İSTANBUL TEKNİK ÜNİVERSİTESİ

ELEKTRİK-ELEKTRONİK FAKÜLTESİ

KÜPSAT UYDUSU VE FAYDALI YÜKÜNÜN GERÇEKLENMESİ

LİSANS BİTİRME TASARIM PROJESİ

Emre ERBUĞA

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ BÖLÜMÜ

Tez Danışmanı: Doç. Dr. Metin YAZGI

OCAK 2017

İSTANBUL TEKNİK ÜNİVERSİTESİ

ELEKTRİK-ELEKTRONİK FAKÜLTESİ

KÜPSAT UYDUSU VE FAYDALI YÜKÜNÜN GERÇEKLENMESİ

LİSANS BİTİRME TASARIM PROJESİ

Emre ERBUĞA

040120192

Tez Danışmanı: Doç. Dr. Metin YAZGI

Bölüme Teslim Edildiği Tarih: 03 Ocak 2017

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ BÖLÜMÜ

OCAK 2017

ÖNSÖZ

Çalışmalarım sırasında yardımlarını esirgemeyen, saygıdeğer proje danışman Hocam Sayın Doç. Dr. Metin Yazgı'ya en içten teşekkürlerimi sunarım. Ayrıca proje çalışmalarım boyunca maddi ve manevi desteklerinden dolayı Hasbi Sevinç, Mert Kahyaoğlu ve İTÜ Roket Takımı bireylerine teşekkürlerimi sunarım.

Tüm eğitim hayatım boyunca üzerimde büyük emekleri olan tüm saygıdeğer öğretmenlerime, en önemlisi bugünlere gelmemde büyük emeği olan bana maddi ve manevi her türlü destek veren başta Annem, Babam ve Abim olmak üzere tüm aile bireylerime teşekkürü bir borç bilirim.

Ocak 2017 Emre ERBUĞA

İÇİNDEKİLER

Sayfa

ÖNSÖZ……..…….....iii

KISALTMALAR……..……....vi

ŞEKİL LİSTESİ……...vii

1. GİRİŞ……...1

2. AMAÇ VE HEDEF……3

3. ÖN ARAŞTIRMA……...4

4. SİSTEMİN İHTİYAÇLARININ BELİRLENMESİ……...5

4.1 Mikrodenetleyici…. 5

4.1.1 Atmega 328p 5

4.2 Haberleşme Modülü 6

4.2.1 Xbee Pro S2B 6

4.2.2 Xbee Explorer Dongle 7

4.3 Sensörler 8

4.4 Yedekleme Sistemi (SD Kart) 8

4.5 Gps 9

4.5.1 Gps Nedir? 9

4.6 Lojik Seviye Dönüştürücü (Logic Level Converter) 11

4.7 Güç ve Regülatör Katı 12

4.7.1 Batarya Seçimi 12

4.7.2 3v3 Regülatör 13

4.7.3 5V Regülatör 14

4.8 Güneş Paneller ve Enerji Üretimi 15

5. PCB TASARIMI……...……16

5.1 Elektronik Güç Kartı PCB (EPS) 17

5.2 İşlemci Kartı PCB (OBC) 19

6. YAZILIMSAL TASARIM……...……21

6.1 Mikrodenetleyici tarafı 21

6.2 Faydalı yük 23

6.3 Yer istasyonu 26
7. GERÇEKLENEN KÜPSAT……...…….27
7.1 Sistemin Test Çıktıları 28
7.2 Sistemin Ağırlık Bütçesi 32
7.2.1 Ana Sistem 32
7.2.2 Haberleşme Modülü 33
7.2.3 Kamera 33
7.2.4 Batarya 34
7.2.5 Dış İskelet (3D Baskı) 34
7.2.6 Diğer 35
8. SİSTEMİN İYİLEŞTİRMESİNE YÖNELİK ÖNERİLER…….36
9. SONUÇ……...…….....37
KAYNAKLAR……..38
EKLER……...……...42
ÖZGEÇMİŞ……...…….....52

KISALTMALAR

OBC	**:** On Board Computer
EPS	**:** Electrical Power System
PCB	**:** Printed Circuit Board
SMPS	**:** Switched-Mode Power Supply
USB	**:** Universal Serial Bus
UART	**:** Universal Asynchronous Receiver Transmitter
RX/TX	**:** Receiver / Transceiver
LEO	**:** Low Earth Orbit
ITU-R	**:** International Telecommunication Union – Radio
OS	**:** Operating System
GPS	: Global Positioning System

ŞEKİL LİSTESİ

Sayfa

Şekil 4.1 Atmega 328p[9] ... 6
Şekil 4.2 Xbee Pro S2B [10] ... 7
Şekil 4.3 Xbee Explorer Dongle [11] ... 7
Şekil 4.4 Adafruit Sensör Kartı [12] ... 8
Şekil 4.5 İşlemci Devre Kartı Üstündeki Micro SD Kart Soketi ... 9
Şekil 4.6 Adafruit Ultimate GPS Breakout [14] ... 10
Şekil 4.7 BSS138 NMOS Anahtarlama Karakteristikleri [16] ... 11
Şekil 4.8 Lojik Seviye Dönüştürücü Network [15] ... 12
Şekil 4.9 PCB Üstünde Gerçeklenmiş Hali ... 12
Şekil 4.10 Li-po 11v1 25C 1050mAh ... 12
Şekil 4.11 Pololu D24V25F3 [19] ... 13
Şekil 4.12 D24V25F3 Verim-Çıkış Akımı Grafiği, Farklı Giriş Voltajılarına Göre [19] ... 13
Şekil 4.13 Pololu D24V50F5 [20] ... 14
Şekil 4.14 D24V50F5 Verim-Çıkış Akımı Grafiği, Farklı Giriş Voltajılarına Göre [20] ... 14
Şekil 4.15 Güneş Paneli Şarj Devresi İçin Bırakılan Opsiyonel Boşluk ... 15
Şekil 5.1 Alt Sistemler Arası Güç Yolları ... 16
Şekil 5.2 Alt Sistemler Arası Haberleşme Protokolleri ve Yolları ... 17
Şekil 5.3 Diptrace'te Tasarım Esnasında EPS Üstten Görünüm ... 18
Şekil 5.4 Diptrace'te Tasarım Esnasında EPS Alttan Görünüm ... 18
Şekil 5.5 EPS'in Gerçeklenmiş Hali ... 19
Şekil 5.6 Diptrace'te Tasarım Esnasında OBC Üstten Görünüm ... 19
Şekil 5.7 Diptrace'te Tasarım Esnasında OBC Alttan Görünüm ... 20
Şekil 5.8 OBC'in Gerçeklenmiş Hali ... 20
Şekil 6.1 Mikrodenetleyici Kod Akış Diyagramı ... 22
Şekil 6.2 Raspberry Pi için yapılan güncellemeler ve yeniden başlatma ... 23
Şekil 6.3 Raspberry Pi 3 UART Pinleri RX-TX ... 23
Şekil 6.4 ALT0 Modunda TxD ve RxD ... 24
Şekil 6.5 Raspberry Pi 3 Çalışan Kodun Akış Diyagramı ... 25
Şekil 6.6 Yer İstasyonu Akış Diyagramı ... 26
Şekil 7.1 OBC, EPS ve Raspberry Pi birbirlerine bağlı ... 27
Şekil 7.2 OBC, EPS ve Raspberry Pi diğer açıdan ... 27
Şekil 7.3 Uydunun Tam Kapasite Çalışırken ... 28
Şekil 7.4 Uydu Tam Kapasite Çalışırken, 3 Boyutlu Baskılı Mekanik Kabında ... 29
Şekil 7.5 Yer İstasyonunda, Uydudan Gelen Verileri ... 30
Şekil 7.6 Uydudan Gönderilip Yer İstasyonunda Üretilen Fotoğraf, (100,80) Piksel ... 30
Şekil 7.7 Fotoğrafın Fouirer Analizi Yapılıp Yerde Üretilen Fotoğraf (100,80) ... 31
Şekil 7.8 Uyduda Çekilen Fotoğrafın Aslı (1000,800), Yaklaşık 143 KB ... 31
Şekil 7.9 Ana Sistemin Ağırlığı ... 32
Şekil 7.10 Haberleşme Modülü Ağırlığı ... 33
Şekil 7.11 Kamera Ağırlığı ... 33
Şekil 7.12 Batarya Ağırlığı ... 34
Şekil 7.13 Dış İskelet Ağırlığı ... 34

ÖZET

Küp uydu özel amaçlar için tasarlanmış uydudur. Normal yörünge uydularına göre çok daha düşük maliyet ve teknolojiler ile geliştirilebilir. Gelişen teknoloji ile uydu teknolojine sahip ülkelerin gelişim oranları çok daha hızlı yükselmektedir. Normal uyduların yörünge sistemlerine yerleştirilmesi için çok daha büyük roket yapısına ve roket rampa altyapısına ihtiyaç duyulmaktadır. Günümüzde bu teknolojiye sahip oldukça az ülke bulunmaktadır. Uydu sistemlerini yörüngeye çıkarma maliyetleri taşınan sistemin hacmi ve daha önemlisi ağırlığıyla doğrudan etkilenmektedir[1]. Dolayısıyla bu sistemlerin maliyetlerinden yanından daha düşük boyutlarda, belirli görevler için tasarlanmış küpsatların maliyet ve sistem tasarımlarının daha az karmaşıklığı ve hızlı bir şekilde gerçeklemesi nedeniyle tercih edilmektedir. Başta Amerika Birleşik Devletleri olmak üzere Avrupa ve Japonya'nın da içerisinde bulunduğu teknolojik ülkeler Küp Uydu yapımına önem vermektedir[2].

Küp uydu sistemi ve faydalı yükünün tasarımında açık kaynaklı sistemler kullanılmasına özen gösterildi. Raspberry Pi 3 ve ona bağlı kamera sistemi, küp uydunun faydalı yükünü oluşturmaktadır.

Raspberry Pi 3 üzerinde yer istasyonundan gelen komutlara göre önceden tanımlanan algoritmalar işletilmektedir. Bunlardan bazılarına örnek olarak; fotoğraf çekmek, bu fotoğrafı gri skala kodlayıp yer istasyonuna kablosuz olarak aktarmak, bu çekilen fotoğrafın fouirer analizini yapmak ve analiz edilen fotoğrafı kablosuz olarak yer istasyonuna yollamak gibi işlevleri bulunmaktadır. Bu gibi yüksek performans gerektiren işlemler dolayı gömülü Linux işletim sistemi çalıştıran Raspberry Pi 3 tercih edilmiştir.

Anahtar kelimeler: Küpsat, uydu, Raspberry Pi 3, gömülü Linux işletim sistemi

ABSTRACT

Cubesat is satellite designed for special purposes. It can be improved with much lower cost and technologies according to the normal orbit satellite. Growth rates of countries that have satellite technology is rising much faster with the development of technology. Much larger rocket ramp structure and the rocket engine to in place on orbit system of normal satellites. Today, there are a few countries that have this technology. The cost of orbiting satellite systems is directly affected by the volume and more important weight of the system being transported. Technological countries as especially United State, Europe and Japan attach importance to Cubesat structure.

At the design stage, the cubesat system and its payload were taken care of using open source hardwares. The Raspberry Pi 3 and its attached camera systems are payload of satellite.

On Raspberry Pi 3, the predefined algorithms are operated according to commands from the ground station. Some examples of these are; Taking photographs, encoding the photographs on a gray scale and transferring them wirelessly to the ground station, performing a fourier analysis of the photographs taken, and sending the analyzed photographs wirelessly to the ground station. Raspberry Pi 3, which runs the embedded Linux operating system, is preferred because of such high performance requirements.

Key words: Cubesat, satellite, Raspberry Pi 3, Embedded Linux OS

1. GİRİŞ

Amerika Birleşik Devletleri'ndeki Stanford ve Califonia Polytechic Üniversitelerinin ortaklaşa başlattıkları çalışmalar ile küpsat(Cubesat) tasarımları başlamış ve belirli standartlara getirilmiştir. Bunlar 1U 10x10x10 cm, 2U 20x10x10 cm, 3U 30x10x10 cm boyutlarında standartlaştırılmıştır. Küpsatlar genellikle Dünya'dan 500km yüksekliğe kadar olan, alçak dünya yörüngesinde(LEO) görevini yapmaktadır. Bu sistemler uzayda belirli görevleri yapmak amacıyla geliştirilmiştir. Belirlenen esas amaç ileri teknoloji olmadan da bu uyduların yörüngelere yerleştirilmesini sağlamaktır. Küçük bütçeler ile yörüngelerde belirli araştırma veya ticari görevler yapılabilmektedir. Bir kilogramın altında ve 1U küpsatı fırlatma maliyeti yaklaşık 30.000 $'ın altındadır[3].

Türkiye'de son 10 yıldaki uydu projeleri ile tüm Dünya'da "Uydu Geliştiren ve Üreten" sınıfında yer almaktadır. RASAT ve Göktürk-2 2014 yılı itibarı ile başarılı bir şekilde tamamlanmış yerli uydu projeleridir[4]. Son olarak Göktürk-1 yakın yörünge gözlem uydumuz da yörüngeye başarıyla yerleştirilmiştir. Ülkemizdeki bu çalışmalardan etkilenen bizler gelecekte bu projelere teknik alt yapıyı şimdiden sağlamak için en temel küp uydu tasarım gereksinimlerini öğrenmek, irdelemek ve yeni tasarım önerileri ile bu sistemlerin gerçeklemesine katkıda bulunmak amaçlanmaktadır.

Küp uydunun tasarımına başlanmadan önce küp uydu teknolojilerinin gittiği yön incelendi. Bu ön araştırmanın ardından bazı görevler kurgulandı. Bu görevlerin yanında faydalı yük olarak gerçek zamanlı işletim sistemi çalıştıran Raspberry Pi-3 kullanıldı. Faydalı yük tarafında sinyal işleme, fotoğraf çekme, Fouirer analizi vb. yoğun işlem gücü gerektiren işlemler yapıldı. Uydunun görevleri yerine getirebilmek için gerekli alt sistemlerin nasıl oluşacağı konusunda araştırmaya devam edildi. Belirlenen alt elektronik sistemlerin, güç yönetimi kartının ve faydalı yükün uyumlu bir şekilde tasarlanmasına dikkat edildi. Sadece elektronik alt sistemler değil aynı zamanda mekanik ve diğer sistemlerin elektronik sistemlere entegrasyonu noktasında tasarımda bazı farklılıkları yapıldı. Tüm küp uydu elektronik alt sistemi tasarlandıktan sonra baskı devre(PCB) tasarımı ile parçaların yerleşimi ve konumlanması yapıldı. Sistem gereksinimleri üstüne çift taraflı olan farklı iki baskı

devre kartı olarak tasarlandı. Bu baskı devreler soket aracılığıyla birbirine dikeyde girebilmekte ve bu sokete Raspberry Pi-3 kartı da aynı soket aracılığıyla paralel olarak bağlanmaktadır. En üstteki kart(EPS) güç birimleri, şarj devresi için gerekli alan ve regülatör sisteminin bağlantısı olarak tasarlandı. Orta kart(OBC) uydunun mikrokontrollör ve sensör birimlerine yönelik tasarlandı. Alt kart ise Raspberry Pi-3 kartına ayrıldı. Tasarımı yapılan bu baskı devreler(PCB) ürettirildi. Üretilen baskı devre ile parçalar lehimlenerek birleştirildi. Gerçeklenen uydu sistemi modüler olarak programlanarak hatalı parçalara çözüm üretimi hızlandırıldı. Yer istasyonu tarafı için yazılımsal geliştirme yapıldı. Uydu ile yer istasyonu arasında iletişim kurularak, kurgulanan görevler gerçeklenmiş sistem üzerinde denendi.

