       lcd.print("LDR READ DONE ");
       Serial.print("LDR1S =  ");
       Serial.print((sensoranalogA[0]));
       Serial.print("      LDR2S =  ");
       Serial.println((sensoranalogA[1]));
       for (int i = 2; i < 6; i++) {
        Serial.print("LDR  ");
        Serial.print(i  - 1);
        Serial.print("D  = ");
        Serial.print((sensoranalogA[i]));
        Serial.print("   ");
       }
       Serial.println("  ");
       esw_check(7);

       Serial.print("single_esw  =  ");
       Serial.print(esw_array[0]);
       Serial.print("    dual_esw1 =  ");
       Serial.print(esw_array[1]);
```

```
    Serial.print("dual_esw2 =  ");
    Serial.println(esw_array[2]);

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("S.A.esw");
    lcd.setCursor(9, 0);
    lcd.print((esw_array[0]));
    lcd.setCursor(0, 1);
    lcd.print("LDR1&2");
    lcd.setCursor(7, 1);
    lcd.print((sensoranalogA[0]));
    lcd.setCursor(12, 1);
    lcd.print((sensoranalogA[1]));
    delay(2000);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("DUAL  AXIS ");
    lcd.setCursor(0, 1);
    lcd.print("ESW1&2");
    lcd.setCursor(8, 1);
    lcd.print((esw_array[1]));
    lcd.setCursor(12, 1);
    lcd.print((esw_array[2]));
    delay(2000);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("LDR1&2");
    lcd.setCursor(8, 0);
    lcd.print((sensoranalogA[2]));
    lcd.setCursor(12, 0);
    lcd.print((sensoranalogA[3]));
    delay(2000);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("LDR3&4");
    lcd.setCursor(8, 0);
    lcd.print((sensoranalogA[4]));
    lcd.setCursor(12, 0);
    lcd.print((sensoranalogA[5]));
    delay(2000);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("LDR  TEST  END ");
    delay(2000);
    lcd.clear();
    lcd.noBacklight();
    Serial.println("LDR  TEST END  ");
   }
   break;

case 333:
   {
    Serial.println("PANEL  BAT V ");
    lcd.backlight();
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("PANEL  & BAT V ");
    Serial.println("start reading");
        lcd.setCursor(0, 1);
        lcd.print("start reading ");
        delay(2000);
        digitalWrite(relayHON,  HIGH);
        delay(100);
        sensor_analog(numReadings,
numInitialReadingsToDiscard, panelA1,
panelA2, panelA3, BatRed, BatGreen, BatBlue,
BatMotor);

        delay(100);
        digitalWrite(relayHON,  LOW);
        for (int i = 0; i < 3; i++) {

          panelA[i] = (1023 - (sensoranalogA[i]));
          panelV[i] = (((panelA[i])  * 5) * 5);
          panelVR[i] = ((panelV[i] * (R1 + R2)) /
R1);
          panelI[i] = ((panelVR[i]  * 1000) / (R1 +
R2));
        }
        for (int i = 3; i < 7; i++) {
          battery_voltage_array_A[i - 3] = (1023 -
(sensoranalogA[i]));
          battery_voltage_array_V[i - 3] =
(((battery_voltage_array_A[i - 3]) * 5) / 1023);
          battery_voltage_array_V[i - 3] =
((battery_voltage_array_V[i - 3]) - 0.52);  //little
bit tolerance for temporary solution
        }

        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("panelA1=");
        Serial.println("panelA1=");
        lcd.setCursor(11, 0);
        lcd.print(panelA[0]);
        Serial.println(panelA[0]);
        lcd.setCursor(0, 1);
        lcd.print("PANEL1  VR=");
        Serial.println("PANEL1  VR=");
        lcd.setCursor(12, 1);
        lcd.print(panelVR[0]);
        Serial.println(panelVR[0]);
        delay(4000);

        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("PANEL2  V=");
        Serial.println("PANEL2  V=");
        lcd.setCursor(11, 0);
        lcd.print(panelA[1]);
        Serial.println(panelA[1]);
        lcd.setCursor(0, 1);
        lcd.print("PANEL2  VR=");
        Serial.println("PANEL2  VR=");
        lcd.setCursor(12, 1);
        lcd.print(panelVR[1]);
        Serial.println(panelVR[1]);
```

```
delay(4000);

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("PANEL3  V=");
Serial.println("PANEL3  V=");
lcd.setCursor(11, 0);
lcd.print(panelA[2]);
Serial.println(panelA[2]);
lcd.setCursor(0, 1);
lcd.print("PANEL3  VR=");
Serial.println("PANEL3  VR=");
lcd.setCursor(12, 1);
lcd.print(panelVR[2]);
Serial.println(panelVR[2]);
delay(4000);

