T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MİKRODENETLEYİCİ TABANLI AKILLI ARAÇ KAZA TESPİT VE GÖSTERİM SİSTEMİ

ATILAY ARSLANCAN

YÜKSEK LİSANS TEZİ
ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ ANABİLİM DALI
ELEKTRONİK PROGRAMI

DANIŞMAN
YRD. DOÇ. DR. TUNCAY UZUN

İSTANBUL, 2017

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

MİKRODENETLEYİCİ TABANLI AKILLI ARAÇ KAZA TESPİT VE GÖSTERİM SİSTEMİ

Atılay ARSLANCAN tarafından hazırlanan tez çalışması 09.06.2017 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Haberleşme Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Yrd.Doç.Dr. Tuncay UZUN

Yıldız Teknik Üniversitesi

Jüri Üyeleri

Yrd.Doç.Dr. Tuncay UZUN

Yıldız Teknik Üniversitesi _____

Prof. Dr. Herman SEDEF

Yıldız Teknik Üniversitesi _____

Doç. Dr. Mehmet SAĞBAŞ

Yeniyüzyil Üniversitesi _____

ÖNSÖZ

Günümüzde önemi gitgide önemi artan bildirim sistemlerine yönelik bu yüksek lisans tez çalışmasında bana önderlik eden sayın hocam Yrd. Doç. Dr. Tuncay UZUN' a teşekkürlerimi sunarım.

Eğitim öğretim kapsamında emeği geçen değerli Yıldız Teknik Üniversitesi Öğretim Üyelerine de sonsuz teşekkür ederim.

Ayrıca, mühendis olarak örnek aldığım sayın Sami ÖZGÜR'e, meslek hayatıma önemli katkılar sağlayan, vizyon olarak bana liderlik eden saygıdeğer Komutanım Yüksek Mühendis Yarbay Murat Ali ÇÖMEZ'e ve çalışmalarımda bana destek olan değerli büyüğüm, silah arkadaşım Deniz Yüzbaşı İbrahim ALTIN'a teşekkürlerimi arz ederim. Kiymetli tecrübelerini aktaran kadim dostum Bilal KOCAKUŞAKLAR'a teşekkürü bir borç bilirim.

Son olarak, uygun çalışma ortamını sağlayan, stresli anlarımda beni telkin eden ve test faaliyetlerine bizzat katkı sağlayan değerli eşim Sahavet ARSLANCAN'a teşekkürler.

Mayıs, 2017

Atılay ARSLANCAN

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ viii

KISALTMA LİSTESİ ix

ŞEKİL LİSTESİ xi

ÇİZELGE LİSTESİ v

ÖZET vi

ABSTRACT viii

BÖLÜM 1

GİRİŞ 1

1.1 Literatür Özeti 1
1.2 Tezin Amacı 4
1.3 Orijinal Katkı 4

BÖLÜM 2

KAZA TESPİTİ VE YER BİLDİRİM SİSTEMİ 6

2.1 Kazanın Tespit Edilmesi 6
2.1.1 İvme Algılayıcıları 7
2.1.2 İvme Verisi 11
2.2.3 Kaza Şartları 12
2.2 Kaza Yerinin Tespit Edilmesi 14
2.2.1 Küresel Konumlama Sistemi (GPS) 15
2.2.1.1 Küresel Konumlama Sisteminin Ana Bölümleri 15
2.2.2 Küresel Konumlandırma Sisteminin Çalışma Prensibi 17
2.2.2.1 Sinyallerin Geliş Sürelerinin Ölçümü 17
2.2.2.2 Konumun Hesaplanması 18
2.2.3 GPS Verisi 19
2.3 Kaza Yerinin Bildirilmesi 20
2.3.1 Mobil İşletişim Sistemi 20
2.3.1.1 Mobil Terminal 22

2.3.1.2 Baz İstasyonları Altyapısı 22
2.3.1.3 Şebeke Yönetim Altyapısı 23
2.3.1.4 GSM'de hücresel yapı 24
2.3.1.5 GSM'de Erişim ve Frekans Kanalları 24
2.3.1.6 GSM'de kimlik bilgileri 25
2.3.2 SMS (Short Message Service) 26
2.4 Sistemin Kontrolü ve Yönetimi 27

BÖLÜM 3

KAZA TESPİT VE KAZA YERİ BİLDİRİM SİSTEMİ DONANIMI 29

3.1 Giriş 29
3.2 Denetleme ve Kontrol Sistemi Donanımı 29
3.3 Algılayıcı Donanımı 32
3.4 Panik Buton Donanımı 34
3.5 Konumlandırma ve Haberleşme Donanımı 35
3.6 Antenler 37
3.6.1 Şebeke Anteni 37
3.6.2 Konumlandırma Anteni 37

BÖLÜM 4

KAZA TESPİT VE KAZA YERİ BİLDİRİM SİSTEMİ YAZILIMI 39

4.1 Giriş 39
4.2 Mikrodenetleyici Yazılımı 41
4.3 Ana Kontrol Yazılım 43
4.3.1 Tanımlanma 43
4.3.2 Kurulum 44
4.3.3 İşlem 45
4.4 Kütüphaneler 47
4.4.1 ADXL345 Kütüphanesi 47
4.4.2 SIM808 (Adafruit) Kütüphanesi 48

BÖLÜM 5

SONUÇ VE ÖNERİLER 49

KAYNAKLAR 52

EK-A

ARDIUNO UNO BAĞLANTI NOKTALARI 55

EK-B

ARAÇ KAZA TESPİT VE GÖSTERİM SİSTEMİ YAZILIMI 56

B-1 Ana Kontrol Yazılım 56
B-2 ADXL345 Kütüphanesi 62
B-3 Adafurit 68

ÖZGEÇMİŞ 100

SİMGE LİSTESİ

7/24	7 gün 24 saat
±	Artı/eksi
+	Artı
^{o}	Derece
~	Yaklaşık
C	Celsius (selsiyus)
c	Işık hızı
dT	Hata
G	Yer çekimi-Gravity
R_x	X eksenindeki ivme değeri
R_y	Y eksenindeki ivme değeri
R_z	Z eksenindeki ivme değeri
X_i	Uydu konumları
Y_i	Uydu konumları
Z_i	Uydu konumları
V_{in}	Giriş voltajı