2. AMAÇ VE HEDEF

Yapılan bu çalışma ile ülkemizdeki uydu sistemlerine yeni farklı bir bakış açısı getirmektir. Teorik olarak tasarlanan sistemin gerçeklemesi yapılarak, teorik ile pratikteki farklılıklar incelenmiştir.

Küp uydu ile faydalı yük olan açık donanım(Open Hardware) Raspberry pi-3 ile yüksek işlem gücü kabiliyeti kazanması sağlanmıştır. Küp uydu mikroişlemci ve üzerindeki sensörler ile basınç, sıcaklık, ivme, konum, manyetik alan(3 eksen), açısal hız(3 eksen), ivme(3 eksen), ışık akısı ve giriş voltajını ölçmektedir. Aynı zamanda micro-SD kart birimi ile elde edilen veriler kayıt altına alınmaktadır. Mikrodenetleyici ile haberleşme girişi mevcuttur. Projede yakın mesafe kablosuz haberleşmek için Xbee 2.4 GHz modül kullanılmıştır. Bu modül aracılığıyla yere bu sensör verileri her saniye düzenli bir şekilde kablosuz olarak gönderilmektedir. Yörüngeye oturacak sistemde 410-420 MHz bandı düşünülmektedir[5]. Fakat haberleşme kısmı bu raporda detaylıca incelenmemektir. Senaryo olarak; küp uyduya yer istasyonundan önceden tanımlanmış görevler kablosuz olarak iletilmekte, bu veriyi alan küp uydu, işlemcisiyle bu veriyi değerlendirip eğer faydalı yükü ilgilendiren veri ise bu veriyi faydalı yüke iletmektedir. Faydalı yük bu veri gelince içinde belirlenen görevlerden biri olan fotoğraf çekip bunu uydunun mikroişlemcisine yollayıp mikroişlemciden daha sonra kablosuz haberleşme modülüyle yer istasyonuna aktarılmaktadır. Yer istasyonu tarafından alınan piksel değerleriyle aktarılan fotoğraf tekrar oluşturulmaktadır. Sistemin haberleşme ve tüm sensör birimlerinin 8-bit işlemci üzerinde halledilip yüksek yoğunlukta işlem gücü gerektiren işlemleri ise faydalı yük üzerinde halledilmesi amaçlanmıştır. Bu sayede sistemin iç yapısı daha kontrol edilebilir ve güvenilir bir sistem olması hedeflenmekte, faydalı yük tarafında doğabilecek sorunlardan dolayı tüm sistemin etkilenmesini engellemek amaçlanmaktadır.

3. ÖN ARAŞTIRMA

NASA'nın küpsatlarla ilgili güncel haberleri ve teknolojik gelişmelerini yayınladığı site üzerinde litaratür taraması yapıldı[6].

GomSpace ve Clyde gibi küpsatlar için parça ve alt sistemler üreten endüstriyel firmaların web siteleri üzerinden ürünleri ve yazdıkları makaleler incelendi[7].

ITU-R yayınladığı band kullanım kuralları ve detayları incelendi[5].

İTÜ tarafından tasarlanmış küp uyduların özellikleri öğrenildi. Bu konuda bilgi alındı.

Küpsat ve cansat ile ilgili basılmış geliştirmeyle ilgili makaleler incelendi[1-2][26-27].

Raspberry Pi 3 ile ilgili araştırma yapıldı. Uydunun mikrodenetleyicisiyle haberleşmesine yönelik araştırmalar yapıldı. Ayrıca Raspberry Pi 3 ile yüksek performanslı işlemlerin uygulamasına yönelik araştırma yapıldı. OpenCV görüntü işleme kütüphanesinin dokümantasyonu incelendi[8].

PCB dizayn ve üretim konusunda araştırma yapılıp en uygun fiyatlı ve kaliteli yer araştırıldı[22].

4. SİSTEMİN İHTİYAÇLARININ BELİRLENMESİ

Sistemin parçaları ön araştırma ve geçmiş yıllarda cansat projesinde elde edilen projenin sonucunda kararlaştırıldı.

4.1 Mikrodenetleyici

Küpsatın en kritik parçalarından biri olan mikrokontrolör; işlem hızı, geliştirme ortamı, kütüphane çeşitliliği, güç tüketimi ve sensör birimleriyle haberleşme kanallarının yeterliliği göz önünde bulundurularak seçilecektir.

4.1.1 Atmega 328p

Düşük güç tüketimine sahiptir. CMOS 8-bit, 16 MHz saat hızı, 32KBytes flash hafıza, 1 KBytes EEPROM, 2 KBytes iç SRAM' e sahiptir.

Pin dağılımı ve Çevresel haberleşme birimleri:

1 * TTL UART(aynı zamanda FTDI çipine bağlı programlarken flash hafızaya yazım sırasında kullanıyor), 1 * I2C, 1 * SPI, 6 * PWM, 8 * ADC, 14 * GPIO çıkışı mevcuttur.

Ağırlık: 5gr.

Çevresel koşullara karşı direnci:

-40°C - 105°C koşullar arasında çalışabildiği söylenmektedir. Bu değerlere güvenlik payı verilmelidir.

EEPROM'da veri saklama 85°C'de 20 yıl, 25°C'de ise 100 yıl olarak verilmiştir.

Güç Tüketimi:

1 MHz, 1.8V, 25°C'de

Aktif: 0.2mA, Güç kapalı: 0.1µA

Projemiz için yeterli, kolay bulunabilir, hafif, düşük güç tüketimine sahip ve ucuz olan Şekil 4.1'de görülen Atmega 328p mikrokontrollöründen yana tercih yapılmıştır[9].

Şekil 4.1 Atmega 328p[9]

4.2 Haberleşme Modülü

4.2.1 Xbee Pro S2B

Proje kapsamında kablosuz haberleşme için 2.4GHz'de çalışan Xbee modülleri arasından Xbee Pro2SB seçilmiştir. Yer istasyonu ve uydu tarafında aynı modül kullanılmıştır. 115200 Baud rate seçilmiştir. Xbee ile birlikte FTDI sürücüsüyle birlikte bir breakout board kullanılmıştır. Bu kart Xbee verilerini mikrodenetleyiciye RX-TX(UART) üzerinden atabilmektedir. Sunum sırasında demo amaçlı bu modül seçilmiştir.

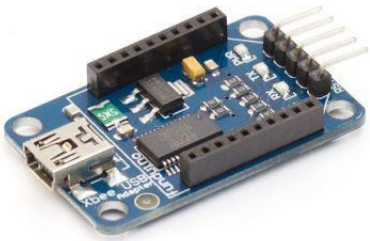
Uydunun LEO versiyonunda 410-420 MHz arası bandı kullanan farklı haberleşme birimi kullanabileceği düşünülmektedir. ITU-R regülasyonları ve atmosfer bastırma oranlarına göre en uygun aralık seçilebilir[5].

Menzil : 1500m (yönlendirilmiş anten ile) [10]
Ağırlık: 6gr.
Çevresel koşullara karşı direnci: -40°C ile +85°C arası, 0-95% nem
Alıcı Hassasiyeti : -102 dBm
Güç Tüketimi: 3.3V @ 295mA (Uydu tarafında veriyi yollama anında) Saniyede yaklaşık 1 adet 50bytelık veri paketi yollarken DC olarak 8V giriş katında 80mA çekmektedir.

Şekil 4.2 Xbee Pro S2B [10]

4.2.2 Xbee Explorer Dongle

XBee Explorer Dongle kartı BM Robotics tarafından üretilen oldukça kullanışlı ve çok yönlü bir haberleşme kartıdır. XBee Explorer Dongle, FT232RL USB-Seri TTL dönüştürücü temelli olup, XBee kablosuz modülleri ile bilgisayarınızı haberleştirmek ve XBee modülleri programlamak için kullanılan bir ara bağlantı kartıdır[11]. Bu kart Seri 1, Seri 2.5, standart ve pro gibi tüm Xbee modülleri ile beraber kullanılabilir. Kart bilgisayara takıldıktan sonra sanal bir COM port oluşmaktadır. Bu COM port üzerinden X-CTU programı ile XBee modülünüzü programlanmış. Daha sonra hem uydu tarafında mikrodenetleyeci ile haberleşmek için kullanılmış hem de yer istasyonu tarafından bilgisayara gelen verinin xbee'den aktarımı sırasında kullanılmıştır.



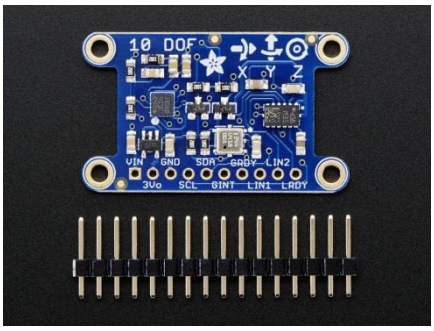
Şekil 4.3 Xbee Explorer Dongle [11]

4.3 Sensörler

Uydunun yönelim ve dünya ile yaptığı açıları takip etmek gerekmektedir. Eğer uyduya itki (propulsion) sistemi eklenecek ise bu verilerin ışığında hareketi mümkündür. 3 eksende manyetik alan, 3 eksende ivme ve 3 eksende gyro sensörleri sayesinde kinamatik denklemleri elde edilip uydunun dünya çevresindeki yönelimi hakkında bilgi sahibi olunur. Bu sayede itki sistemi kullanılacaksa uydunun yüzü sürekli olarak Dünya'ya veya bir yıldız galaksisine odaklanması sağlanabilir.

Sistemde Şekil 4.4'te görülen Adafruit tarafından tasarlanan farklı üreticilerin ürettiği sensörlerin toplandığı kart(breakout board) kullanılmıştır. Bu kart ile mikroişlemci I^2C ile haberleşmektedir. Aşağıda belirtilen her sensör entegresinin farklı bir I^2C adresi vardır. Bu sayede SDA ve SCL hattı üzerinden ekstra hatta gerek kalmadan senkron olarak iletişim kurulabilmektedir.

Bu kart üzerindeki sensörlerin özellikleri şöyledir [12]:
L3GD20H – 3 eksen Jiroskop
LSM303 – 3 eksen ivme ölçer, 3 eksen manyetik alan ölçer
BMP180 – Basınç ölçer
Ağırlık: 2.8gr.
Çevresel koşullara karşı direnci: -30°C ile +85°C, 0-95% nem
Güç Tüketimi: yaklaşık 1mA

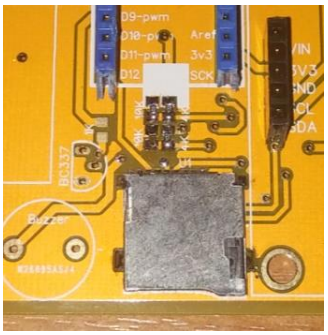


Şekil 4.4 Adafruit Sensör Kartı [12]

4.4 Yedekleme Sistemi (SD Kart)

Projede gerçek zamanlı veri kopmalarına karşı ve ya bazı ayarların sd kart üzerinde saklaması ve görev sırasında bu saklanan verilerin kullanılması istenebilir. Bu

nedenden dolayı esnek bir şekilde tasarıma gidilmiştir. Micro Sd kart'a uyumlu işlemci devre kartı tasarımına eklenmiştir.



Şekil 4.5 İşlemci Devre Kartı Üstündeki Micro SD Kart Soketi

4.5 Gps

4.5.1 Gps Nedir?

GPS, ingilizce Global Positioning System kelimesinin kısaltmasıdır. GPS'in Türkçe anlamı ise Küresel Yer Belirleme Sistemi veya Küresel Konumlama Sistemi'dir. GPS, düzenli olarak kodlanmış bir bilgi yollayan bir uydu ağıdır ve uydularla arasındaki mesafeyi ölçerek dünya üzerindeki konumu tespit etmektir. GPS uygulamaları ilk defa askeri amaçlar için Amerika Birleşik Devletleri (ABD) tarafından kurulmuştur. Bu sistem ABD tarafından askerlerin yön bulmasında, askeri çıkartmalarda, roket atışlarında kullanılması hedeflenmiştir. Sistem yaklaşık 12 milyon dolara kurulmuş ve 1980'li yıllarda sivil amaçlı kullanıma açılmıştır[13].

GPS'in temel çalışma prensibi sürekli yayın yapan Dünya çevresinde dönen uyduların yayını sayesinde olur. GPS alıcısı içindeki işlemci çekirdeği farklı GPS uydularından gelen sinyal seviyeleri ve süreleri ile özel algoritmalar işleterek, yörüngesi belirli olan GPS uydularına göre kendi konumunu uzayda bulabilmesiyle sonuçlanmaktadır.

$$Mesafe = Sinyalin\ Geliş\ Süresi * Işık\ hızı$$

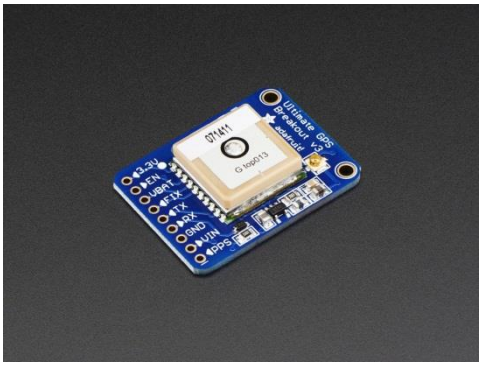
Burada atmosferdeki elektromanyetik dalgaların kırınımları, hava koşulları ve geçişlerine bağlı değişiklikler olmaktadır.

Bir uydudan sinyalin dünyaya ulaşma süresi ortalama 0.06 saniyedir. Saniyenin binde birinde oluşacak bir hata ortalama 300 km'lik bir sapma meydana getirecektir. Bu hataya engel olmak için 4 adet uydu kullanılarak sinyalin geliş süresindeki hata minimuma indirilinceye kadar ölçüm yapılır.

Bu modül bir çok ihtiyaca aynı anda cevap verebilen oldukça gelişmiş bir üründür. Bu üründe MTK3339 chipset kullanılmaktadır. 66 kanal üzerinde 22 farklı uydu takibini hassas bir şekilde (-165 dB takip yeteneği) gerçekleştirebilmektedir. Modül üzerinde, -165 dB hassasiyette çalışabilen dahili seramik anten bulunmaktadır. Ancak daha güçlü bir anten kullanmak istendiğinde, harici anten bağlanma özelliği sayesinde üzerinde uFL anteni bulunan ve 3V ile çalışan aktif bir GPS anteni kullanılabilir. Modül üzerine takılan anteni otomatik olarak algılar ve onu kullanmaya başlar. Çoğu anten üzerinde SMA konnektörü bulunmaktadır. Bu antenleri kullanmak için uFL SMA dönüştürücülerden kullanılabilir[14].

Saniyede 10 konum güncellemesi ile yüksek hızlı kayıt ve takip işlemlerini gerçekleştirebilmektedir. Ayrıca navigasyon sırasında güç tüketimi düşüktür[14].