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("BAT RED V=");
Serial.println("BAT  RED V=");
lcd.setCursor(12, 0);
lcd.print(battery_voltage_array_V[0]);
Serial.println(battery_voltage_array_V[0]);
lcd.setCursor(0, 1);
lcd.print("BAT RED A=");
Serial.println("BAT  RED A=");
lcd.setCursor(12, 1);
lcd.print(battery_voltage_array_A[0]);
Serial.println(battery_voltage_array_A[0]);
delay(2000);

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("B GREEN V=");
Serial.println("B  GREEN V=");
lcd.setCursor(12, 0);
lcd.print(battery_voltage_array_V[1]);
Serial.println(battery_voltage_array_V[1]);
lcd.setCursor(0, 1);
lcd.print("B GREEN A=");
Serial.println("B  GREEN A=");
lcd.setCursor(12, 1);
lcd.print(battery_voltage_array_A[1]);
Serial.println(battery_voltage_array_A[1]);
delay(2000);

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("B. BLUE V=");
Serial.println("B. BLUE V=");
lcd.setCursor(12, 0);
lcd.print(battery_voltage_array_V[2]);
Serial.println(battery_voltage_array_V[2]);
lcd.setCursor(0, 1);
lcd.print("B. BLUE A=");
Serial.println("B. BLUE A=");
lcd.setCursor(12, 1);
lcd.print(battery_voltage_array_A[2]);
Serial.println(battery_voltage_array_A[2]);
delay(2000);

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("B MOTOR V=");
Serial.println("B  MOTOR V=");
lcd.setCursor(12, 0);
lcd.print(battery_voltage_array_V[3]);
Serial.println(battery_voltage_array_V[3]);
lcd.setCursor(0, 1);
lcd.print("B MOTOR A=");
Serial.println("B  MOTOR A=");
lcd.setCursor(12, 1);
lcd.print(battery_voltage_array_A[3]);
Serial.println(battery_voltage_array_A[3]);
delay(2000);

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("PANEL  & BAT V ");
Serial.println("PANEL  & BAT V ");
lcd.setCursor(0, 1);
lcd.print("DONE  READING");
Serial.println("DONE  READING");
delay(2000);
lcd.clear();

lcd.noBacklight();
}
break;
*/ }
return;
}

void userinput() {

resetArray();
lcd.backlight();
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("TEST  MODE ON");
lcd.setCursor(0, 1);
lcd.print("111 FOR LIST");
bool exit = false;
bool wait = true;

int buttonfilter = 0;

int ix = 0;
bool functionstage = false;

delay(2000);
lcd.clear();
resetArray();

while (exit == false) {
lcd.setCursor(0, 0);
lcd.print("waiting  for input");
```

```
    timed(true, 1);
    timed(true, 2);

    while (ix < 3) {
      timed(false, 1);
      if (mytime1c > 10000) {
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("TIMEOUT  EXIT");
        ix = 0;
        exit = true;
        break;
      }
      buttonarray[ix][0][0] =
digitalRead(button1);
      buttonarray[ix][1][0] =
digitalRead(button2);
      buttonarray[ix][2][0] =
digitalRead(button3);

      if ((buttonarray[ix][0][0]) == true ||
(buttonarray[ix][1][0]) == true ||
(buttonarray[ix][2][0]) == true) {
        lcd.clear();
        if ((buttonarray[ix][0][0]) == true) {
          Serial.println("button 1 pressed ");
          buttonarray[ix][0][1] = true;
          lcd.setCursor(8, 0);
          lcd.print(1);
        }
        if ((buttonarray[ix][1][0]) == true) {
          Serial.println("button 2 pressed ");
          buttonarray[ix][1][1] = true;
          lcd.setCursor(8, 0);
          lcd.print(2);
        }
        if ((buttonarray[ix][2][0]) == true) {
          Serial.println("button 3 pressed ");
          buttonarray[ix][2][1] = true;
          lcd.setCursor(8, 0);
          lcd.print(3);
        }