KISALTMA LİSTESİ

AMPS	Advanced Mobile Phone Service
ASCII	American Standard Code for Information Interchange
AT	ATtention
AUC	Authentication Center
BSC	Base Station Controller
BTS	Base Transciever Station
C/A	Coarse/Acquisition
CC	Country Code
CS	Circuit Switched
CGI	Cell Global Identity
CID	Cell Identity DCS Digital Cellular System
dBm	Decibel-Milliwatts
EDGE	Enhanced Data Rate for GSM Evolution
EEPROM	Electronically Erasable Programmable Read-Only Memory
EIR	Equipment Idendity Register
FAC	Final Assembly Code
FDMA	Frequency Division Multiple Access
FMV	Fevziye Mektepleri Vakfı
GLL	Geographic Position - Latitude/Longitude
GLONASS	Global Navigation Satellite System
GND	Ground
GNSS	Global Navigation Satellite System
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global Systems for Mobile Communication
HB	Hactback
HLR	Home Location Register
HSCSD	High Speed Circuit Switched Data
HTTP	Hyper-Text Transfer Protocol
IDE	Integrated Development Environment
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
LAC	Location Area Code
MCC	Mobile Country Code

MEMS	Mikroelektromekanik Sistemler
MNC	Mobile Network Code
MOSI	Master Out Slave In
MSC	Mobile Switching Center
MSIN	Mobile Subscriber Identification Number
MSISDN	Mobile Station ISDN Number
NAVSTAR	NAVigation System Using Time and Ranging
NDC	National Destination Code
NET	Network
NMC	Network Management Center
NMEA	National Marine Electronics Association
NMT	Nordic Mobile Telephone ODBOn-Board Diagnostic Systems
OMC	Operationand Maintenance Center
P	Precision
PRN	Pseudo Random Noise
PWM	Pulse with Modulation
RX	Receive
SCK	Serial Clock
SIM	Subscriber Identity Module
SMS	Short Message Service
SN	Subscriber Number
Sn	Saniye
SNR	Serial Number
SRAM	Static Random Access Memory
SMT	Surface Mount Technology
Sp	Yedek
SS	Slave Select
STP	Signalling Transfer Point
SUV	Sport Utility Vehicle
TAC	Type Approval Code
TACS	Total Access Communication System
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
TMSI	Temporary Mobile Subscriber Identity
TTL	Transistor–Transistor Logic
TUİK	Türkiye İstatistik Kurumu
TX	Transmit
UART	Universal Asynchronous Receiver/Transmitter
USB	Universial Serial Bus
VLR	Visitor Location Register

ŞEKİL LİSTESİ

Sayfa

Şekil 2. 1 Kaza Tespit ve Algılama Sistemi Mimarisi 6
Şekil 2. 2 MikroElektroMekanik sensör (MEMS) [16] 7
Şekil 2. 3 İvmeölçer kutu ve küre modelinin yerçekimsiz alandaki görünümü 8
Şekil 2. 4 İvmeölçerin X+ yönünde hareketi sonucunda uygulanan kuvvetler 9
Şekil 2. 5 İvmeölçerin düz yerleştirilmesi sonucunda uygulanan kuvvetler 10
Şekil 2. 6 Y+ yönünde 45 derecelik dönme sonucunda uygulanan kuvvetler 10
Şekil 2. 7 İvmeölçerden elde edilen ivme vektörünün görünümü 11
Şekil 2. 8 Aracın koordinat diyagramı 12
Şekil 2. 9 Aracın sağ/sol eksendeki maksimum devrilme açısı 14
Şekil 2. 10 Uydu üzerinden konum tespiti 14
Şekil 2. 11 GPS uyduları 15
Şekil 2. 12 Uydulardan gelen sinyallerin geliş süresinin ölçümü 18
Şekil 2. 13 GPS ile konum hesaplanması 18
Şekil 2. 14 GSM mimari yapısı [32] 22
Şekil 2. 15 GSM hücresel yapısı 24
Şekil 2. 16 GSM şebekesi üzerinde SMS iletimi blok şeması 27
Şekil 2. 17 Mikrodenetleyici ve birimleri 28
Şekil 3. 1 Kaza Donanıma ait blok diyagram 29
Şekil 3. 2 ADXL345 uç konfigürasyonu ve fiziksel görünümü 33
Şekil 3. 3 ADXL345 İvme Algılayıcısı ile Arduino Uno mikrodenetleyici donanım geliştirme kartının bağlantısı 34
Şekil 3. 4 “Push-Buton“ donanımın Arduino Uno mikrodenetleyici geliştirme kartına bağlantısı 35
Şekil 3. 5 İvmeölçerin SIM 808 tümleşik GSM/GPS devre kartının Arduino Uno mikrodenetleyici donanım geliştirme kartıyla bağlantısı 37
Şekil 3. 6 GPS ve GSM Antenlerinin Fiziksel Görünümü 38
Şekil 4. 1 Kaza tespit ve gösterim sistemi yazılımına ait akış şeması 40
Şekil 4. 2 Kütüphane tanımlaması 43
Şekil 4. 3 Değişkenlerin tanımlaması 44
Şekil 4. 4 Kurulum işlemleri 44
Şekil 4. 5 İvme değerini okunması 45
Şekil 4. 6 Bileşke ivmenin hesaplanması 45
Şekil 4. 7 Eğimin hesaplanması 45
Şekil 4. 8 Panik butonun kontrol edilmesi 45
Şekil 4. 9 Konum bilgisinin alınması 45

Şekil 4. 10 Konum Bilgisinin kaydedilmesi alınması 46
Şekil 4. 11 Kaza ve acil durum sorgulama 46
Şekil 4. 12 SMS gönderme 47