Modülün üzerinde 3.3V regülatörü bulunduğu için 3.3 – 5V aralığında bir güç kaynağı ile rahatça kullanılabilir. ENABLE pini sayesinde herhangi bir mikrokontrolcü üzerinden veya basit bir anahtar sayesinde açma kapama yapılabilir. Kartın arka kısmında yer alan CR1220 pil yuvası sayesinde RTC'yi çalışır durumda tutarak "warm start" olanağı ve üzerindeki LED'in yanmasını sağlar. Kart üzerindeki LED, uyduları ararken 1Hz frekansında yanıp söner, bulduğu zaman ise 15 saniyede bir yanıp sönmektedir[14].



Şekil 4.6 Adafruit Ultimate GPS Breakout [14]

Yukarıda sayılan birçok özelliğinden dolayı uydumuzda GPS modülü olarak Adafruit Ultimate GPS Breakout kullanıldı. Bu modülle tarih, saat, enlem ve boylam bilgileri alındı. LEO yörüngesinde görev yapacak uydumuzda GPS maalesef modifikasyon

yapılmadığı takdirde çalışmayacaktır. Fakat sistemde gösterilmek amacıyla GPS konabilecek şekilde tasarlanmıştır.

4.6 Lojik Seviye Dönüştürücü (Logic Level Converter)

Uydumuzda Atmega328p ile Raspberry pi 3 uart üzerinden haberleşmektedir. Fakat lojik seviyeleri atmega'da 5V iken Raspberry Pi'da 3V3'tür. Bu yüzden voltaj bölücü veya Lojik seviye dönüştürücü alt sistem kurmamız gerekmektedir. Alt sistemimizde Sparkfun şirketine ait basit bir tasarım kullanılmıştır[15]. Tasarımda BSS138 NMOS ile düşük Vgs voltajlarında akım sürülüp voltaj dönüşümü yapılabilmiştir. NMOS'un Şekil 4.7'de görüleceği üzere anahtarlama karakteristikleri verilmiştir[16].

Switching Characteristics (Note 2)

$t_{d(on)}$	Turn–On Delay Time	V_{DD} = 30 V, I_D = 0.29 A,		2.5	5	ns
t_r	Turn–On Rise Time	V_{GS} = 10 V, R_{GEN} = 6 Ω		9	18	ns
$t_{d(off)}$	Turn–Off Delay Time			20	36	ns
t_f	Turn–Off Fall Time			7	14	ns

Şekil 4.7 BSS138 NMOS Anahtarlama Karakteristikleri [16]

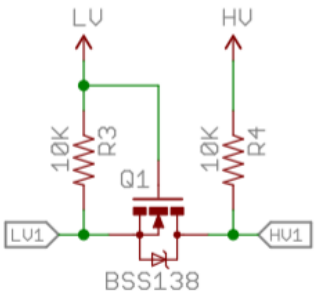
$$t_{ON} = t_{d(on)} + t_r = 11.5\,ns$$

$$t_{OFF} = t_{d(off)} + t_f = 27ns$$

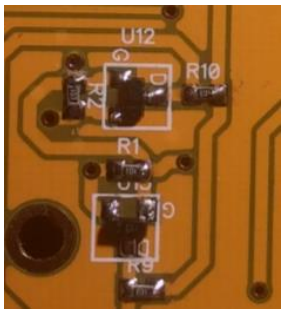
$$t_{ON} + t_{OFF} = 38.5\,ns \rightarrow \sim 25.9MHz$$

Kullanacağımız uart hattının maksimum 25.9 MHz hızı desteklemektedir. Biz ise bunun oldukça altını saniyede 9600 sembol gönderir bir şekilde kullanacağız. Sistemin ilerleyen dönemde bu hattının hızlandırılması söz konusu olabilir. Raspberry Pi ve Atmega 115.2K sembol/saniye hıza kadar desteklemektedir. Bu noktada da sıkıntı oluşmayacağı aşikârdır. PCB yollarının kapasitif ve resistif etkisiyle bir alçak geçiren filtre görevi görmektedir. Yüksek hızda iletişimlerde bu da dikkate alınmalıdır.

Lojik seviye dönüştürücü network'u Şekil 4.8'de görülmektedir. Direnç, HV(5V), LV(3V3) ve BSS138'den oluşmaktadır. PCB üzerinde gerçeklenmiş hali de Şekil 4.9'da gözükmektedir.



Şekil 4.8 Lojik Seviye Dönüştürücü Network [15]



Şekil 4.9 PCB Üstünde Gerçeklenmiş Hali

4.7 Güç ve Regülatör Katı

4.7.1 Batarya Seçimi

Küp uydularda Li-ion bataryalar yanında nikel katyum(Ni-Cd) pilleri de kullanılabilmektedir[17]. Küp uydumuzda Li-po 11v1 25C 1050mAh kullandık. Elimizdeki imkânlar böyle seçim yapmaya itti. Fiziksel gerçeklemede sistem bu batarya ile 30 dakika aralıksız tüm güçte sıkıntısız bir şekilde çalışmıştır.

Çevresel koşullara karşı direnci: 10°C ile 45°C

Şekil 4.10 Li-po 11v1 25C 1050mAh

4.7.2 3v3 Regülatör

Sistemimizde 3v3 ve 5V kanalları alt sistemlerimiz tarafından kullanılacaktır. Bu yüzden son derece verimli bir regülatör seçmeliyiz. Bu noktada anahtarlamalı modda güç regülatörü (SMPS) kullanılmıştır[18].

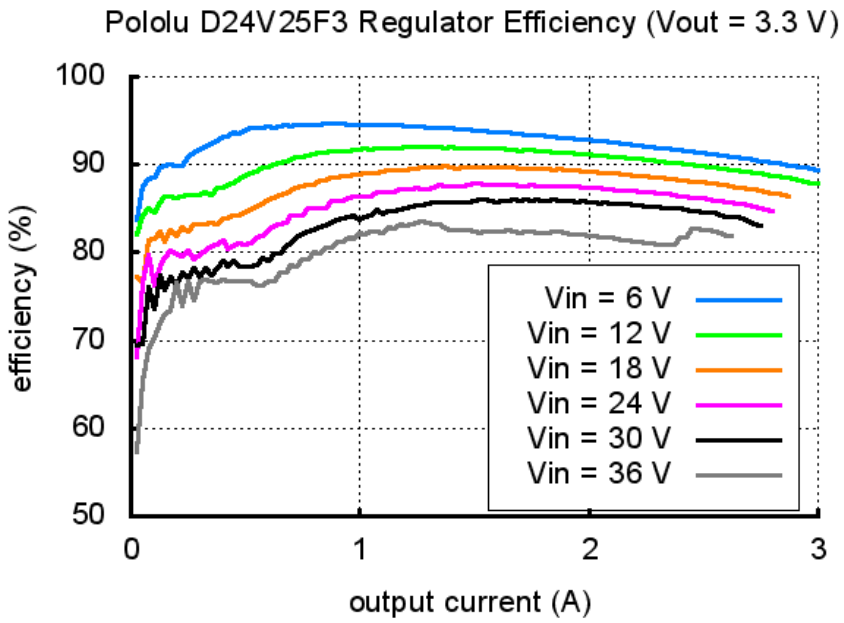
Seçimimiz piyasada bulabileceğimiz Pololu D24V25F3'den yana kullandık. Verimi %90'ların üzerindedir. Uydunun özellikle güç kayıplarını minamilize etmek gerekmektedir. Termal akışın havasız ortamda ışıma ve uydu içindeki aktarımın katı iletim yoluyla olduğu düşünülünce verim sistemin geneli itibariyle çok önemlidir.

Giriş: 4.5 V - 38 V

Çıkış: 3V3

Maksimum desteklenen 2.5 A'dir[19].

Şekil 4.11 Pololu D24V25F3 [19]



Şekil 4.12 D24V25F3 Verim-Çıkış Akımı Grafiği, Farklı Giriş Voltajlarına Göre [19]

4.7.3 5V Regülatör

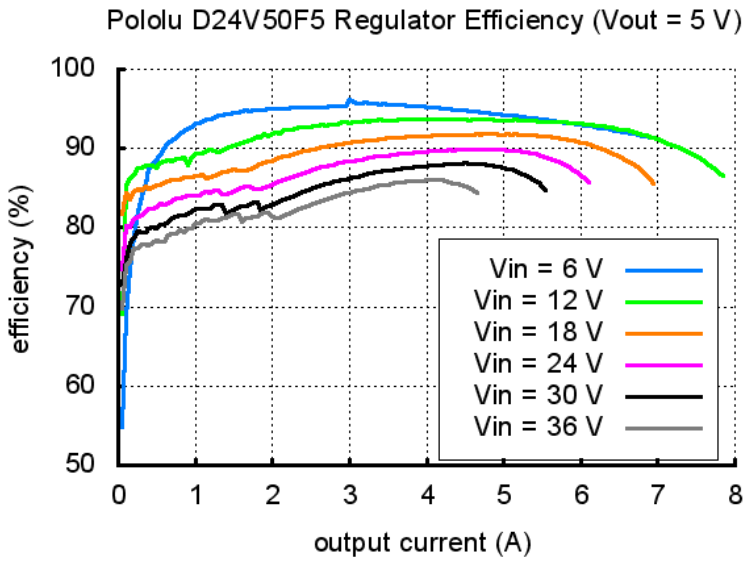
Sistemimizde 5V kanalıyla özellikle yüksek miktarda güç faydalı yüke iletilecektir. Bu yüzden son derece verimli bir regülatör seçmeliyiz. Bu noktada anahtarlamalı modda güç regülatörü (SMPS) kullanılmıştır[18].

Seçimimiz piyasada bulabileceğimiz Pololu D24V50F5'den yana kullandık. Verimi %90'ların üzerindedir. Uydunun özellikle güç kayıplarını minilimize etmek gerekmektedir. Termal akışın havasız ortamda ışıma veya katı iletim yoluyla olduğu düşünülünce verim sistemin geneli itibariyle çok önemlidir.

Giriş: 5v5 - 38 V

Çıkış: 5V [20]

Şekil 4.13 Pololu D24V50F5 [20]



Şekil 4.14 D24V50F5 Verim-Çıkış Akımı Grafiği, Farklı Giriş Voltajlarına Göre [20]

4.8 Güneş Paneller ve Enerji Üretimi

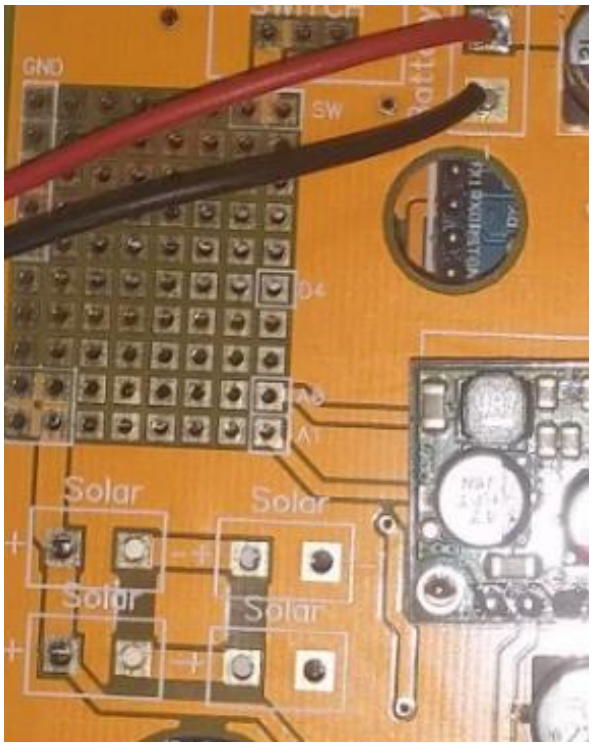
Sistem için uygun solar panel seçilmemiştir. Fakat dış yüzeyinde yaklaşık olarak

$$Güneş\ Panel\ Kullanılabilicek\ Maks.\ Alan$$
$$= 10cm * 10cm * 6(yüzey\ sayısı) = 600cm^2$$

Bu alandan açılır haberleşme sisteminin antenleri veya sabit yönlü antenlerin kapladığı alan, ışıma yoluyla ısı atımının yapılacağı alan, aktif kontrol sistemlerinin itki üreteçlerinin nozüllerinin oluşturacağı alan, ışık ve yıldız takip sistemlerinin kaplayacağı alan ve bunun gibi sistemlerin detaylı alan analizi ve kullanılacak güneş panellerinin alanlarının dikkatli hesabı gerekmektedir.

Her bir güneş paneli ekstra ağırlık ve dolayısıyla ek maliyet gerektirmektedir[3]. Bu noktada optimum çözüm aramak gerekir.

PCB üstünde Şekil 4.15'te görüleceği üzere, güç ve regülatör kartında güneş panellerinden şarj için özel bir alan boşta bırakılmış, gerekirse bu alan detaylı analizler sonucunda gerekli elektronik parçalarla kurgulanıp enerji üretimi ve şarj ünitesi kolayca yerleştirilebilir.