        lcd.setCursor(0, 0);
        lcd.print("button");
        lcd.setCursor(0, 1);
        lcd.print("pressed");
        delay(2000);
        lcd.clear();
        timed(true, 1);
        ix++;
        lcd.setCursor(0, 0);
        lcd.print("ready for ");
        lcd.setCursor(0, 1);
        lcd.print("next input ");
      }
    }
    if (ix >= 3) {
      Serial.println("button code cal ");
      wait = true;
      buttonfilter = 0;
      buttonfilter = buttonfilter +
(((buttonarray[0][0][1]) * 100) +
((buttonarray[1][0][1]) * 10) +
((buttonarray[2][0][1]) * 1));
      buttonfilter = buttonfilter +
(((buttonarray[0][1][1]) * 200) +
((buttonarray[1][1][1]) * 20) +
((buttonarray[2][1][1]) * 2));
      buttonfilter = buttonfilter +
(((buttonarray[0][2][1]) * 300) +
((buttonarray[1][2][1]) * 30) +
((buttonarray[2][2][1]) * 3));
      Serial.print("button code= ");
      Serial.println(buttonfilter);
      resetArray();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("button code =");
      lcd.setCursor(0, 1);
      lcd.print(buttonfilter);
      delay(2000);
      lcd.setCursor(0, 8);
      lcd.print("Y1/N3");
    }

    timed(true, 1);
    timed(true, 2);
    while (wait == true) {
      timed(false, 1);
      resetArray();
      buttonarray[0][0][2] = digitalRead(button1);
      buttonarray[0][2][2] = digitalRead(button3);
      if (buttonarray[0][0][2] == true) {
        wait = false;
        functionstage = true;
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("CONFIRMED  ");
        lcd.setCursor(0, 1);
        lcd.print("FUNCTION  PHASE ON");
        delay(2000);
        lcd.clear();
        timed(true, 1);
      }
      if (buttonarray[0][2][2] == true) {
        wait = false;
        ix = 0;
        resetArray();
        buttonfilter = 0;
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("RESETTING   ");
        lcd.setCursor(0, 1);
        lcd.print("RETURN  READING");
        delay(2000);
        lcd.clear();
        timed(true, 1);
```

```
    }

    if (mytime1c > 10000) {
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("TIMEOUT  EXIT");
      exit = true;
      lcd.clear();
      lcd.noBacklight();
      resetArray();
      return;
    }
   }
   if (functionstage == true) {

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("FUNCTION  STAGE  ");
    test_functions(buttonfilter);
    exit = true;
   }
  }

  lcd.clear();
  lcd.noBacklight();
  resetArray();
  return;
}

void handleEncoder() {

  encoderValue++;
}

//////////////////////////////TEST FUNCTIONS END
////////////////////////////

//////////////////////////////GENERIC FUNCTIONS
START ////////////////////////////

void resetArray();
void battery_voltage();
void toggleswchecker();
void timed(bool x, int y);
int freeMemory();



void resetArray() {
  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 3; j++) {
      for (int k = 0; k < 3; k++) {
        buttonarray[i][j][k] = 0;  // Assign 0 to each
element of the array
      }
    }
  }
  return;
}

void battery_voltage() {
  ldr_dual_C.battery_measure();
  holderS = "RED Battery V =";
  holderS +=
String((ldr_dual_C.battery_voltage_array_V[0])
);
  holderS += "  GREEN Battery V =";
  holderS +=
String((ldr_dual_C.battery_voltage_array_V[1])
);
  holderS += "  BLUE Battery V =";
  holderS +=
String((ldr_dual_C.battery_voltage_array_V[2])
);
  holderS += "  MOTOR Battery V =";
  holderS +=
String((ldr_dual_C.battery_voltage_array_V[3])
);
  SD_system_event_recorder(holderS, false);
}

void toggleswchecker()  {
  bool x = false;
  x = digitalRead(togglesw1);
  // Serial.print("togglesw1=  ");
Serial.println(x);
  if (x == true) {
    userinput();
  }

  return;
}

void timed(bool x, int y) {
  switch (y) {
    case 1:
      if (x == true) {
        mytime1a = millis();
      }
      mytime1b = millis();
      mytime1c = (mytime1b - mytime1a);
      break;
    case 2:
      if (x == true) {
        mytime2a = millis();
      }
      mytime2b = millis();
      mytime2c = (mytime2b - mytime2a);
      break;
    case 3:
      if (x == true) {
        mytime3a = millis();
      }
      mytime3b = millis();
      mytime3c = (mytime3b - mytime3a);
      break;
  }
  return;
```

```
}

extern unsigned int __heap_start;
extern void *__brkval;

int freeMemory() {
  int free_memory;
  if ((int)__brkval == 0)
    free_memory = ((int)&free_memory) -
((int)&__heap_start);
  else
    free_memory = ((int)&free_memory) -
((int)__brkval);
  return free_memory;
}