ÇİZELGE LİSTESİ

Sayfa

Çizelge 1. 1 Ticari/Endüstriyel Takip ve Bildirim Ürünleri 2
Çizelge 2. 1 Araçların Fiziksel Özellikleri ve Kaza Anındaki Hız, Mesafe ve İvme Değeri 13
Çizelge 3. 1 Arduino Uno Donanım Geliştirme Kartının Teknik Özellikleri 30
Çizelge 4. 1 AT Komutları 43
Çizelge 5. 1 Yapılan Çalışmanın Diğer Çalışmalar ile Karşılaştırma Çizelgesi. 51

ÖZET

MİKRODENETLEYİCİ TABANLI AKILLI ARAÇ KAZA TESPİT VE GÖSTERİM SİSTEMİ

Atılay ARSLANCAN

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Yrd. Doç. Dr. Tuncay UZUN

Trafik kazalarında can kayıplarının önüne geçilmesindeki en önemli faktörlerden biri olan erken müdahalenin, daha etkin olarak yapılabilmesi amacıyla geliştirilen kaza tespit ve bildirim sistemlerinin önemi gün geçtikçe artmaktadır.

Bu önem ışığında yapılan tez çalışmasında, mikrodenetleyici tabanlı bir kaza tespit ve gösterim sistemi tasarlanmıştır. Sistemde, "Atmega328" mikrodenetleyicisine sahip bir Arduino geliştirme platformunda MEMS tabanlı 3 eksenli bir ivmeölçer (ADXL345) ve GPS teknolojisini Quad-Band GSM/GPRS modem ile birleştiren bir modül (SIM 808) kullanılmıştır. Sistem kontrolünü sağlayan program, Arduino platformunda C dilinde yazılmıştır.

Sistemin çalışma prensibi, ivme takibiyle kaza tespitinin yapılması, kaza anındaki konum bilgisinin daha önceden belirlenmiş bir alıcıya SMS ile gönderilmesine dayanmaktadır. Sistem üzerinde yer alan panik buton donamıyla araç kullanıcısının zor durumlarda SMS ile konum gönderebilmesi sağlanmıştır.

Sistem karayolları üzerinde test edilmiş olup açık alanlarda haberleşme uydularına erişim sağlanmaktadır. Tüneller ve kapalı alanlarda yaşanacak kazalar esnasında konum bilgisinin gönderilmesine yönelik olarak konumlandırma uydularından alınan son konum bilgisi, mikrodenetleyicinin EEPROM'larında saklanmaktadır. Karayollarında yapılan testler boyunca, şebeke sinyalinin SMS gönderimi için yeterli seviyede olduğu da görülmüştür.

Bu çalışma, insansız hava aracı, zeplin, uçak ve helikopter gibi hava araçları uygulamalarında da kullanılabilir. Bununla birlikte bu tür uygulamalarda araçla ilgili konum, yükseklik ve hız bilgilerinin yanı sıra bulunduğu ortama ait sıcaklık, basınç, nem vb. bilgiler de alınabilir. Ayrıca sistem, üzerinde yapılacak çalışma ile sivil ve askeri hava araçlarında klasik arama kurtarma donanımlarına alternatif olabilecektir.

Anahtar Kelimeler: Kaza tespit, kaza bildirim, SMS ile konum gönderimi, ivmeölçer, GSM modem, GPS navigasyon.

ABSTRACT

MICROCONTROLLER BASED INTELLIGENT VEHICLE ACCIDENT DETECTION AND MONITORING SYSTEM

Atılay ARSLANCAN

Department of Electronics and Communications Engineering

MSc. Thesis

Adviser: Asst. Prof. Tuncay UZUN

Accident detection and notification systems, which are developed in order to be able to perform early intervention more effectively, which is one of the most important factors for preventing loss of lives in traffic accidents, are increasing day by day.

In this thesis study, a microcontroller based accident detection and display system is designed. The system uses a MEMS based 3-axis accelerometer (ADXL345) and a module (SIM 808) that combines GPS technology with Quad-Band GSM / GPRS modem in an Arduino development platform with "Atmega328" microcontroller. The system control program is written in the C language on the Arduino platform.

The working principle of the system is to detect the accident by acceleration and to send the position information of the accident to SMS to a predetermined receiver. The panic button on the system allows the driver to send the location via SMS in difficult situations.

The system has been tested on highways and access to telecommunications satellites is provided in open areas. The last position information received from positioning satellites for the transmission of location information during tunnels and accidents occurred in confined spaces is stored in the EEPROMs of the microcontroller. During the tests on the road, it was also seen that the network signal was at a sufficient level for sending SMS.

This work can also be used in aeronautical applications such as unmanned aerial vehicles, airships, airplanes and helicopters. However, in such applications, information about the position, height and speed of the vehicle as well as the temperature, pressure, humidity, etc., Information can also be taken. In addition, the system will be an alternative to conventional search and rescue equipment in civilian and military aircraft with the work to be done.

Keywords: Accident detection, accident notification, location sending by SMS, accelerometer, GSM modem, GPS navigation.

BÖLÜM 1

GİRİŞ

1.1 Literatür Özeti

Son yıllarda, özellikle Küresel Konumlandırma Sistemi (Global Positioning System-GPS) teknolojisindeki gelişmelere paralel olarak gerçek zamanlı takip, yönetim ve bildirim sistemleri üzerinde birçok araştırmacı ve tasarımcı çalışmalar yapmıştır. GPS ve Mobil İletişim Sistemi (Global Systems for Mobile Communication-GSM) temelli uygulamalar ile ilgili birçok yapılan çalışmalara göz atıldığında; 2009 yılında yapılan bir çalışma da GPS teknolojisinden yararlanarak web tabanlı konum tespit si stemi geliştirilmiştir [1]. Aygen ve arkadaşları tarafından yapılan çalışmada ise kısa mesaj servisi (SMS) tabanlı GPS araç takip sistemi geliştirilmiştir. Birden fazla aracın harita üzerindeki konumlarının takibi bu sistem yardımıyla sağlanmıştır [2]. Takip sistemlerinde kaydedilen gelişmeler güvenlik projelerini tetiklemiştir. 2016 yılının nisan ayında yapılan çalışmada aracın çalınması halinde araç sahibi tarafından atılan SMS ile araç durdurulması ve aracın konumu tespit edilmesi sağlanmıştır [3]. Yapılan çalışmalar ışığında ticari işletmeler için yakıt tüketiminin anlık takibinin sağlandığı ve ayrıca sürücü personelin performansının analiz edildiği ürünler de piyasada yerini almıştır. Bu proje ve ürünler genel olarak mikrodenetleyeci tabanlı çalışmalar olup işletmeciye bilgilendirme yapılmasını amaçlamıştır.