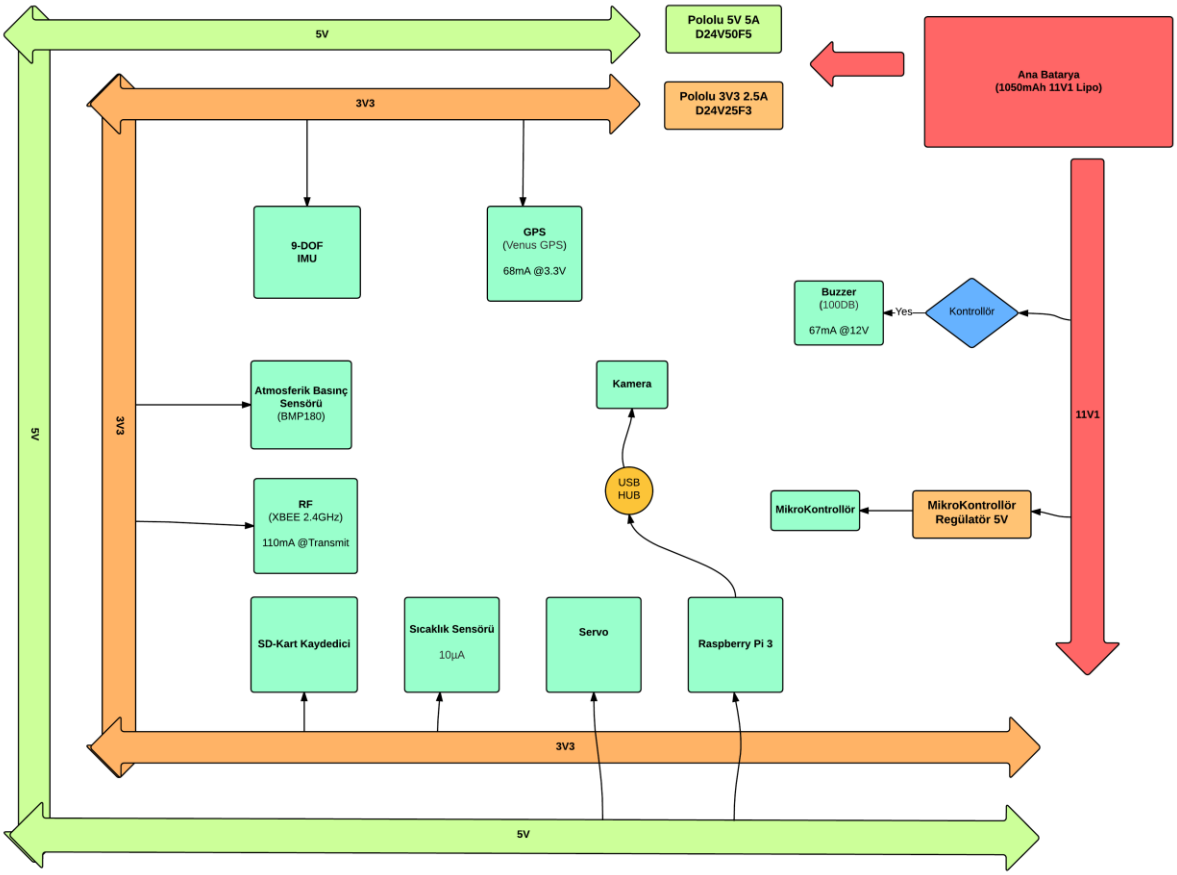


Şekil 4.15 Güneş Paneli Şarj Devresi İçin Bırakılan Opsiyonel Boşluk

5. PCB TASARIMI

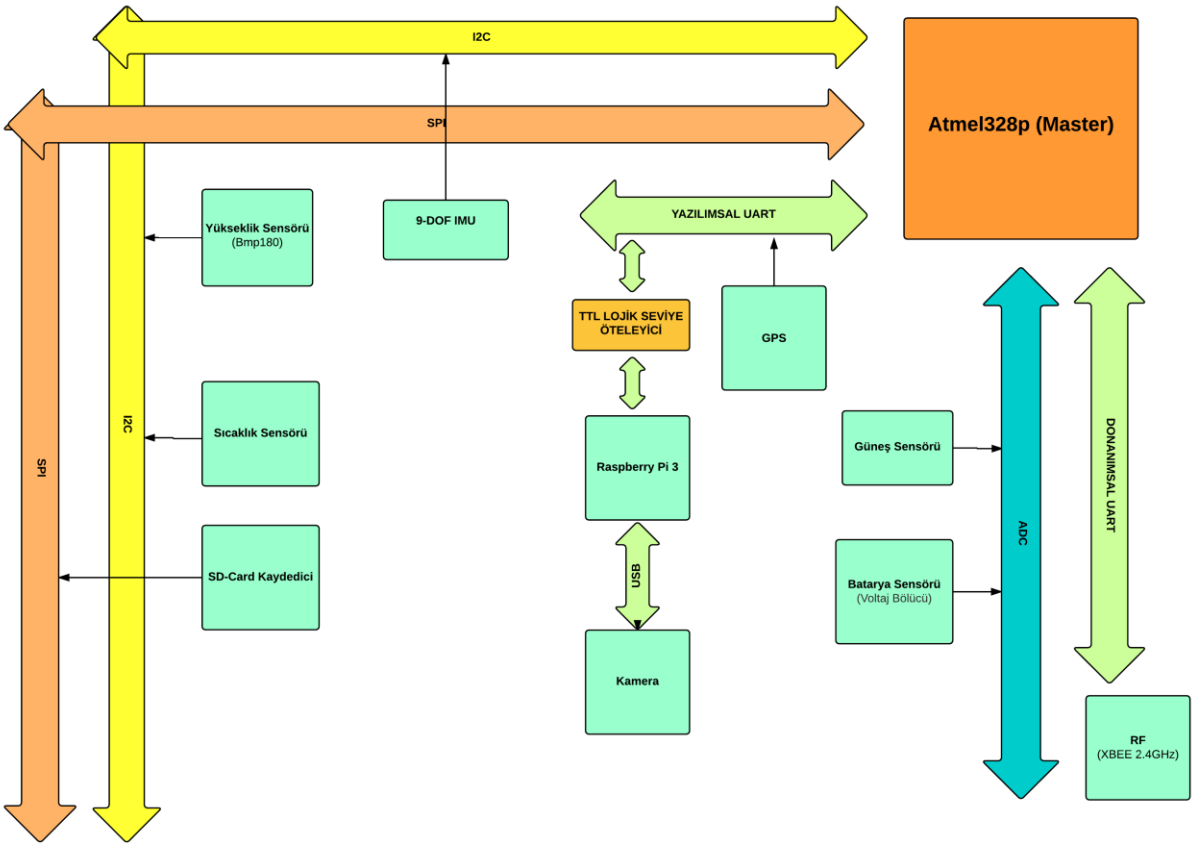
Sistemin tasarımı genel hatlarıyla belirlenip şematik şeklinde bağlantı yolları güç ve iletişim kanalları olarak ayrı ayrı belirlendi.

Tasarımda ücretsiz ve daha önce tecrübe edilmiş olması nedeniyle Diptrace PCB Design programı kullanıldı[22]. Şekil 5.1'de alt sistemler arası güç yolları gözükmektedir.



Şekil 5.1 Alt Sistemler Arası Güç Yolları

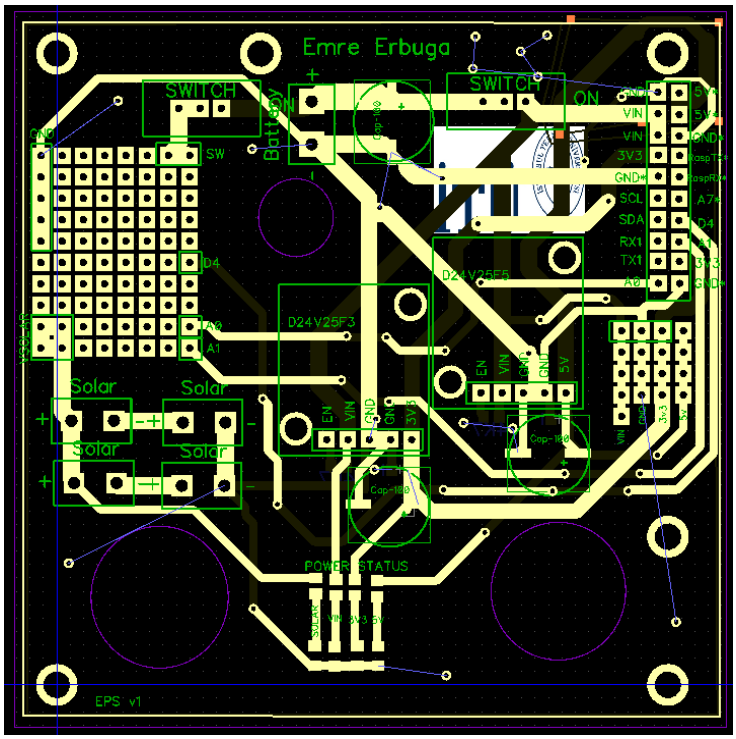
Daha sonra çipten çipe haberleşme yolları ve protokolleri belirlendi. Bunlar Şekil 5.2'de görülen genel bir şemada gösterildi. PCB tasarımlarında bu yollara dikkat edilerek yapıldı.



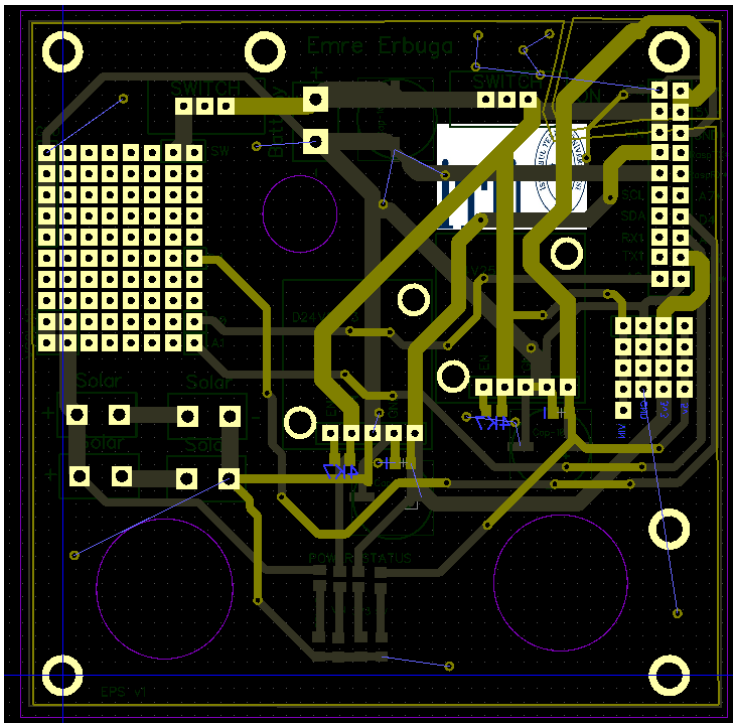
Şekil 5.2 Alt Sistemler Arası Haberleşme Protokolleri ve Yolları

5.1 Elektronik Güç Kartı PCB (EPS)

Şekil 5.3 ve Şekil 5.4'te güç katının Diptrace'te tasarım aşamasındaki görüntüleridir. Bu kartta voltaj regülasyon ve dağıtım, güneş paneli şarj devresi için boşluk, batarya giriş ünitesi mevcuttur. Bu katın en üstte olmasının nedeni ısıl bütçenin olabildiğince uydu yüzeyine yakın yapılmak istenmesindendir. Yüksek akım taşınan yollar olabilidiğince kalın yapılmıştır. Ayrıca dizayn'da kartın üst ve alt tabakası "ground pour" yapılarak taze bir ground kaynağı oluşturulmak istenmiştir.

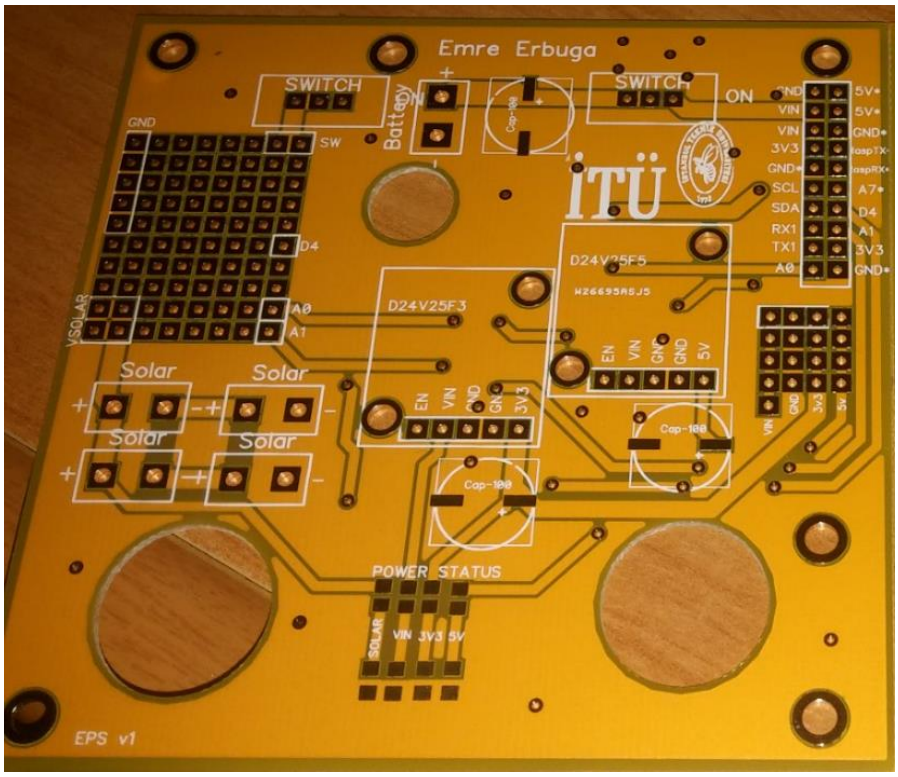


Şekil 5.3 Diptrace'te Tasarım Esnasında EPS Üstten Görünüm



Şekil 5.4 Diptrace'te Tasarım Esnasında EPS Alttan Görünüm

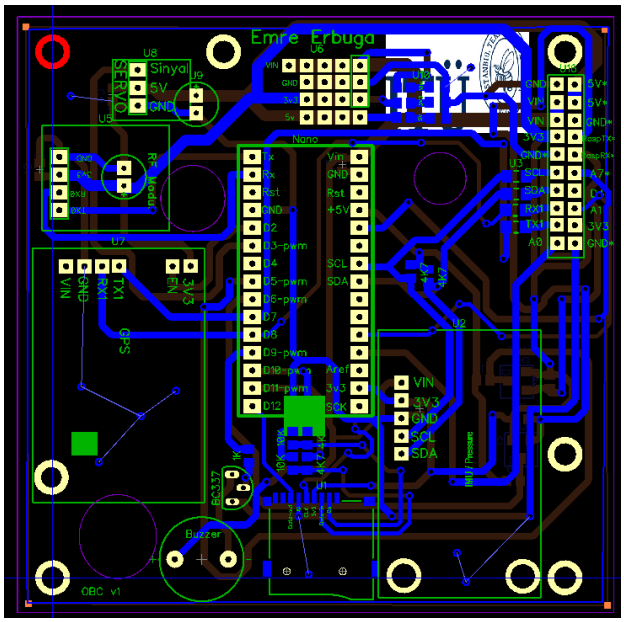
PCB'nin gerçeklenmiş hali de Şekil 5.5'te görülmektedir.



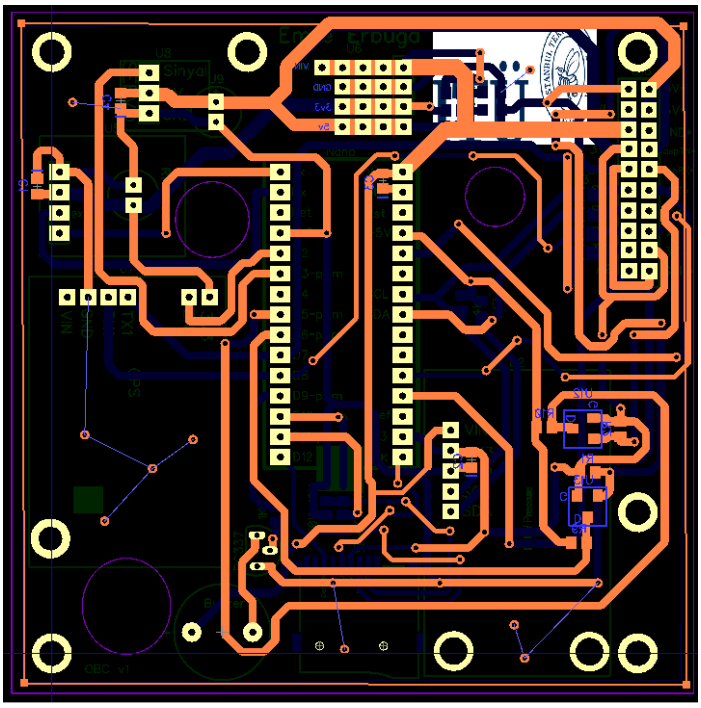
Şekil 5.5 EPS'in Gerçeklenmiş Hali

5.2 İşlemci Kartı PCB (OBC)

İşlemci kartında, mikrodenetleyici, sensörler, gps, lojik seviye dönüştürücü, haberleşme birimi girişi, sd kart soketi, buzzer ve sürücü devresi mevcuttur. Güç taşıyan yollar kalın çizilmeye çalışılmıştır. Şekil 5.6 ve Şekil 5.7'de güç katının Diptrace'te tasarım aşamasındaki görüntüleridir.