//////////////////////////////GENERIC FUNCTIONS
END ////////////////////////////
```

EK-9 20 Temmuz tarihleri arasında yapılan ölçümler çizelge

Çizelge 6.1. 20 Temmuz tarihleri arasında yapılan ölçümler

	Gerilim (V)			Akım (A)			Güç(W)		
Zaman	Sabit	Tek	Çift	Sabit	Tek	Çift	Sabit	Tek	Çift
07:00	1.82	3.75	5.22	0.063	0.130	0.181	0.12	0.49	0.94
07:15	2.46	4.28	5.59	0.085	0.148	0.193	0.21	0.63	1.08
07:30	3.07	4.81	6.02	0.106	0.166	0.208	0.33	0.80	1.25
07:45	3.69	5.32	6.41	0.128	0.184	0.222	0.47	0.98	1.42
08:00	4.28	5.81	6.74	0.148	0.201	0.233	0.63	1.17	1.57
08:15	4.81	6.47	7.37	0.166	0.224	0.255	0.80	1.45	1.88
08:30	5.34	7.08	8.03	0.185	0.245	0.278	0.99	1.74	2.23
08:45	5.86	7.55	8.66	0.203	0.261	0.300	1.19	1.97	2.60
09:00	6.37	8.31	9.36	0.220	0.288	0.324	1.40	2.39	3.03
09:15	6.86	8.68	9.60	0.237	0.300	0.332	1.63	2.61	3.19
09:30	7.33	9.05	9.89	0.254	0.313	0.342	1.86	2.83	3.38
09:45	7.70	9.54	10.17	0.266	0.330	0.352	2.05	3.15	3.58
10:00	8.11	10.01	10.79	0.281	0.346	0.373	2.27	3.47	4.03
10:15	9.07	10.93	11.42	0.314	0.378	0.395	2.85	4.14	4.52
10:30	10.11	11.57	12.37	0.350	0.400	0.428	3.54	4.63	5.29
10:45	11.05	12.39	13.35	0.383	0.429	0.462	4.23	5.31	6.16
11:00	11.87	13.27	14.19	0.411	0.459	0.491	4.88	6.09	6.96
11:15	12.47	13.94	14.51	0.431	0.482	0.502	5.38	6.73	7.29
11:30	13.33	14.41	14.86	0.461	0.499	0.514	6.15	7.19	7.64
11:45	13.90	14.96	15.19	0.481	0.518	0.526	6.69	7.75	7.98