Öte yandan, konum tespit ve araç takip sistemlerinin gelişmesine paralel olarak kaza anının algılanması ve kaza konumunun bildirilmesine yönelik yeni çalışmalar da yapılmaktadır. Bu çalışmalardan birinde hava yastığı kontrol ünitesi üzerinden kaza algılaması yapılarak konum bilgisi SMS ile gönderilmiştir [4]. Başka bir çalışmada ise 1996 ve sonrasında üretilen araçlarda mevcut elektronik kontrol ünitesinden alınan

bilgiye istinaden kazanın algılanması sağlanmıştır [5]. Bu tip çalışmalarda, piyasada yaygın olarak araçların arıza tespitinde kullanılan ODB-II cihazları üzerinden kaza algılaması yapılmıştır. Bu diagnostik cihazları bluetooth aracılığıyla akıllı telefon ile haberleşmekte olup, akılı telefonlar üzerinden konumun tespiti ve gönderimi sağlanmıştır. Ancak akıllı telefonların, kesintisiz güç (batarya tüketimi), GPS çekimi ve performans zafiyetleri nedeniyle bu tip projeler için güven tesis edememiştir.

Yapılan çalışmalar ışığında geliştirilen piyasadaki ticari/endüstriyel takip ve bildirim cihaz/sistemleri incelenmiş olup Çizelge 1.1'de ürünler hakkında bilgiler verilmiştir.

Çizelge 1. 1 Ticari/Endüstriyel Takip ve Bildirim Ürünleri

Ürün	Çalışma Prensibi/Özellikleri
Dijital Takip Cihazı "Weewell WTC100"	Cihaz, 2.4 GHz RF bandında çalışan gösterge (alıcı) ve sinyal (verici) ünitesinden oluşmaktadır. Başlıca özellikleri; Açık alanda 500 m arama menzili, Açık alanda mesafe ayarına göre yakın için 30m, orta için 75m, uzak için 100m alarm menzili, Arama yapabilmek için yön göstergesi, Menzil dışına çıkıldığında alarm ve titreşim, Sinyal ünitesinde panik/stres butonu, Alarm susturmak için buton, Hem kapalı hem de açık alanda çalışabilme, Hafif ve kolay kullanım, Montaj ya da kurulum gerektirmez (sadece bataryalar), Güvenli 2.4 GHz RF sinyal teknolojisi, Bağlantı durumu göstergesi [6].
Personel, Çocuk, Yaşlı, Hasta Takip Cihazı "GPS TRACKER"	Cihaz, GSM altyapısı çalışmakta olup, GPS uydularında alınan konum bilgisini SMS/GPRS yolu ile akılı telefon veya web tabanlı bir platforma göndermektedir. Başlıca özellikleri: İnsanları, araçları veya ekipmanları izlemek için Internet üzerinden gerçek zamanlı takip, Uzun batarya kapasitesi (2 haftalık pil ömrü), Tablet veya telefon üzerinden takip imkânı, SMS ve e-posta kanalıyla bildirim, İzleme verilerini bir yıl boyunca saklama/raporlama kabiliyeti, Google Haritalar'da ve Google Earth üzerinde görüntüleme, ergonomik tasarım [7].
GPS Destekli Araç, İnsan Takip Ve Dinleme Cihazı "MINI A8"	Android işletim sistemi üzerime tasarlanmış cihaz diğer takip cihazlarına oranla daha ergonomik yapıdadır. GSM tabanlı olan cihaz, arama yaparak 10 metrelik mesafeye kadar canlı ortam dinlemesi, GPS uyduları üzerinden SMS ile konum bilgisi göndermesi yapabilmektedir [8].

Çizelge 1. 1 Ticari/Endüstriyel Takip ve Bildirim Ürünleri (Devamı)