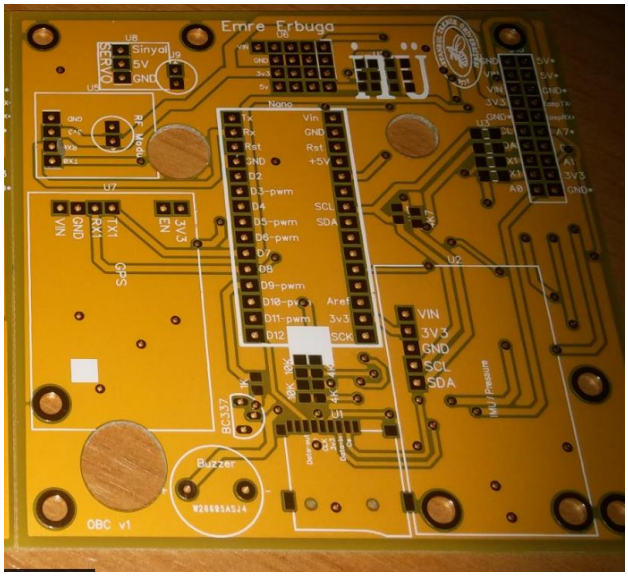


Şekil 5.6 Diptrace'te Tasarım Esnasında OBC Üstten Görünüm



Şekil 5.7 Diptrace'te Tasarım Esnasında OBC Alttan Görünüm

PCB'nin gerçeklenmiş hali de Şekil 5.8'te görülmektedir. Şekilde sağ üstte bulunan uyumlu soketler aracığılıyla dikeyde güç, haberleşme yolları iletilmektedir. Çok güç kullanan sistemlerin en üst katta voltaj regülasyonu yapıldıktan sonra alt katlara iletiminde olabildiğince çok paralel yol kullanılarak yol direnci düşürülüp oluşacak yol ısısı azaltılmaya çalışmıştır. Aynı zamanda yollardaki dönüşler yuvarlağa yakın bir şekilde yapılmasına özen gösterilmiştir(elektron hareketi-sınır koşul).



Şekil 5.8 OBC'in Gerçeklenmiş Hali

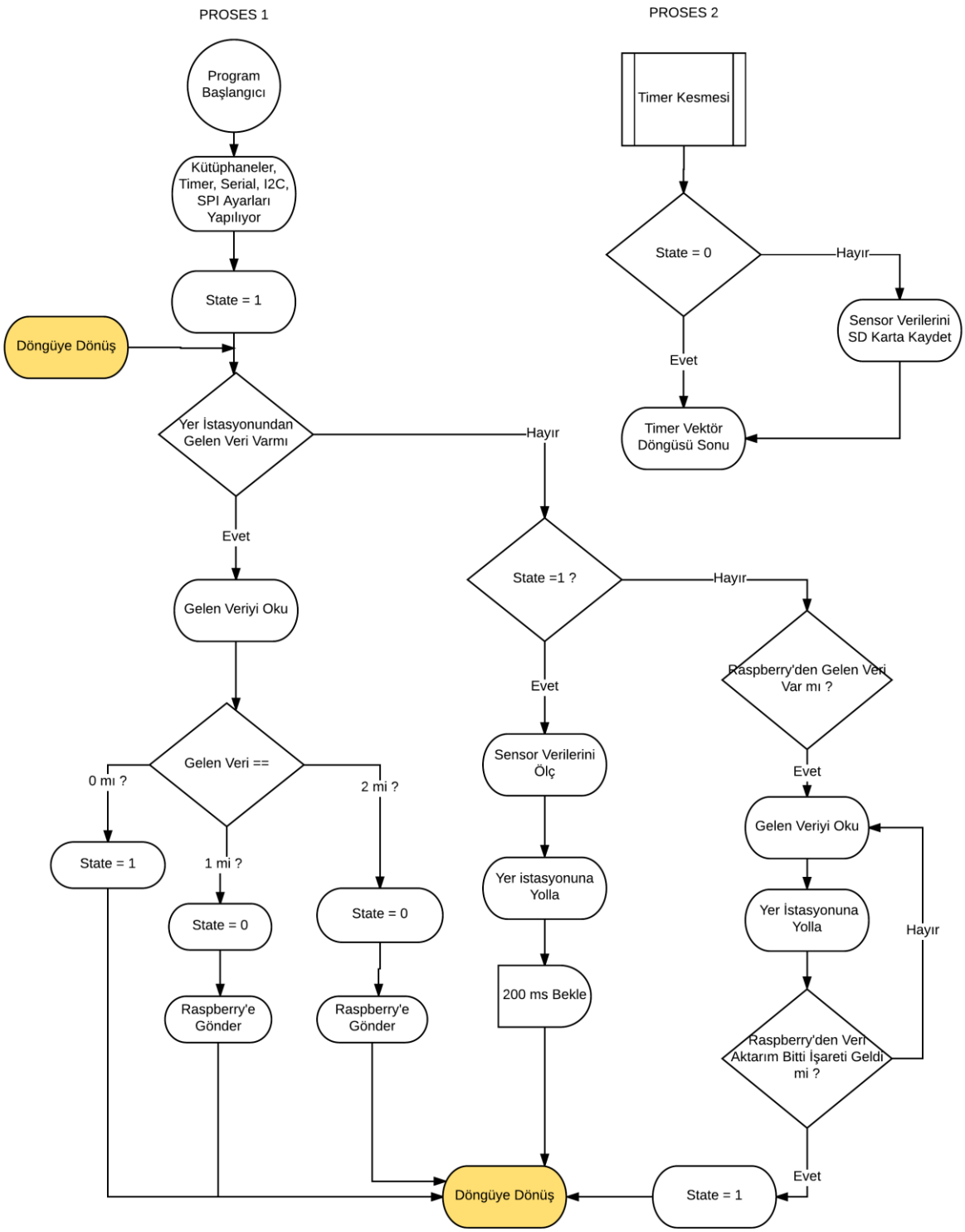
6. YAZILIMSAL TASARIM

Uyduda mikrodenetleyici, faydalı yükte farklı programlama dilleriyle yazılmış 2 ayrı kod çalışmaktadır. Birbirleriyle uyumludurlar. Ayrıca yer istasyonunda çalıştırılan kod da mevcuttur. Bu nedenle 3 ayrı başlıkta inceleceğiz.

6.1 Mikrodenetleyici tarafı

Mikrodenetleyicimizde çalışan C/C++ yazılmış uydunun temel görevlerini yerine getirmektir; sensörde üretilen verileri anlamlandırmak, haberleşme birimiyle olan veri alışverişini düzenlemek, sd karta verileri yazmaktır. Aynı zamanda yer istasyonundan gelen verilerin ilk değerlendirmesinin yapıldığı daha sonra bu anlamlı verilerin faydalı yüke gönderilip orada çalışan kodun kontrolünü yapmaktadır.

Şekil 6.1'de kodun akış diyagramı görülmektedir.



Şekil 6.1 Mikrodenetleyici Kod Akış Diyagramı

6.2 Faydalı yük

Raspberry Pi 3 ile kamera modülü yer almaktadır. Raspberry pi 3 ile yüksek işlem gücü gerektiren işlemler yaptırılıp elde edilen küçültülmüş veri mikrodenetleyiciye gönderilip oradan yer istasyonuna kablosuz olarak gönderilmektedir. Raspberry Pi 3 içinde Python ile yazılmış kod çalıştırılmaktadır.

Raspberry Pi aşağıdaki sıralamada ayarlar yapılmıştır. İlk olarak Şekil 6.2'deki kodlar terminal üzerinden raspberry pi'ye uygulanmıştır.

```
pi@raspberrypi:~ $ sudo apt-get update && sudo apt-get upgrade -y
```

Update the firmware:

```
pi@raspberrypi:~ $ sudo rpi-update
```

Reboot to apply the changes:

```
pi@raspberrypi:~ $ sudo reboot
```

Şekil 6.2 Raspberry Pi için yapılan güncellemeler ve yeniden başlatma

Daha sonra üstünde kurulu olan Python ile Opencv kullanmak için altta yer alan kod girilip Opencv kütüphanesi yüklenmiştir.

```
sudo apt-get install libopencv-dev python-opencv
```

UART için bazı ayarlar gerekmektedir. Raspberry Pi 3 üstünde bulunan, Şekil 6.3'te görünen, GPIO 15 ve GPIO 16 olan pinler konsol için seri haberleşmede kullanılmaktadır. Bunu kapatıp GPIO üstünden haberleşme için kullanmak için aşağıdaki ayarlar sırasıyla yapılmıştır.

Şekil 6.3 Raspberry Pi 3 UART Pinleri RX-TX

"/boot/cmdline.txt" dosyası sudo nano komutuyla aşağıdaki gibi kod satırı varsa silinir ve kaydedilir[23].

```
console=ttyAMA0,115200 kgdboc=ttyAMA0,115200
```

“/boot/config.txt” dosyası sudo nano komutuyla açılıp, aşağıdaki satır en sona eklenir ve kaydedilip çıkılır[24].

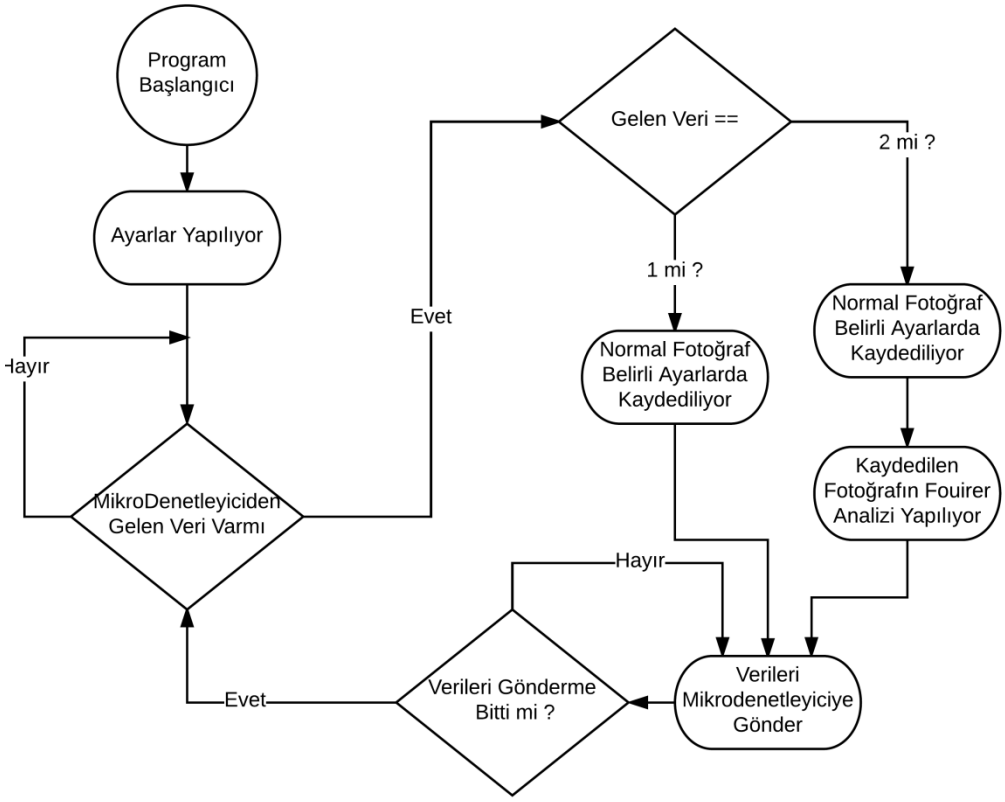
```
enable_uart=1
```

Uart için ayarlar bittikten sonra ayarları kontrol etmek için “gpio readall” komutu konsola yazılarak GPIO 15 ve 16’nın Şekil 6.4’teki gibi ALT0 modunda olduğunu görmemiz gerekmektedir.

```
| BCM | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | BCM |
+-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
|     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
|   2 |   8 |   SDA.1 | ALT0 | 1 |  3 || 4  |   |      | 5V      |     |     |
|   3 |   9 |   SCL.1 | ALT0 | 1 |  5 || 6  |   |      | 0v      |     |     |
|   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 1 | ALT0 | TxD     | 15  | 14  |
|     |     |      0v |      |   |  9 || 10 | 1 | ALT0 | RxD     | 16  | 15  |
|  17 |   0 | GPIO. 0 |   IN | 0 | 11 || 12 | 0 | IN   | GPIO. 1 | 1   | 18  |
|  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
|  22 |   3 | GPIO. 3 |  OUT | 0 | 15 || 16 | 0 | IN   | GPIO. 4 | 4   | 23  |
|     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 | 5   | 24  |
|  10 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |     |
|   9 |  13 |    MISO |   IN | 0 | 21 || 22 | 0 | IN   | GPIO. 6 | 6   | 25  |
|  11 |  14 |    SCLK |   IN | 0 | 23 || 24 | 1 | IN   | CE0     | 10  | 8   |
```

Şekil 6.4 ALT0 Modunda TxD ve RxD

Mikrodenetleyiciden gelen komutların algılanması ve bu algılanan senaryolara göre fotoğraf çekilip bu fotoğraf belirlenen piksel boyutlarında kesilip gri-skalaya dönüştürülür(aktarım hızını düşürmek için). Daha sonra genişlik, yükseklik ve 8 bit gri renk genliği olan bir matris oluşur. Bu elemanlar sırayla byte byte mikrodenetleyiciye gönderilir, bunu algılayan mikrodenetleyici kablosuz haberleşme modülüne gönderip oradan yer istasyonuna iletilir. Şekil 6.5’te kodun akış diyagramı verilmiştir.



Şekil 6.5 Raspberry Pi 3 Çalışan Kodun Akış Diyagramı

Raspberry Pi 3'e yazılan bu kodun güç verildiğinde otomatik çalıştırılması düşünülmüştür. Bu nedenle aşağıdaki adımlar izlenerek sisteme güç verildiğinde Raspberry Pi 3 içindeki "payload.py" scripti otomatik olarak çalıştırılmaya başlar.

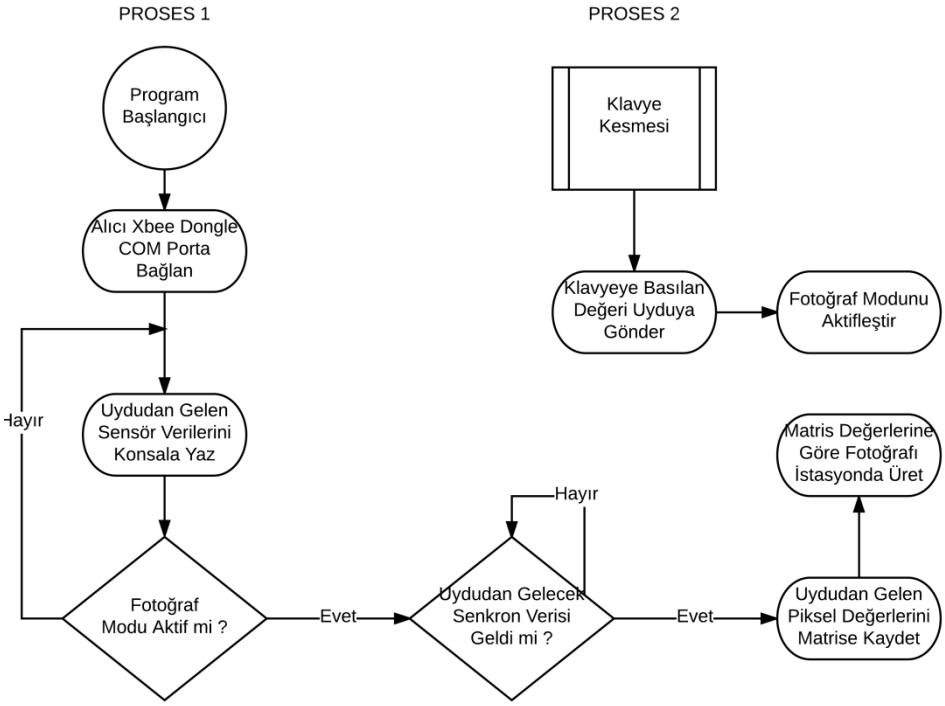
Terminal ekranında "sudo nano etc/profile" girilir. Daha sonra açılan pencerenin en altına "sudo python /home/pi/payload.py" girilmiştir[25]. Bu dosya uzantısı ve script ismi çalıştırılmak istenen scriptin bulunduğu yere göre değişmektedir.