12:00	14.62	15.39	15.76	0.506	0.533	0.545	7.39	8.20	8.60
12:15	14.90	15.46	15.76	0.516	0.535	0.545	7.69	8.27	8.60
12:30	15.19	15.70	15.87	0.526	0.543	0.549	7.98	8.53	8.71
12:45	15.68	15.87	16.25	0.543	0.549	0.562	8.51	8.71	9.14
13:00	15.89	16.17	16.25	0.550	0.560	0.562	8.73	9.05	9.14
13:15	15.58	16.17	16.09	0.539	0.560	0.557	8.40	9.05	8.96
13:30	15.39	15.91	16.17	0.533	0.550	0.560	8.20	8.75	9.05

13:45	14.99	15.76	16.11	0.519	0.545	0.557	7.77	8.60	8.98
14:00	14.72	15.72	15.93	0.509	0.544	0.551	7.50	8.55	8.78
14:15	14.45	15.54	15.68	0.500	0.538	0.543	7.23	8.35	8.51
14:30	14.00	15.11	15.44	0.485	0.523	0.534	6.78	7.90	8.24
14:45	13.61	14.72	15.13	0.471	0.509	0.523	6.41	7.50	7.92
15:00	13.27	14.33	14.76	0.459	0.496	0.511	6.09	7.11	7.54
15:15	12.69	13.84	14.60	0.439	0.479	0.505	5.57	6.63	7.37
15:30	11.96	13.31	14.39	0.414	0.460	0.498	4.95	6.13	7.17
15:45	11.08	12.57	13.98	0.383	0.435	0.484	4.24	5.47	6.76
16:00	10.46	11.89	13.78	0.362	0.412	0.477	3.79	4.90	6.57
16:15	9.79	11.36	12.82	0.339	0.393	0.443	3.31	4.47	5.68
16:30	9.25	11.08	12.16	0.320	0.383	0.421	2.96	4.24	5.12
16:45	8.62	10.48	11.34	0.298	0.363	0.392	2.57	3.80	4.45
17:00	7.98	10.05	10.63	0.276	0.348	0.368	2.21	3.50	3.91
17:15	7.35	9.27	10.15	0.254	0.321	0.351	1.87	2.98	3.57
17:30	6.59	8.48	9.79	0.228	0.293	0.339	1.50	2.49	3.31
17:45	5.94	7.74	9.52	0.205	0.268	0.329	1.22	2.07	3.14
18:00	5.12	6.94	8.99	0.177	0.240	0.311	0.91	1.67	2.80