Araç Takip Cihazı “Piranha ATS 911”	Cihaz, GSM altyapısı çalışmakta olup, GPS uydularında alınan konum bilgisini SMS/GPRS yolu ile akılı telefon veya web tabanlı bir platforma göndermektedir. Başlıca özellikleri; 7/24 uydudan takip sayesinde hırsızlığa karşı koruma, İnternet bağlantısı olan tüm bilgisayarlardan herhangi bir program yüklemesine ihtiyaç duymadan ve ayrıca Ücretsiz Android&Apple uygulamaları ile mobil cihazlardan takip yeteneği, Aracın çalınması durumunda uzaktan motor blokaj ve yeniden çalıştırabilme imkanı, limitlenmiş hız aşımında sms ile bildirim, Park halindeki aracınızın çekilmesi durumunda (mail veya SMS ile) alarm verme, Aracını rölantide uzun bekleme yapması durumunda (mail veya SMS ile) alarm verme, Haritada bölge tanımlama yapıp, tanımlanan bölge içine araç giriş veya çıkışlarında (mail veya SMS ile) alarm verme, Aracınızın çalışma, park ve mola süresini günlük raporlama, Aracın yaptığı her sefere ait süre ve km bilgisi, Geçmişe yönelik aracın konum ve hız bilgisi otomatik raporlama, Filo yönetimine uygun profesyonel raporlama ve analiz sistemi, Akünün sökülmesi durumunda Mail/SMS bildirim, Acil durumlar için araç içine gizlenebilir panik butonu, Isı-nem ölçüm sensörü, Kapı açılma sensörü, Şoför tanıma sistemi, OBD gibi harici donanımlar ile tespit kabiliyetinin artırılması [9].
Araç Yönetim Sistemi “OBDPLUS AT-90”	Cihaz, daha çok araç filoları için geliştirilmiştir. Araçların veri hattına takılan cihaz, araçların hareket halindeyken anlık konum, hız, devir ve tüketim bilgilerini GSM üzerinden Web tabanlı bir ara yüze göndermektedir [10].
Çarpışma Algılama Uygulaması “SOSMART”	Akılı telefonlar için geliştirilen uygulama, akılı telefonların ivmeölçer donanımın kullanarak çarpışmayı algılamaktadır. Çarpışma bilgisi, uygulamanın yüklü olduğu başka bir kullanıcıya kaza ikazı yazılım ile gönderilmektedir. Ayrıca uygulamanın ara yüzünde üzerinde panik buton özelliği bulunmaktadır [11].
Araç Koruma Sistemi “GrayBOX”	Cihaz, koruma modundayken GPS üzerinden yer değiştirmeyi SMS ile bildirmektedir. Anahtarlama donanımı ile aracın çalışması ikazı sonrasında SMS ile motorun susturulması sağlanmaktadır [12].
Kaza Tespit Cihazı “MICROTRONICS TECH.-ACCIDENT DETECTION”	Araçların ön ve arka tamponlarına monte edilen indüktif algılayıcı ile kaza tespiti yapan cihaz, GSM modem üzerinde SMS ile konum bilgisi göndermektedir [13].

1.2 Tezin Amacı

Trafik kazaları, karayolları üzerinde meydana gelen, insanların yaralanması veya ölmesi ile sonuçlanabilen olaylardır. Ülkemiz karayolu ağında 2015 yılında toplam 1 milyon 313 bin 359 adet trafik kazası meydana gelmiştir. Bu kazaların 1 milyon 130 bin 348 adedi maddi hasarlı, 183 bin 11 adedi ise ölümlü ve yaralanmalı trafik kazasıdır. Yıl içerisinde meydana gelen ölümlü-yaralanmalı trafik kazalarının %75'i yerleşim yeri içinde, %25'i ise yerleşim yeri dışında gerçekleşmiştir [14]. Meydana gelen kazalara erken müdahale edilmesi can kayıplarının önlenmesinde önemli ölçüde fayda sağlayacaktır.

Yapılan bu çalışmada, araçların kaza tespitine dayalı olarak kaza yapan aracın konum bilgilerinin bir alıcıya gönderilmesiyle kaza mahalline ulaşım zamanını kısaltmayı amaçlayan bir sistem tasarlanmıştır. Tasarlanan sistem ile araçların nerede ne zaman kaza geçirdiği belirlenecek, bu sayede kaza yerine erken erişim imkanı sağlanacaktır.

Kaza mahalline biran önce ulaşılmasıyla yapılacak erken müdahale, yaralıların kurtarılmasını sağlayacak ya da durumun daha kötüye gitmesini engelleyecektir. Ayrıca olası yangın tehlikesinin bertaraf edilmesi ile diğer can ve mal kayıplarının önüne geçilecektir.

1.3 Orijinal Katkı

Yapılan çalışmada sistem mikrodenetleyici tabanlı olarak kurulmuş olup ivme algılayıcısı ile kaza tespiti sağlanmıştır. Kaza konum ise GPS ile elde edilerek, GSM altyapısı kısa üzerinden mesaj servisiyle (SMS) gönderilmektedir. Daha önce yapılan tespit, takip ve bildirim sistemlerinden farklı olarak, kazanın tespit edilmesinde ivme algılayıcısı kullanılmıştır. Mikrodenetleyici üzerinden her aracın tasarımında yer alan fiziksel özelliklerine göre kaza kriterlerinin limit ivme değerleri kullanıcı kontrollü olarak programlanabilmektedir. Ayrıca GPS uydularının kesintiye uğradığı tüneller ve kapalı alanlarda yaşanacak kazalarda konum bilgisinin gönderilmesine yönelik olarak konumlandırma uydularından alınan son konum bilgisi, mikrodenetleyicinin EEPROM'larında saklanmaktadır. Bu sayede nerede kaza yapılırsa yapılsın kazaya ait yaklaşık konum bilgisinin gönderimi bu sayede sağlanmaktadır. Daha önce yapılan çalışmalarda, GSM altyapısı üzerinden alınan konum bilgisinin değerlendirildiği

görülmektedir. Bu değerlendirme sonucunda, konumun baz istasyonuna yakın rastgele bir mevki olacağından kaza gösterim sisteminde aracı yoldan bağımsız yanlış bir noktada konumlandıracaktır.

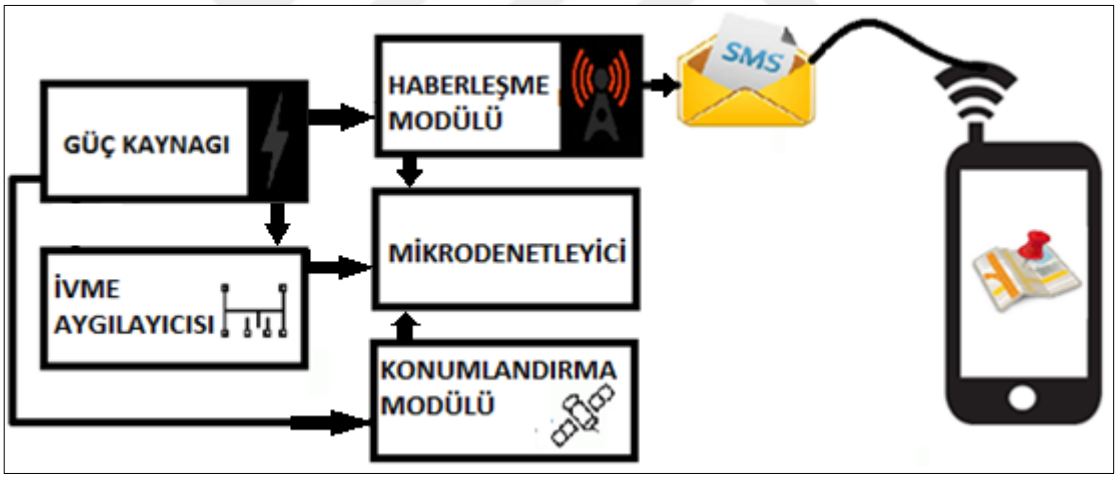
Buna ek olarak, sistem üzerinde yer alan panik butonu donamıyla araç kullanıcısının rahatsızlanması veya güvenliğin tehdit edilmesi hallerinde de konum bildirimi bu çalışmada gerçekleştirilmiştir.