Daha sonra "Ctrl + X" ile çıkılır ve kaydedilir. Reboot ettiğimizde artık Raspberry Pi 3 içindeki ana algoritmamız otomatik olarak çağıralacaktır.

6.3 Yer istasyonu

Kablosuz haberleşme birimi tarafından alınan veri COM port üzerinden okunmaktadır. Yazılım dili olarak Python kullanılmıştır. Gelen veriler terminal ekranına yazdırılmaktadır. Aynı zamanda klavyeye basıldığında kesmeye girip bu veriyi bilgisayara bağlı XBee modülü ile uyduya göndermektedir.

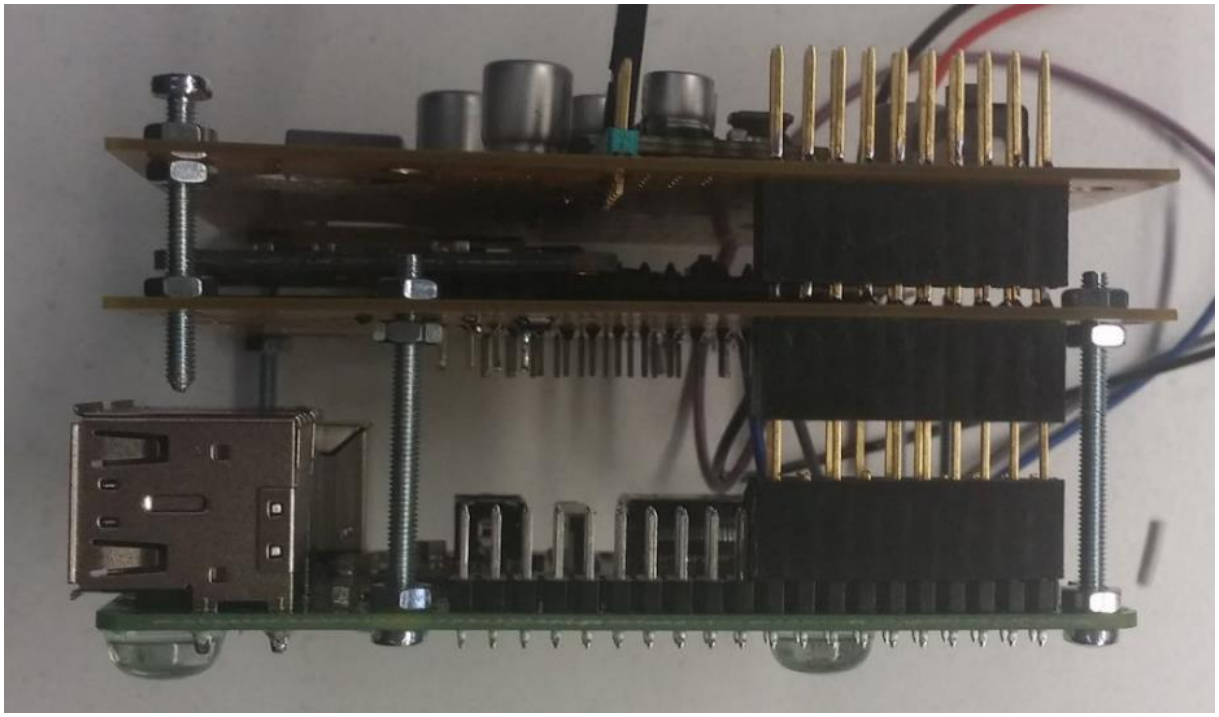
Şekil 6.6'da programın akış diyagramı gözükmektedir. Eğer fotoğraf çekme komutu gönderildiyse uydudan bu fotoğrafın piksel değerleri beklenir. Gelen piksel değerleri hafızaya alınıp tüm veriler geldikten sonra fotoğraf yer istasyonunda oluşturulup bilgisayar üzerindeki klasöre kaydedilmektedir.



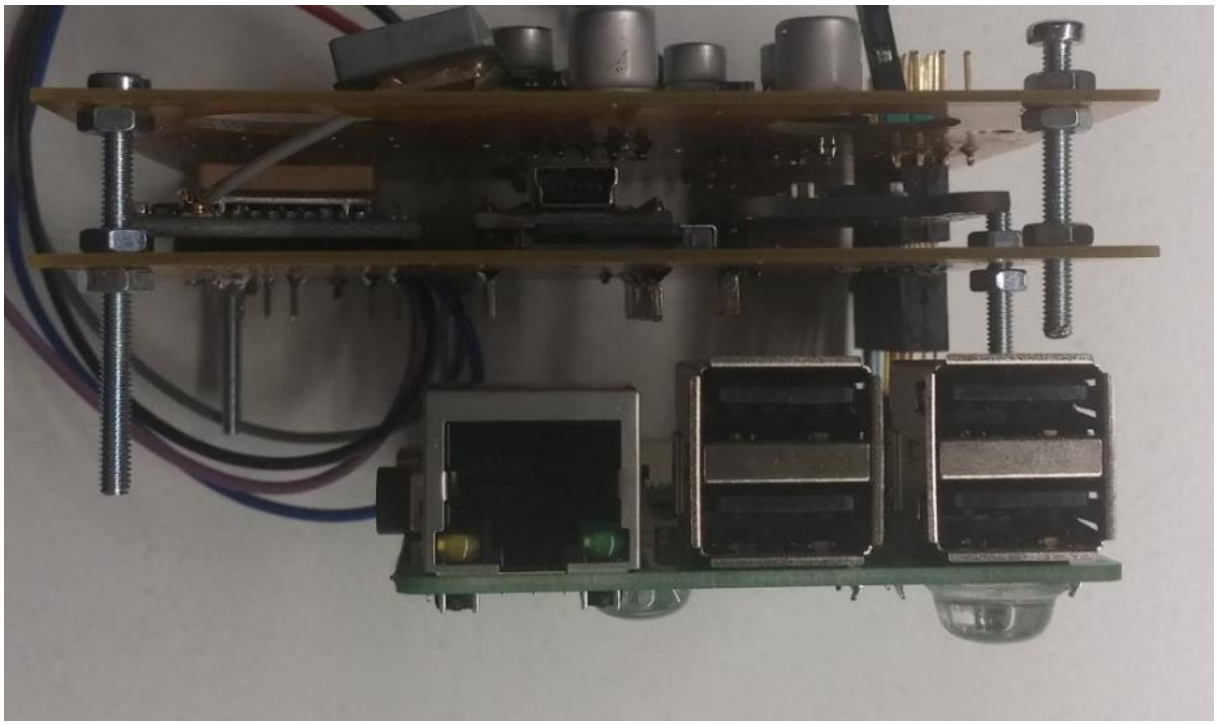
Şekil 6.6 Yer İstasyonu Akış Diyagramı

7. GERÇEKLENEN KÜPSAT

Gerçeklenen küp uydunun obc, eps ve raspberry pi'ın birbirine soket aracılığıyla bağlanmış olarak Şekil 7.1 ve 7.2'de gözükmektedir.



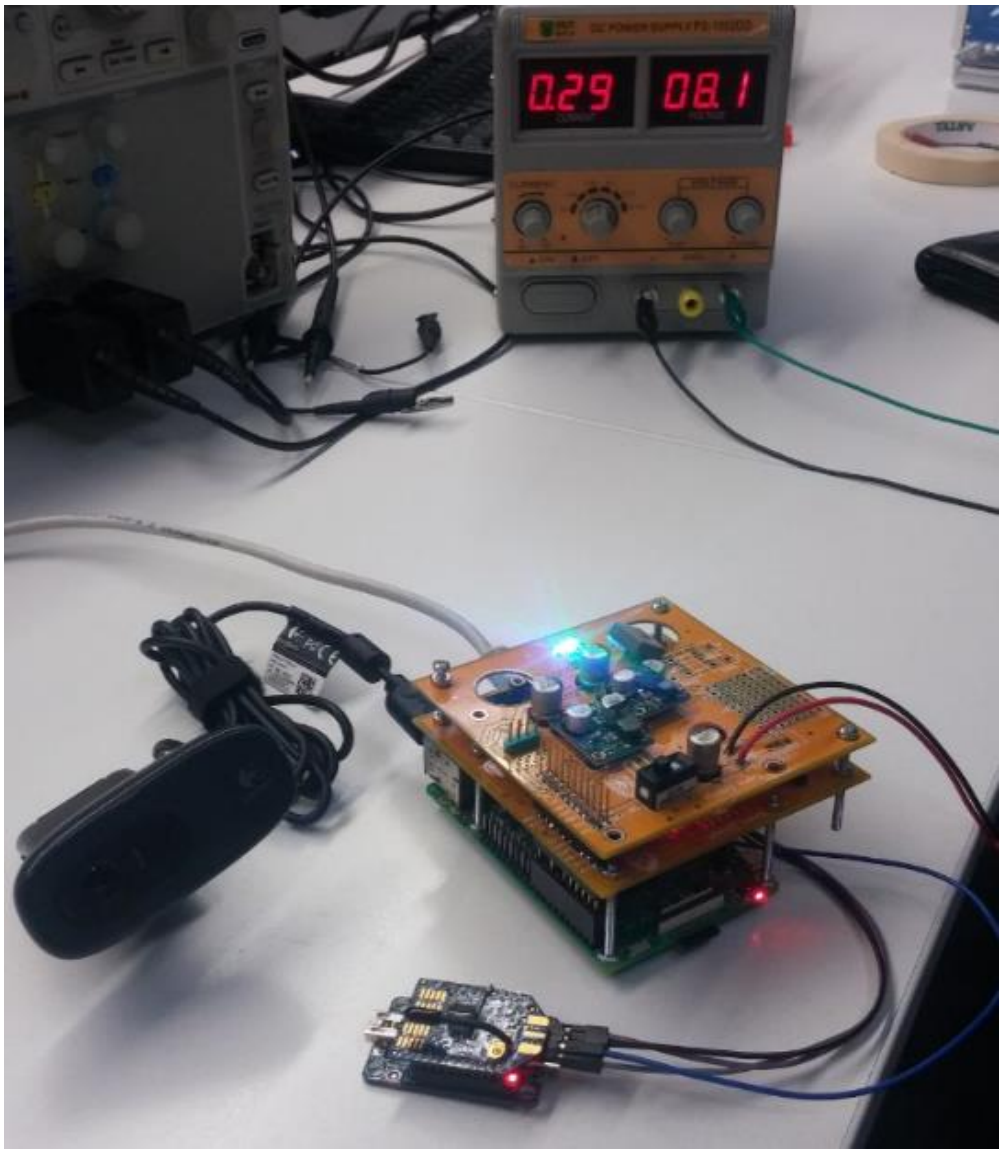
Şekil 7.1 OBC, EPS ve Raspberry Pi birbirlerine bağlı



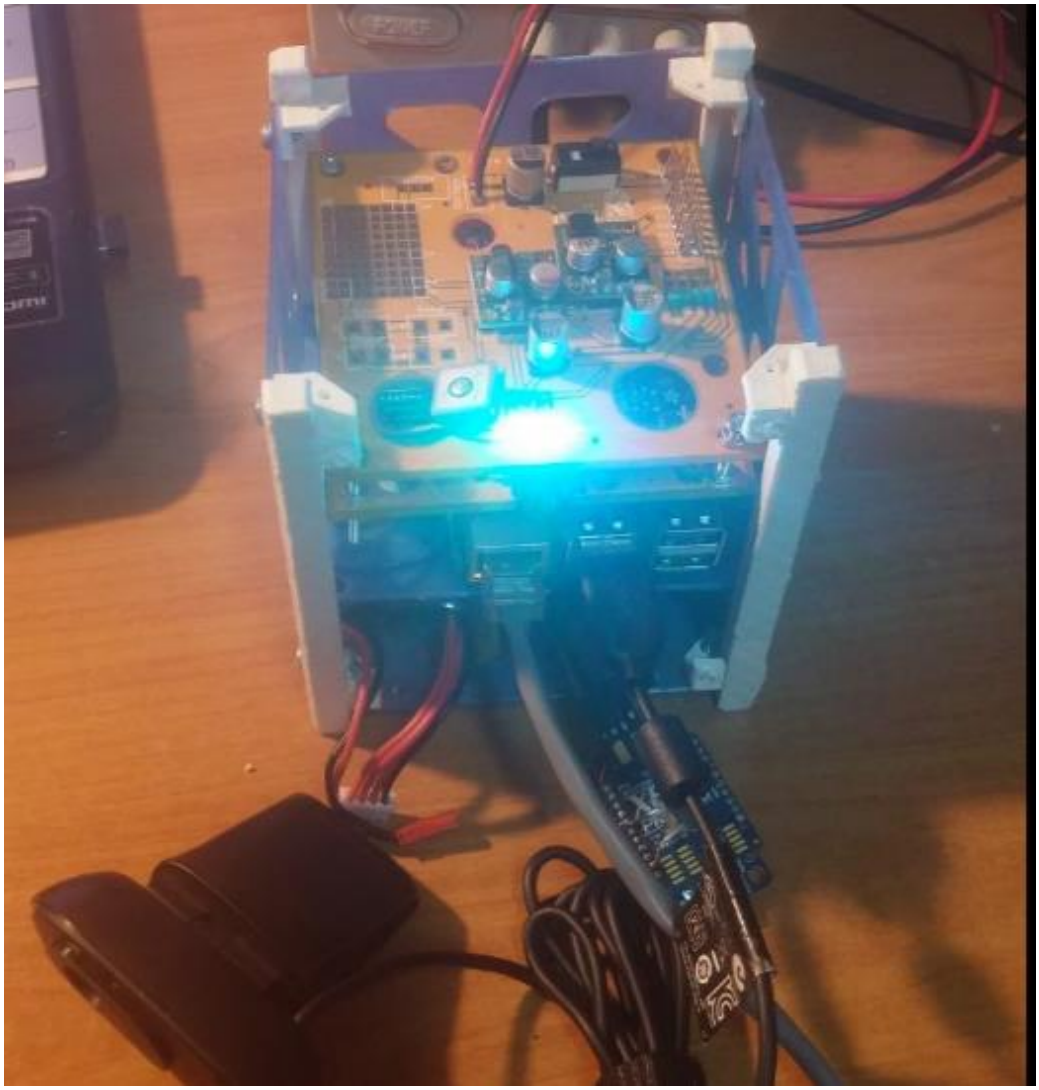
Şekil 7.2 OBC, EPS ve Raspberry Pi diğer açıdan

7.1 Sistemin Test Çıktıları

Sistemin fotoğraf çekmiyor iken yaklaşık 8V besleme ile 300mA güç tüketimine sahiptir. Şekil 7.3'te görülmektedir. Şekil 7.4'te ise sistemin 3 boyutlu mekanik kutusu içindeki görünümü gözükmektedir.