18:15	4.65	6.29	8.48	0.161	0.217	0.293	0.75	1.37	2.49
18:30	4.07	5.73	7.86	0.141	0.198	0.272	0.57	1.14	2.14
18:45	3.54	5.12	7.23	0.123	0.177	0.250	0.43	0.91	1.81
19:00	2.95	4.52	6.57	0.102	0.157	0.227	0.30	0.71	1.49

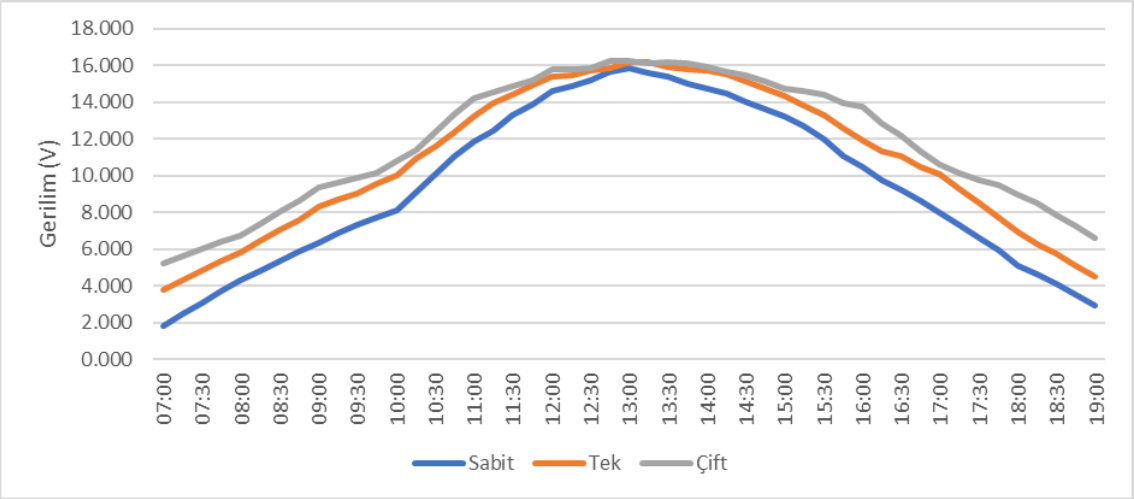
EK-10 16 Ağustos tarihleri arasında yapılan ölçümler çizelge

Çizelge 6.2. 16 Ağustos arasında yapılan ölçümler

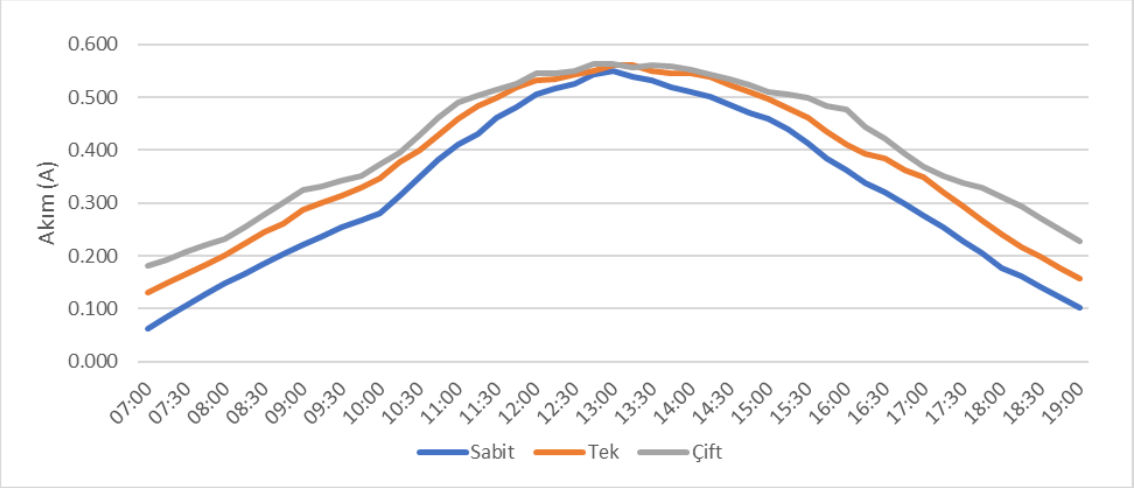
	Gerilim (V)			Akım (A)			Güç(W)		
Zaman	Sabit	Tek	Çift	Sabit	Tek	Çift	Sabit	Tek	Çift
07:00	1.801	3.746	5.097	0.062	0.130	0.176	0.112	0.486	0.899
07:15	2.436	4.278	5.507	0.084	0.148	0.191	0.205	0.633	1.049
07:30	3.009	4.893	6.019	0.104	0.169	0.208	0.313	0.828	1.253
07:45	3.603	5.445	6.489	0.125	0.188	0.225	0.449	1.026	1.457
08:00	4.278	5.896	6.755	0.148	0.204	0.234	0.633	1.203	1.579
08:15	4.831	6.633	7.206	0.167	0.230	0.249	0.808	1.522	1.797
08:30	5.323	7.083	8.086	0.184	0.245	0.280	0.980	1.736	2.262
08:45	5.937	7.431	8.598	0.205	0.257	0.298	1.220	1.911	2.558
09:00	6.305	8.188	9.396	0.218	0.283	0.325	1.376	2.320	3.055
09:15	7.001	8.536	9.540	0.242	0.295	0.330	1.696	2.522	3.149
09:30	7.472	9.089	9.908	0.259	0.315	0.343	1.932	2.859	3.397
09:45	7.697	9.417	10.133	0.266	0.326	0.351	2.050	3.068	3.553
10:00	8.004	10.195	10.993	0.277	0.353	0.380	2.217	3.596	4.182
10:15	9.151	10.829	12.549	0.317	0.375	0.434	2.897	4.058	5.449