Yapılan tez çalışması, sürekli konum bildirim yapan bir sistem olmayıp sadece acil durumlarda bildirim sağlayan bir sistemdir.

BÖLÜM 2

KAZA TESPİTİ VE YER BİLDİRİM SİSTEMİ

Kaza tespit ve gösterim sisteminin genel olarak çalışma prensibi, kazanın algılanması ve kaza konumunun alıcıya gönderilmesi olarak tasarlanmış olup sistemin mimarisi Şekil 2.1'de gösterilmiştir. Sistemde yer alan alıcı istasyonun mobil olması sayesinde her noktadan alıcıya bilgilendirme yapılması sağlanmıştır.

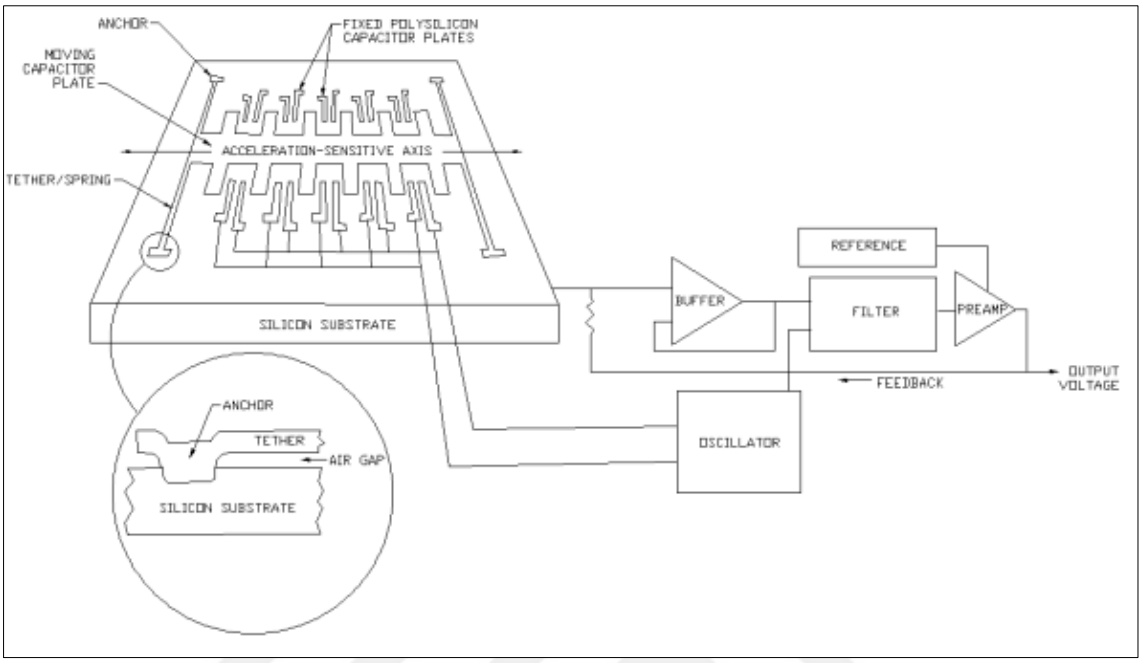


Şekil 2. 1 Kaza Tespit ve Algılama Sistemi Mimarisi

2.1 Kazanın Tespit Edilmesi

Kaza, aracın maruz kaldığı fiziksel büyüklüklerindeki ani değişimlerin algılanması ve değerlendirilmesi ile tespit edilmektedir. Algılama, fiziksel büyüklükleri elektriksel işarete çeviren algılayıcılar yapılmaktadır. Fiziksel değişimleri elektriksel işarete çeviren algılayıcılar, Mikroelektromekanik Sistemler (MEMS) olup 1990'lı yıllardan itibaren yarıiletken teknolojisinde yaşanan gelişmelerle birlikte önem kazanmıştır. Bu gelişmeler sayesinde günümüzde var olan mekanik ve elektrik sistemlerin minyatürleştirilmiş halleri mikron boyutlarında gerçekleştirilmektedir [15]. MEMS

teknolojisinin gelişimine bağlı olarak oldukça önem kazanmış algılayıcılardan birisi ivme sensörüdür. İvme algılayıcısı, kuvveti elektrik sinyaline dönüştürerek, ivme ile orantılı elektriksel (analog veya dijital) bir çıkış sinyali üretmektedir (Şekil 2.2).