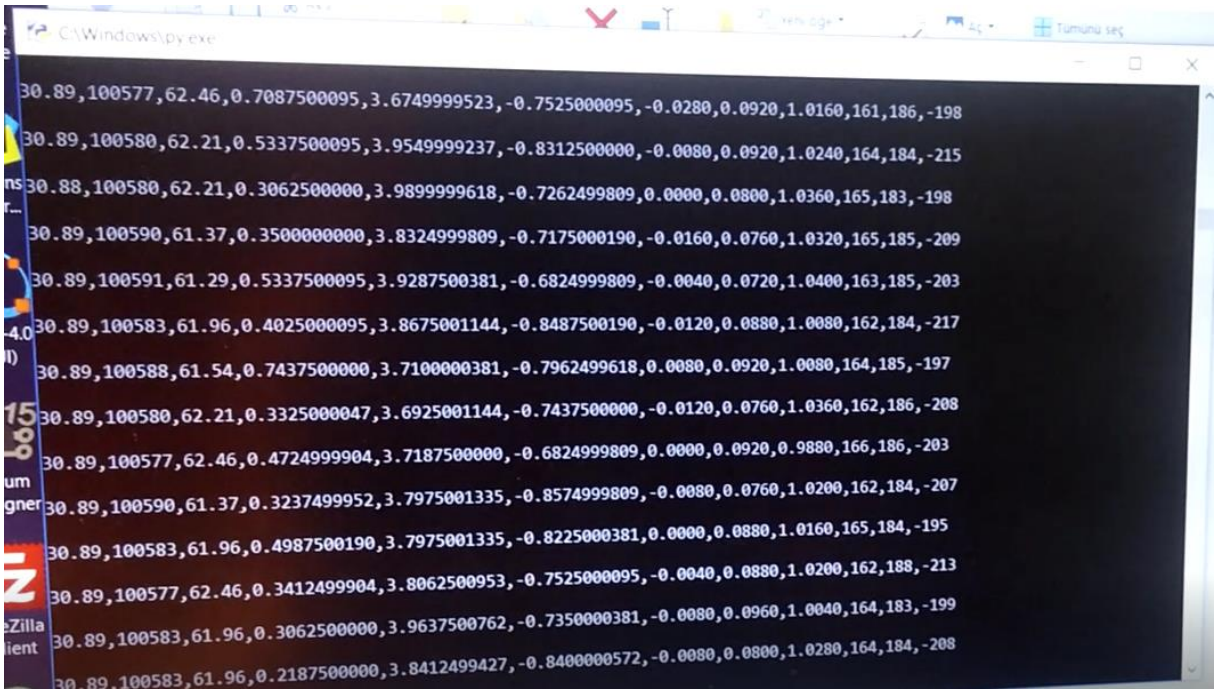
Şekil 7.3 Uydunun Tam Kapasite Çalışırken



Şekil 7.4 Uydu Tam Kapasite Çalışırken, 3 Boyutlu Baskılı Mekanik Kabında

Uydu normal modda iken yer istasyonuna sensör verilerini kablosuz olarak göndermesi durumunda yer istasyonundaki terminalden görünen veriler Şekil 7.5'te gözükmektedir. Veriler aşağıdaki gibi sıralanmıştır.

Sıcaklık, Basınç, Referansa Göre Yükseklik, X Jireskop, Y Jireskop, Z Jireskop, X İvme, Y İvme, Z İvme, X Manyetik Alan, Y Manyetik Alan, Z Manyetik Alan



Şekil 7.5 Yer İstasyonunda, Uydudan Gelen Verileri

Raspberry Pi üzerinde yüksek işlem gücü için senaryolar tek tek denenmiştir. Bunun için "1" verisi yer istasyonundan gönderilip, yer istasyonuna gelen veriler ile fotoğraf tekrar oluşturulmuştur. Şekil 7.6'te görülmektedir. Yer istasyonunda üretilen resmin boyutu yaklaşık 3.39 KB'tır.

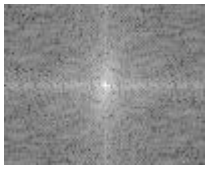
$50 * 40 * 1\text{Byte}(\text{Gri skala resim genliği}) = 2\text{KB}$ verinin kablosuz aktarımı ve yer istasyonunda tekrar resmin üretilmesi yaklaşık 15 saniye sürmektedir.

$100 * 80 * 1\text{Byte}(\text{Gri skala resim genliği}) = 8\text{KB}$ verinin kablosuz aktarımı ve yer istasyonunda tekrar resmin üretilmesi yaklaşık 40-42 saniye sürmektedir.

Şekil 7.6 Uydudan Gönderilip Yer İstasyonunda Üretilen Fotoğraf, (100,80) Piksel

Aynı fotoğrafın fouirer analizi yapılmış verileri sadece gönderilmiştir. Bu veriler ile yer istasyonunda üretilen fotoğraf Şekil 7.7'te görülmektedir. Bu fouirer analizi yapılmış fotoğraftan tekrar resmi elde etmek de pek tabii mümkündür. Böyle bir senaryo eklenmesinin nedeni: uydunun aktarım sırasında daha az veri aktarımı ile

aynı amaca ulaşmaktır. Ayrıca yüksek işlem gücü gerektiren işlemlerin faydalı yük üzerinde yapılabildiğini, yapılmasının uygun olduğunu göstermektedir. Yer istasyonunda üretilen resmin boyutu yaklaşık 4.46 KB'tır.



Şekil 7.7 Fotoğrafın Fouirer Analizi Yapılıp Yerde Üretilen Fotoğraf (100,80)

Şekil 7.8 Uyduda Çekilen Fotoğrafın Aslı (1000,800), Yaklaşık 143 KB

Sistem fotoğraf çekme sırasında 25-30 saniye boyunca 8V beslemeden 400-450mA güç tüketimine çıkmaktadır.

Ayrıca tüm sistemin(uydu + faydalı yük) ilk çalıştırma sırasında güç tüketimi 3-4 saniye için 8V, 600mA civarına çıkmaktadır.

7.2 Sistemin Ağırlık Bütçesi

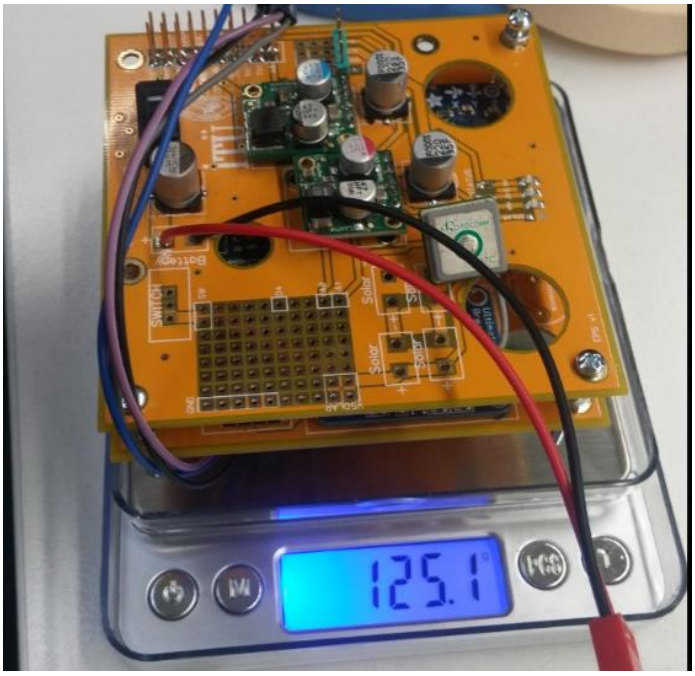
Sistemin var olan kısmının ağırlıkları kısmi olarak hassas tartı ile tartılmıştır. Alt başlıklarda şekillerde görülen değerlerde yer almaktadır.

$$Toplam\ Ağırlık\ Tahmini(gr) = 125.1 + 10.2 + 95.1 + 93.2 + 81.1 + G_{diğer}$$
$$= 404.7 + G_{diğer} = 754.7\ gr$$

$$G_{diğer} \cong 350\ gr\ (varsayım)$$

7.2.1 Ana Sistem

Ağırlık: 125.1 gram



Şekil 7.9 Ana Sistemin Ağırlığı

7.2.2 Haberleşme Modülü

Ağırlık: 10.2 gram

Şekil 7.10 Haberleşme Modülü Ağırlığı

7.2.3 Kamera

Ağırlık: 95.1 gram

Şekil 7.11 Kamera Ağırlığı

7.2.4 Batarya

Ağırlık: 93.2 gram

Şekil 7.12 Batarya Ağırlığı

7.2.5 Dış İskelet (3D Baskı)

Ağırlık: 81.1 gram

Şekil 7.13 Dış İskelet Ağırlığı

7.2.6 Diğer

Uyduya güneş panelleri, şarj devresinin parçaları, termal ve radyasyon koruma kılıfları vb gibi ekstra parçaların eklenmesi söz konusudur. Ayrıca yörüngeye çıkacak uyduda haberleşme modülü değiştirilip, yeni anten ve modül ağırlıkları göz önüne alınmalıdır. Bu noktada toplam ağırlığa yaklaşık bu diğer kısım için bir değer vermemiz gerekirse 300-350 gram arası tahmininde bulunabiliriz. Bu tahmini uydumuza uygun eksik parçaların araştırılıp kesin ağırlıklarının öğrenilmesiyle güncelleyebiliriz.

8. SİSTEMİN İYİLEŞTİRMESİNE YÖNELİK ÖNERİLER

Mikrodenetleyici ile Raspberry Pi 3 arasındaki yazılımsal Uart'ın hızı 9600 baud/saniye'den 115200 baud/saniye'e yükseltilebilir. Bu sayede faydalı yükte çekilen fotoğrafın mikrodenetleyiciye aktarımı ve dolayısıyla oradan yer istasyonuna aktarımı yaklaşık 12 kat hızlanacaktır.

Sistemin haberleşme modülleri test için 2.4 GHz'de çalışan Xbee modülleri seçilmiştir. Uydu alçak dünya yörüngesinde (LEO) haberleşme için 410-420 MHz bandını kullanabilir. Bu band için ITU-R vb. kurumlardan izin, kiralama yapılması gerekmektedir.

Uyduda kullanılan GPS alıcısı yüksek hız ve yükseklikte çalışmamaktadır. LEO uyduları için özel GPS alıcı kullanılmalıdır.

Raspberry Pi'ın usb girişine takılan kameramız yerine daha kompakt ve hafif bir kamera kullanılıp ağırlık azaltılabilinir.

Termal bütçe hesaplanmamıştır. Bunun için gerekli çalışma koşulları oluşturulup detaylı analizler yapılmalıdır.

Kullanılan elektronik parçalar; direnç, kapasitör(özellikle elektrolitik) vb. parçaların sıcaklık toleransları araştırılmalıdır.

Yazılımsal oluşabilecek hatalar için test yazılımı oluşturulup denemeler otomatik bir şekilde yapılabilir.

Sisteme güneş paneli eklenmesi için uygun alan bırakılmıştır. Bu alana yeterli güneş panelinin sığdırılması ve şarj devresinin EPS üzerinde bırakılan boş alanda gerçeklemesi yapılmalıdır.

Titreşim ve şok testi yapılmalıdır. Uyduyu uzaya çıkaracak roketin atmosferden çıkarken roket motorunun ve atmosferdeki aerodinamik etkiler nedeniyle oluşacak etkilerin simülasyonu yapılmalıdır.

Termal vakum ortamında tüm uydu sistmenin analizi yapılıp, hatalı olduğu düşünülen bölümlerin giderilebiliniyorsa aynı tasarım üzerinden giderilemiyorsa eski tasarım ve hatalı kısmı giderecek şekilde yeni tasarım yapılmalıdır.

9. SONUÇ

Tasarlanan uydu ile uzay sistemleri hakkında genel tasarımsal bilgi edinilmiştir. Aynı zamanda hem donanım hem de yazılım tasarımı farklı tasarım programları ve programlama dillerinde yapılmıştır. Elde edilen deneyim, veriler ve sistemin gelişimine yönelik öneriler ışığında gelecekte daha gelişmiş uydu ve uzay projeleri için alt yapı oluşturulması sağlanmıştır.

KAYNAKLAR

[1] Interorbital.com. (n.d.). *NEPTUNE Launch Services*. [online] Available at: http://www.interorbital.com/interorbital_06222015_003.htm [Accessed 30 Dec. 2016].

[2] Unoosa.org. (2016). *KiboCUBE: UN / Japan Cooperation Programme*. [online] Available at: http://www.unoosa.org/oosa/en/ourwork/psa/hsti/kibocube.html [Accessed 30 Dec. 2016].

[3] Heidt, H., Puig-Suari, J., Moore, A., Nakasuka, S. and Twiggs, R. (2000). *CubeSat: A new Generation of Picosatellite for Education and Industry Low-Cost Space Experimentation*. 1st ed. [ebook] p.18. Available at: http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=2069&context=smallsat [Accessed 31 Dec. 2016].

[4] TAI. (2016). *GÖKTÜRK-2*. [online] Available at: https://www.tai.com.tr/tr/proje/gokturk-2 [Accessed 31 Dec. 2016].

[5] P. B. Hederson, "High Data Rate Radio Transmitter for Cube, Satellites", pp.38, 2009

[6] NASA. (2016). *NASA's CubeSat Launch Initiative*. [online] Available at: https://www.nasa.gov/directorates/heo/home/CubeSats_initiative [Accessed 31 Dec.2016].

[7] Gomspace.com. (2016). *Cubesat and nano-satellite solutions*. [online] Available at: http://gomspace.com/index.php?p=products [Accessed 31 Dec. 2016].

[8] Docs.opencv.org. (2016). *OpenCV: OpenCV modules*. [online] Available at: http://docs.opencv.org/3.2.0/ [Accessed 31 Dec. 2016].

[9] Atmega328p. (n.d.). 1st ed. [ebook] Atmel, p.2. Available at: http://www.atmel.com/Images/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf [Accessed 31 Dec. 2016]. S

[10] Xbee. (n.d.). 1st ed. [ebook] Digi, p.2. Available at: https://cdn.sparkfun.com/datasheets/Wireless/Zigbee/ds_xbeezbmodules.pdf [Accessed 31 Dec. 2016].

[11] Sparkfun.com. (n.d.). *SparkFun XBee Explorer Dongle - WRL-11697 - SparkFun Electronics*. [online] Available at: https://www.sparkfun.com/products/11697 [Accessed 31 Dec. 2016].

[12] [Online]. Available: https://www.adafruit.com/product/1604. [Accessed 19 10 2016].

[13] GLOBAL POSITIONING SYSTEM STANDARD POSITIONING SERVICE SIGNAL SPECIFICATION. (1995). 2nd ed. [ebook] p.18. Available at: http://www.gps.gov/technical/ps/1995-SPS-signal-specification.pdf [Accessed 31 Dec. 2016].

[14] [Online]. Available: http://www.adafruit.com/products/746. [Accessed 15 09 2016].

[15] [Online]. Available: https://www.sparkfun.com/products/12009. [Accessed 19 09 2016].

[16] BSS138. (2005). 1st ed. [ebook] Fairchild Semiconductor, p.2. Available at: https://cdn.sparkfun.com/datasheets/BreakoutBoards/BSS138.pdf [Accessed 31 Dec. 2016].

[17] Lynch, B. and Wallace, C. (2014). CubeSat electronic power system. p.7.

[18] Switch-Mode Power Supply. (2014). 1st ed. [ebook] On Semiconductor, p.6. Available at: http://www.onsemi.com/pub_link/Collateral/SMPSRM-D.PDF [Accessed 31 Dec. 2016].

[19] [Online]. Available: https://www.pololu.com/product/2849. [Accessed 11 09 2016].

[20] [Online]. Available: https://www.pololu.com/product/2851. [Accessed 11 09 2016].