10:30	10.154	11.566	13.020	0.351	0.400	0.451	3.567	4.629	5.865
10:45	10.850	12.406	13.716	0.375	0.429	0.475	4.073	5.325	6.509
11:00	12.160	13.143	13.777	0.421	0.455	0.477	5.116	5.977	6.568
11:15	12.549	13.388	14.248	0.434	0.463	0.493	5.449	6.202	7.024
11:30	13.552	13.654	14.248	0.469	0.472	0.493	6.355	6.451	7.024
11:45	13.695	14.105	13.757	0.474	0.488	0.476	6.490	6.884	6.548
12:00	14.309	14.330	14.289	0.495	0.496	0.494	7.085	7.105	7.065
12:15	14.125	14.002	13.859	0.489	0.485	0.480	6.904	6.784	6.646
12:30	14.330	14.289	13.982	0.496	0.494	0.484	7.105	7.065	6.764
12:45	14.432	14.596	13.450	0.499	0.505	0.465	7.207	7.372	6.259
13:00	14.412	14.616	13.593	0.499	0.506	0.470	7.187	7.392	6.393
13:15	14.064	14.371	14.371	0.487	0.497	0.497	6.844	7.146	7.146
13:30	14.084	14.637	14.268	0.487	0.506	0.494	6.864	7.413	7.045
13:45	13.879	14.043	14.064	0.480	0.486	0.487	6.666	6.824	6.844
14:00	13.941	14.268	14.105	0.482	0.494	0.488	6.725	7.045	6.884
14:15	13.143	14.330	14.002	0.455	0.496	0.485	5.977	7.105	6.784
14:30	12.733	14.002	13.470	0.441	0.485	0.466	5.610	6.784	6.278
14:45	12.938	14.064	13.552	0.448	0.487	0.469	5.792	6.844	6.355
15:00	12.815	13.961	14.330	0.443	0.483	0.496	5.682	6.745	7.105
15:15	12.631	13.777	14.514	0.437	0.477	0.502	5.520	6.568	7.289
15:30	12.242	13.511	14.371	0.424	0.468	0.497	5.186	6.316	7.146
15:45	11.116	12.590	14.187	0.385	0.436	0.491	4.276	5.485	6.964