Şekil 2. 2 MikroElektroMekanik sensör (MEMS) [16]

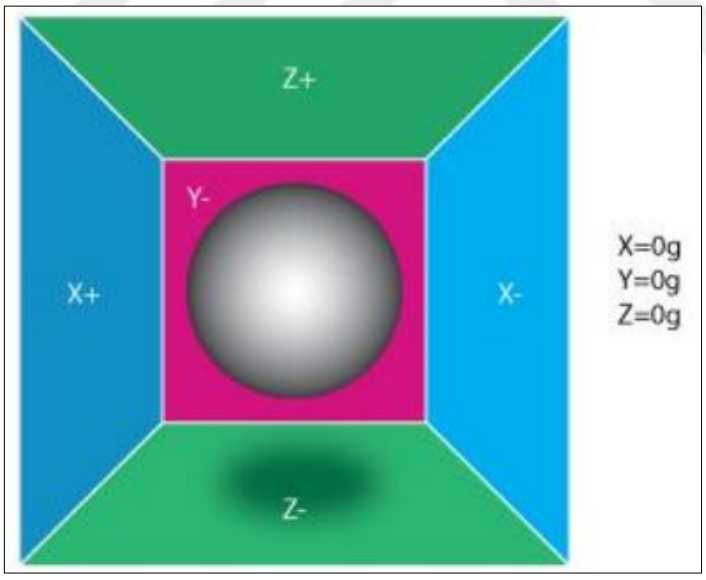
Nesnenin hızında ya da hızın yönlendiği doğrultuda bir değişim meydana gelmesinden dolayı ivme oluşur. Bu durum ivme algılayıcısı ile tespit edilmekte olup, nesnelerin hız değişiminden kaynaklı çarpma, titreşim, dönme ve eğim gibi değişken durumların ölçümü ve belirlenmesi de ivme algılayıcılar ile sağlanabilmektedir [17]. Bu durum itibariyle, ivme algılayıcısı tarafından gönderilen ivme verilerinin bir algoritma ile anlamlandırılması sonrasında anlık ivme değişimlerinden araçların trafik kazası geçirip geçirmediği proje ile tespit edilmiştir.

2.1.1 İvme Algılayıcıları

İvme algılayıcıları, bağlı bulunduğu cismin farklı eksenler üzerinde oluşan ivmelenmesini hesaplamak için kullanmaktadır. Burada ivmelenme statik (yerçekimsel) ve dinamik (harekete bağlı) olarak ikiye ayrılmaktadır. Cismin yeryüzüne göre yönelimini belirlemek için gereken veri yerçekimsel ivmedir. Bunun için hareketten kaynaklanan dinamik ivmelenmenin etkilerinin göz ardı edilip sadece yerçekimsel ivmeden elde edilen verilerin kullanılacaktır. İvmeölçerler bu iki veriyi de ölçtükleri için hareketten doğan ivmelenmenin filtrelenmesi gerekmektedir [18].

İvme algılayıcıdan elde edilen veriler yer çekimi ivmesi cinsinden tanımlanır ve bu ivmenin büyüklüğü 9,8 m/s'dir. Bu değer aynı zamanda algılayıcının ölçüm yapabileceği ivme aralığını tanımlamak için de kullanılmaktadır. Bazı algılayıcılar tek ölçüm aralığında çalışabilirken daha gelişmiş algılayıcılarda ölçüm aralığı, çözünürlük, örnekleme hızı gibi parametreleri önceden ayarlamak mümkündür [18].

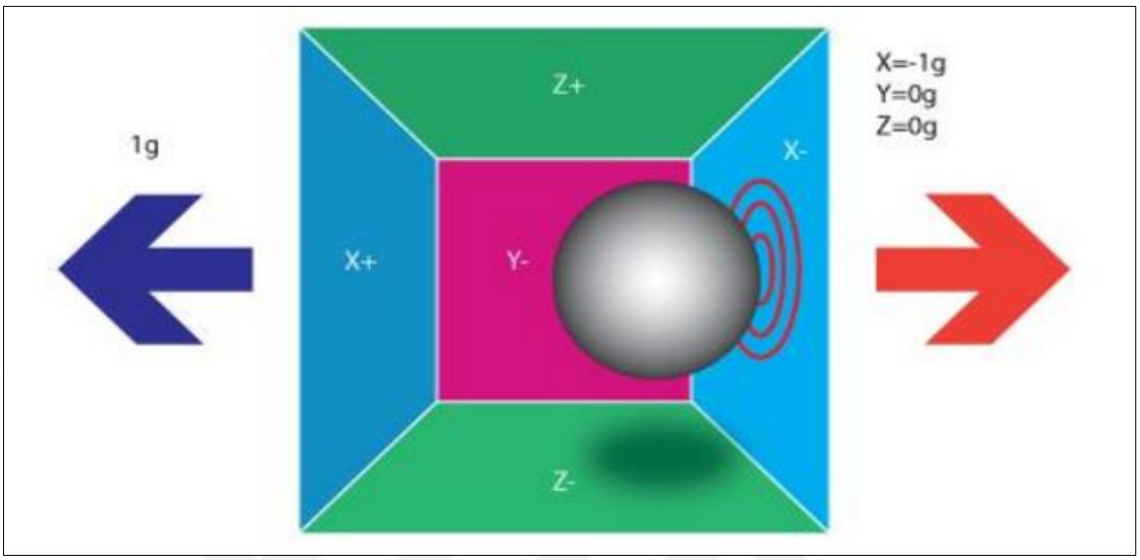
İvme algılayıcısından alınan veriyi yorumlamak için Şekil 2.4'deki gibi ortasında küre bulunan bir kutu modeli kullanılmıştır. Kutu modelde üç eksenli bir algılayıcı ele alınmış olup farklı durumlar için ölçülen ivme verileri incelenmiştir. Bu modelde kutunun kenarları eksenleri, küre ise cismin hareketine veya yönelimine göre bu eksenlere uygulanan kuvveti göstermektedir. Uygulanan kuvvetin büyüklüğü hesaplanırken kürenin ağırlığı m olarak alınmıştır. Ayrıca burada Y+ ekseni kürenin farklı durumlar için eksenlere uyguladığı kuvvetlerin görülebilmesi için gösterilmemiştir. Aşağıda verilen örneklerde her durum için ivme algılayıcısından alınan değerler modelin sağ kısmında verilmiştir. Yerçekimi kuvveti olmayan bir ortamda küre Şekil 2.3'deki gibi ortada asılı şekilde kalacaktır ve bulunduğu kutunun hiçbir kenarına kuvvet uygulanmayacaktır.