[22] [Online]. Available: http://diptrace.com/tr/. [Accessed 12 10 2016].

[23] Oscar Liang. (n.d.). *Raspberry Pi and Arduino Connected Over Serial GPIO - Oscar Liang*. [online] Available at: https://oscarliang.com/raspberry-pi-and-arduino-connected-serial-gpio/ [Accessed 31 Dec. 2016].

[24] Raspberrypi.org. (n.d.). *Raspberry Pi • View topic - UART sometimes working, arbitrary MODE change to IN/OUT*. [online] Available at: https://www.raspberrypi.org/forums/viewtopic.php?f=28&t=151430 [Accessed 31 Dec. 2016].

[25] Opentechguides.com. (n.d.). *Auto Login and Auto Start in Raspberry Pi*. [online] Available at: http://www.opentechguides.com/how-to/article/raspberry-pi/5/raspberry-pi-auto-start.html [Accessed 31 Dec. 2016].

[26] The Electronic System Design, Analysis, Integration, and Construction of the Cal Poly State University CP1 CubeSat. (n.d.). 1st ed. [ebook] Available at: http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1928&context=smallsat [Accessed 15 Oct. 2016].

[27] JEM Payload Accommodation Handbook. (2013). 1st ed. [ebook] JAXA. Available at: http://aerospacebiz.jaxa.jp/wp-content/uploads/2016/07/jem_handbook_eng.pdf [Accessed 3 Oct. 2016].

EKLER

EK-A : Küp Uyduda Mikrodenetleyicide Çalışan Kod (C/C++)

EK-B : Küp Uyduda Raspberry Pi 3'te Çalışan Kod (Python)

EK-C : Yer İstasyonunda Çalışan Kod (Python)

EK-A

```
// KUTUPHANELER cagiriliyor
#include <Wire.h>
#include <BMP085.h>
#include <I2Cdev.h>
#include <L3GD20H.h>
#include <LSM303DLHC.h>
#include <SoftwareSerial.h>
#include <SPI.h>
#include <SD.h>

SoftwareSerial mySerial(6, 5); // RX, TX
const int chipSelect = 10;

//objeler olusturuluyor arka plandaki kutuphanede kullanılmak uzere
LSM303DLHC accelMag;
BMP085 bmp;
L3GD20H gyro;

int16_t ax, ay, az;
int16_t mx, my, mz;

float absoluteAltitude ;
long realPressure;
double realTemperature;




int16_t avx, avy, avz;
double referencePressure;

void setup()
{

  // I2C haberlesme baslatiliyor
  Wire.begin();

  // Xbee haberlesme hizi 115200 baud/saniye ayarlandi
  Serial.begin(115200);
  // Mikrodenetleyici ile raspberry arasi haberlesme hizi 9600
baud/saniye ayarlandi
  mySerial.begin(9600);

  //SD kart SPI iletisimi baslatiliyor
  if (!SD.begin(10))
  {
    Serial.println("SD kart hatali!");
    return;
  }

  accelMag.initialize();


  accelMag.setAccelFullScale(2);

  // accelerometer -  data aktarimi 220Hz ayarlandi
  accelMag.setAccelOutputDataRate(200);

  // magnetometer - data aktarimi 220Hz ayarlandi
```

```
  accelMag.setMagOutputDataRate(220);


  accelMag.setMagGain(1100);

  // surekli magnetometer calisir durumda
  accelMag.setMagMode(LSM303DLHC_MD_CONTINUOUS);
  Serial.println(accelMag.getMagMode());



  //basinc sensoru baslatiliyor
  while (!bmp.begin(BMP085_ULTRA_LOW_POWER))
  {
    delay(500);
  }

  // referans basinc degiskene ataniyor
  referencePressure = bmp.readPressure();
  gyro.initialize();

  // gyrometer - data aktarimi 250Hz ayarlandi
  gyro.setFullScale(250);

  checkSettings();


  // SD kart icin kullanilan Timer ayarlamasi, yaklasik 1 saniye
olarak ayarlaniyor
  cli();
  TCCR1A = 0;
  TCCR1B = 0;
  TCNT1  = 0;
  OCR1A = 15624;
  TCCR1B |= (1 << WGM12);
  TCCR1B |= (1 << CS12) | (1 << CS10);
  TIMSK1 |= (1 << OCIE1A);
  sei();

}

void checkSettings()
{
  long bmpVersion = bmp.getVersion();
  Serial.print("Chip version: ");
  Serial.print(bmpVersion >> 8); Serial.print(".");
Serial.print(bmpVersion & 0xFF);
  Serial.print(" (0x"); Serial.print(bmpVersion, HEX);
Serial.println(")");

  Serial.print("Oversampling: ");
  Serial.println(bmp.getOversampling());
  Serial.print("Software Oversampling: ");
  Serial.println(bmp.getSoftwareOversampling());
}


//Sensor verileri okunup degiskenlere kaydediliyor
void imuHesapla()
{
```

```
  // sicaklik olculuyor
  realTemperature = bmp.readTemperature();
  // basinc olculuyor
  realPressure = bmp.readPressure();

  // Yukseklik hesaplaniyor
  absoluteAltitude = bmp.getAltitude(realPressure);
  //jiroskop verileri olculuyor
  gyro.getAngularVelocity(&avx, &avy, &avz);
  // ivme verileri olculuyor
  accelMag.getAcceleration(&ax, &ay, &az);

  accelMag.getMag(&mx, &my, &mz);

}
// Xbee veya Konsol sensor verilerini gosterme fonksiyonu
void  serialSensorGoster()
{
  Serial.print(realTemperature);
  Serial.print(",");
  Serial.print(realPressure);
  Serial.print(",");
  Serial.print(absoluteAltitude);
  Serial.print(",");
  Serial.print(avx * 0.00875F, DEC); Serial.print(",");
  Serial.print(avy * 0.00875F, DEC); Serial.print(",");
  Serial.print(avz * 0.00875F, DEC); Serial.print(",");
  Serial.print(ax * 0.0000625F, 4); Serial.print(",");
  Serial.print(ay * 0.0000625F, 4); Serial.print(",");
  Serial.print(az * 0.0000625F, 4);  Serial.print(",");
  Serial.print(mx); Serial.print(",");
  Serial.print(my); Serial.print(",");
  Serial.println(mz);

}
//SD karta kaydetme fonksiyonu
void  sdKaydet()
{

  File dataFile = SD.open("uydu.txt", FILE_WRITE);

  if (dataFile)
  {
    dataFile.print(realTemperature);
    dataFile.print(",");
    dataFile.print(realPressure);
    dataFile.print(",");
    dataFile.print(absoluteAltitude);
    dataFile.print(",");
    dataFile.print(avx * 0.00875F, DEC); dataFile.print(",");
    dataFile.print(avy * 0.00875F, DEC); dataFile.print(",");
    dataFile.print(avz * 0.00875F, DEC); dataFile.print(",");
    dataFile.print(ax * 0.0000625F, 4); dataFile.print(",");
    dataFile.print(ay * 0.0000625F, 4); dataFile.print(",");
    dataFile.print(az * 0.0000625F, 4);  dataFile.print(",");
    dataFile.print(mx); dataFile.print(",");
    dataFile.print(my); dataFile.print(",");
    dataFile.println(mz);
    dataFile.close();
  }
```

```
}
int gsVeri ;
byte gelenVeri;
bool state = 1;

//Timer vektoru geldiginde yapilacaklarin oldugu fonksiyon
ISR(TIMER1_COMPA_vect)
{
  // Yer istasyonuna resim aktarilirken girmesi engelleniyor,
bitlerin aktarim senkronu bozuluyor.
  if (state) {
    //SD karta kaydediliyor
    sdKaydet();
  }
}
void loop()
{
  if (Serial.available())
  { //Yer istasyonundan gelen veri okunup degiskene ataniyor
    gsVeri = Serial.read();
    //Sensor verilerini gosterme durumu, "0" gonderilirse yer
istasyonundan ascii karsiligi 48 oldugundan bu sekilde sorgulama
yapiliyor
    if (gsVeri == 48)
    {
      state = 1;
    }
    //Raspi normal foto durumu, "1" geldi mi diye bakiliyor
    else if (gsVeri == 49)
    {
      state = 0;
      // Raspberryde calisan yazilima 1 komutu yollaniyor
      mySerial.write("1");

    }
    //Raspi fouirer durumu, "2" geldi mi diye bakiliyor
    else if (gsVeri == 50)
    {
      state = 0;
      // Raspberryde calisan yazilima 2 komutu yollaniyor
      mySerial.write("2");

    }

    else
    {
      state = 0;
    }
  }
  // Xbee ile sensor verilerini yer istasyonuna gonderme normal
durumda
  if (state)
  {
    imuHesapla();
    serialSensorGoster();
    //  200ms bekleme yapiyoruz
    delay(200);
  }
  //diger durumda resim aktarimi yapilirken sensor verileri
gonderilmiyor
  else {
```

```
    if (mySerial.available())
    {
      // okunan veriler, xbee uzerinden yer istasyonuna aktariliyor
      Serial.write(mySerial.read());
    }
  }

}
```

EK-B

```
import cv2
import numpy as np
import time
import serial

ser = serial.Serial(

    port='/dev/ttyAMA0',
    baudrate=9600,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_ONE,
    bytesize=serial.EIGHTBITS,
    timeout=2
)


def takePhotoSend():
    cap = cv2.VideoCapture(0)

    ret, frame = cap.read()

    original_photo = cv2.resize(frame, (1000, 800))
    cv2.imwrite("original.jpeg",original_photo)

    resized_image = cv2.resize(frame, (100, 80))
    cv2.imwrite("1.jpeg",resized_image)
    img = cv2.imread("1.jpeg",cv2.IMREAD_GRAYSCALE)


    cv2.imwrite("raspi.jpeg",img)
    maxX = img.shape[0]
    maxY = img.shape[1]

    print (maxX,maxY)
    ser.write("basladi \n")         # Baslama mesaji yollaniyor
    string = str(maxX) + " " + str(maxY) + "\n"
    ser.write(string)               # Cekilen resmin boyutlari
hakkinda bilgiler
    for i in range(maxX):
        for j in range(maxY):
            ser.write("%d \n"  %img[i][j])      # 8 bit olarak
resimin her pixeli grayscale olarak yollaniyor


    ser.write("\nbitti \n")            # Bitti mesaji yollaniyor
    print ("bitti")

def takePhotoAndFouirer():
    cap = cv2.VideoCapture(0)

    ret, frame = cap.read()

    resized_image = cv2.resize(frame, (100, 80))
    cv2.imwrite("1.jpeg",resized_image)
    img = cv2.imread("1.jpeg",cv2.IMREAD_GRAYSCALE)
    cv2.imwrite("no_f.jpeg",img)

    f = np.fft.fft2(img)
    fshift = np.fft.fftshift(f)
```

```
    magnitude_spectrum = 20*np.log(np.abs(fshift))

    cv2.imwrite("f.jpeg", magnitude_spectrum)
    imgF = cv2.imread("f.jpeg")

    maxX = imgF.shape[0]
    maxY = imgF.shape[1]

    print (maxX,maxY)
    ser.write("basladi \n")  # Baslama mesaji yollaniyor
    string = str(maxX) + " " + str(maxY) + "\n"
    ser.write(string)  # Cekilen resmin boyutlari hakkinda bilgiler
    for i in range(maxX):
        for j in range(maxY):
            ser.write("%d \n" % imgF[i][j][0])  # 8 bit olarak
resimin her pixeli grayscale olarak yollaniyor
    ser.write("\nbitti \n")  # Bitti mesaji yollaniyor
    print ("bitti")


while 1:

        bytesToRead= ser.inWaiting()
        print (ser.read(bytesToRead))
        if ser.readline() == "1":
            try:
                print ("girdi normal")
                takePhotoSend()
            except:
                print "excepte girdi"
                ser.flushInput()
                ser.flushOutput()
                pass

        if ser.readline() == "2":
            try:
                print ("girdi Fouirer")
                takePhotoAndFouirer()
            except:
                print "excepte girdi"
                ser.flushInput()
                ser.flushOutput()
                pass
```

EK-C

```
import serial
import cv2
import numpy as np
from PIL import Image
import threading, time
from msvcrt import getch

ser = serial.Serial("COM11", 115200)

key = "0"
imgs = []

yazmaDurumu = 0


def thread1():
    global key
    lock = threading.Lock()
    while True:
        with lock:
            key = getch()
            if key == "1":
                ser.write("1")
                key = "1"
                yazmaDurumu = 1

            elif key == "2":
                ser.write("2")
                key = "2"
                yazmaDurumu = 1
            elif key == "0":
                ser.write("0")
                key = "0"
                yazmaDurumu = 0


def resmiUret():
    del imgs[-1]  # sondaki elemani sil bitti kelimesi
    del imgs[0]  # bastaki elemani sil  girdi kelimesi

    maxlar = imgs[0].split()
    maxX = int(maxlar[0])
    maxY = int(maxlar[1])
    del imgs[0]     # pixel matrisinin max degerleri

    array = np.zeros((maxX, maxY))
    counter = 0

    for i in range(maxX):
        for j in range(maxY):
            array[i][j] = imgs[counter]
            counter = 1 + counter

    cv2.imwrite("C:\Users\Emre\Desktop\Bitirme
Projesi\Python\ground_station\images\im1.jpg", array)
    im = Image.fromarray(array)
    im.convert('RGB').save("C:\Users\Emre\Desktop\Bitirme
Projesi\Python\ground_station\images\im2.jpeg")
```

```
    print "resim uretildi"
    return

def resmiUretFourier():
    del imgs[-1]  # sondaki elemani sil bitti kelimesi
    del imgs[0]  # bastaki elemani sil  girdi kelimesi

    maxlar = imgs[0].split()
    maxX = int(maxlar[0])
    maxY = int(maxlar[1])
    del imgs[0]  # pixel matrisinin max degerleri

    array = np.zeros((maxX, maxY))
    counter = 0

    for i in range(maxX):
        for j in range(maxY):
            array[i][j] = imgs[counter]
            counter = 1 + counter

    cv2.imwrite("C:\Users\Emre\Desktop\Bitirme
Projesi\Python\ground_station\images\imf1.jpg", array)
    im = Image.fromarray(array)
    im.convert('RGB').save("C:\Users\Emre\Desktop\Bitirme
Projesi\Python\ground_station\images\imf2.jpeg")

    print "resim uretildi"
    return

def veriAl():
    baslama = 0
    while 1:
        x = ser.readline()

        if x == "basladi \n":
            print ("basladi")
            baslama = 1

        if baslama == 1:
            imgs.append(x)
            if x == "bitti \n":
                print ("bitti")
                break

        if yazmaDurumu == 0:
            print x

    print "resmi uretmeye gecti"




def ana():
    global key
    while 1:
        print "-"
        try:
            veriAl()
        except:
```

```
            print "exceptte"
            pass
        try:
            print key
            if key == "1":
                resmiUret()
                print "normal resim uretildi"

            elif key == "2":
                resmiUretFourier()
                print "fourier resim uretildi "
        except:
            print "resmiUret exceptte"
            pass
        del imgs[:]


threading.Thread(target=thread1).start()
ana()
```

ÖZGEÇMİŞ

Ad Soyad: Emre ERBUĞA

Doğum Yeri ve Tarihi: İzmir, 22/11/1993

Adres: Sarıyer, İstanbul

Lisans Üniversitesi: İstanbul Teknik Üniversitesi /
Elektronik ve Haberleşme Mühendisliği (2012-2017)