16:00	10.706	11.771	13.818	0.370	0.407	0.478	3.966	4.794	6.607
16:15	9.990	11.321	12.815	0.346	0.392	0.443	3.453	4.434	5.682
16:30	9.335	10.973	12.406	0.323	0.380	0.429	3.015	4.166	5.325

16:45	8.680	10.706	11.362	0.300	0.370	0.393	2.607	3.966	4.467
17:00	7.984	9.949	10.563	0.276	0.344	0.366	2.206	3.425	3.861
17:15	7.513	9.089	10.113	0.260	0.315	0.350	1.953	2.859	3.539
17:30	6.674	8.618	9.621	0.231	0.298	0.333	1.541	2.570	3.203
17:45	5.916	7.656	9.417	0.205	0.265	0.326	1.211	2.028	3.068
18:00	5.159	7.083	9.007	0.179	0.245	0.312	0.921	1.736	2.807
18:15	4.749	6.448	8.536	0.164	0.223	0.295	0.780	1.439	2.522
18:30	4.156	5.589	7.943	0.144	0.193	0.275	0.598	1.081	2.183
18:45	3.562	5.118	7.390	0.123	0.177	0.256	0.439	0.906	1.890
19:00	3.009	4.463	6.407	0.104	0.154	0.222	0.313	0.689	1.421

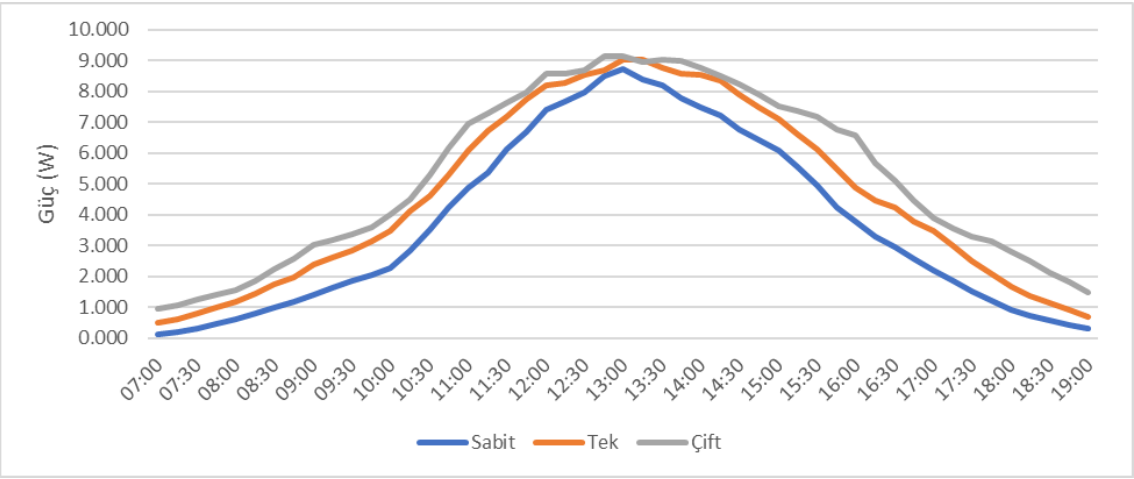
EK-11 20 Temmuz tarihleri arasında yapılan ölçümler grafikler



Şekil 7.1. 20 Temmuz deneysel verileri akım grafiği

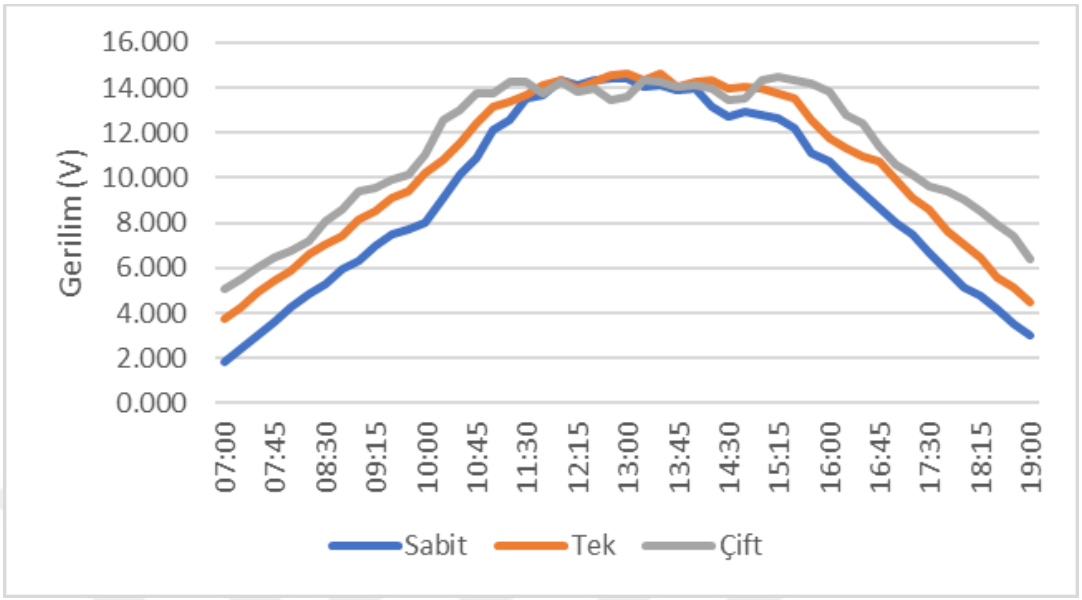


Şekil 7.2. 20 Temmuz deneysel verileri Gerilim grafiği

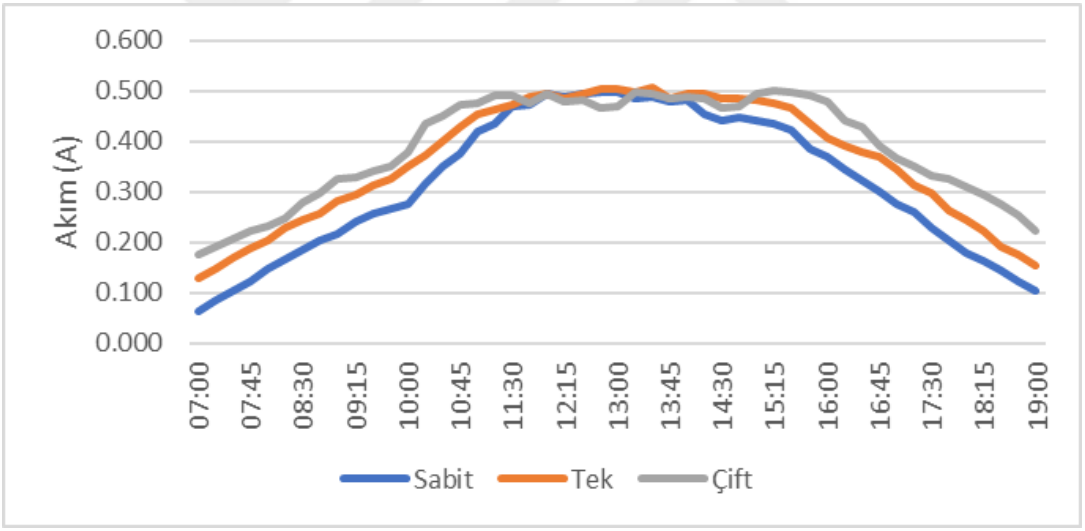


Şekil 7.3. 20 Temmuz deneysel verileri güç grafiği

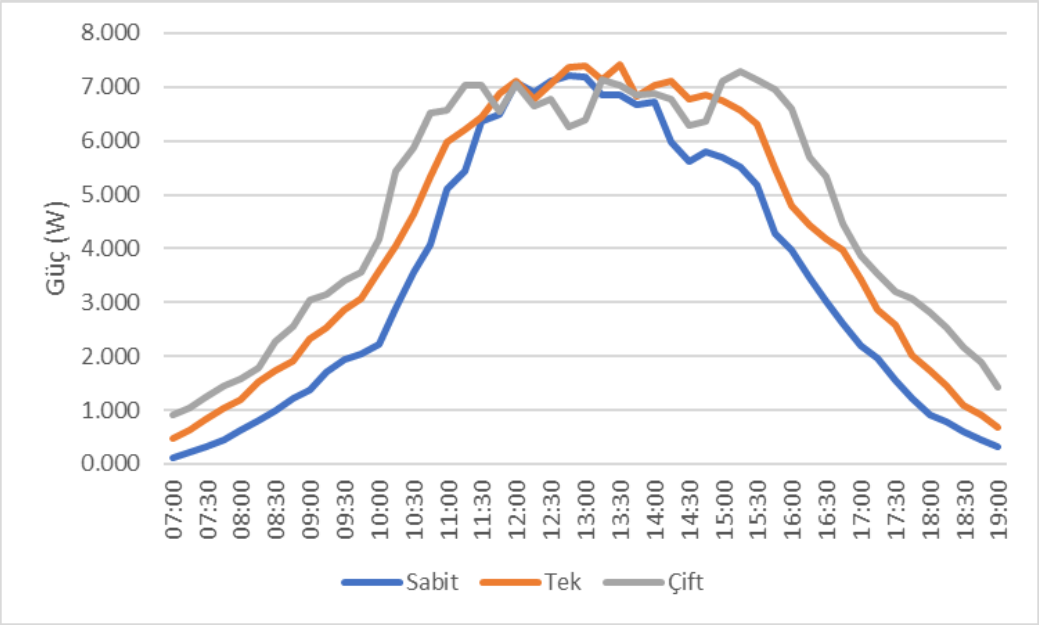
EK-12 16 Ağustos tarihleri arasında yapılan ölçümler grafikler



Şekil 7.4. 16 Ağustos deneysel verileri gerilim grafiği



Şekil 7.5. 16 Ağustos deneysel verileri akım grafiği



Şekil 7.6. 16 Ağustos deneysel verileri güç grafiği