Şekil 2. 3 İvmeölçer kutu ve küre modelinin yerçekimsiz alandaki görünümü

Şekil 2.4'de kutu X+ yönünde hareket ederken kutunun duvarlarına küre tarafından uygulanan kuvvetler gösterilmiştir. Burada mavi ok hareketin yönünü, kırmızı ok ise uygulanan kuvvetin yönünü göstermektedir. Buna göre kutu mavi ok yönünde 1 g büyüklüğündeki ivme ile hızlandırılarak hareket kazandırıldığında kürenin eylemsizliğinden dolayı kutuya modelde gibi X- yönünde bir kuvvet uygulamaktadır. Bu

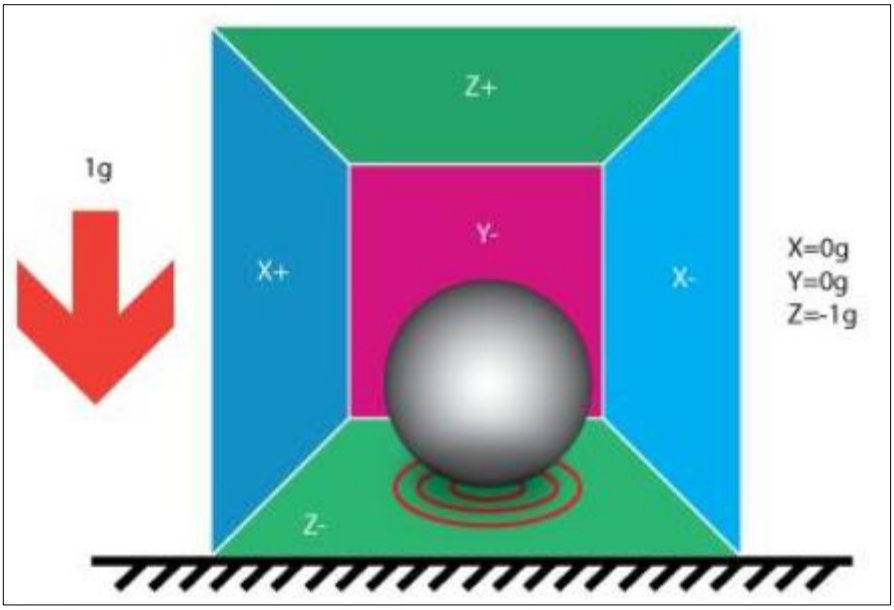
kuvvetin büyüklüğü kürenin ağırlığı ve X+ yönündeki hareketten meydana gelen ivmenin çarpımına eşittir. Buna göre m kg ağırlığına sahip bir küre X- yönünde 1 m x g büyüklüğünde bir kuvvet uygulayacaktır. Modelde ivme algılayıcısından elde edilen verinin yönü hareket yönünün tersine doğru olacaktır.



Şekil 2. 4 İvmeölçerin X+ yönünde hareketi sonucunda uygulanan kuvvetler

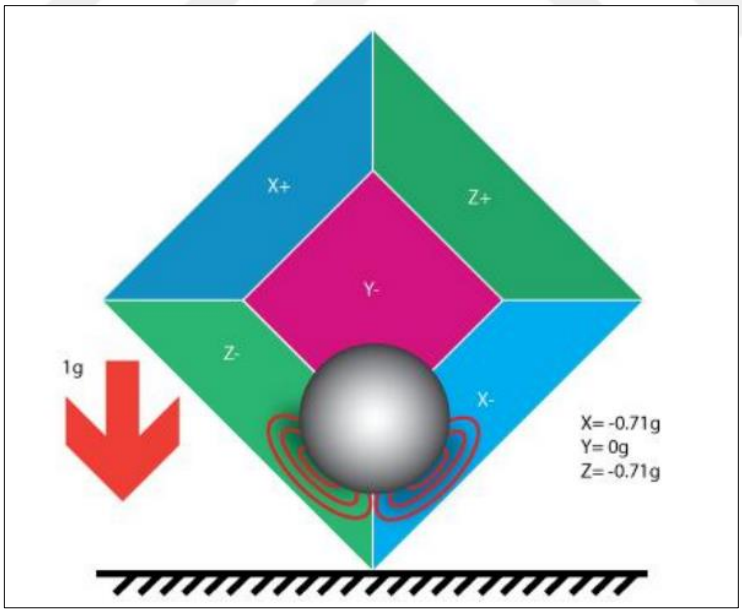
Yukarıda verilen örnekte cismin hareketi sonucunda oluşan dinamik ivmelenme ve bunun etkisiyle eksenlere uygulanan kuvvetler gösterilmiştir. Yönelimin bulunabilmesi için ise statik (yerçekimsel) ivmenin belirlenmesi gerekmektedir. Eğer ivme algılayıcısının bağlı olduğu cisim, düz bir alana konulursa dünyanın yerçekiminden dolayı küre vasıtasıyla Z- eksenine 1 m x g büyüklüğünde bir kuvvet uygulanır. Burada önemli olan ivme algılayıcısına ait eksen takımının ve yeryüzü eksen takımının eksenlerinin üst üste çakışık halde olmasıdır; zira ivme algılayıcısı ters şekilde veya başka bir eksen üzerinde yerleştirmesine istinaden kuvvetin uygulandığı eksen değişecektir. Bu nedenle uygulanan kuvvet Z- yönü yerine başka bir yönde olabilecektir. Normalde cismin statik ivmesi cisim aynı pozisyonda tutulsa dahi cismin bulunduğu konuma göre değişkenlik gösterir. Bunun nedeni yerçekimi kuvvetinin farklı yüksekliklerde değişlik göstermesidir. Fakat referans olarak kullanabilmek maksadıyla ölçüm yapılırken etki eden kuvvet 1 m x g olarak alınmaktadır.

Şekil 2.5'te düz bir alana yerleştirilen cisme uygulanan kuvvetlerin kutu ve küre modelindeki karşılığı verilmiştir. Burada görüldüğü gibi ivme algılayıcısı eksen takımı ve yeryüzü eksen takımı birleşik olduğunda küre Z- yönünde bir kuvvet uygulayacaktır.



Şekil 2. 5 İvmeölçerin düz yerleştirilmesi sonucunda uygulanan kuvvetler

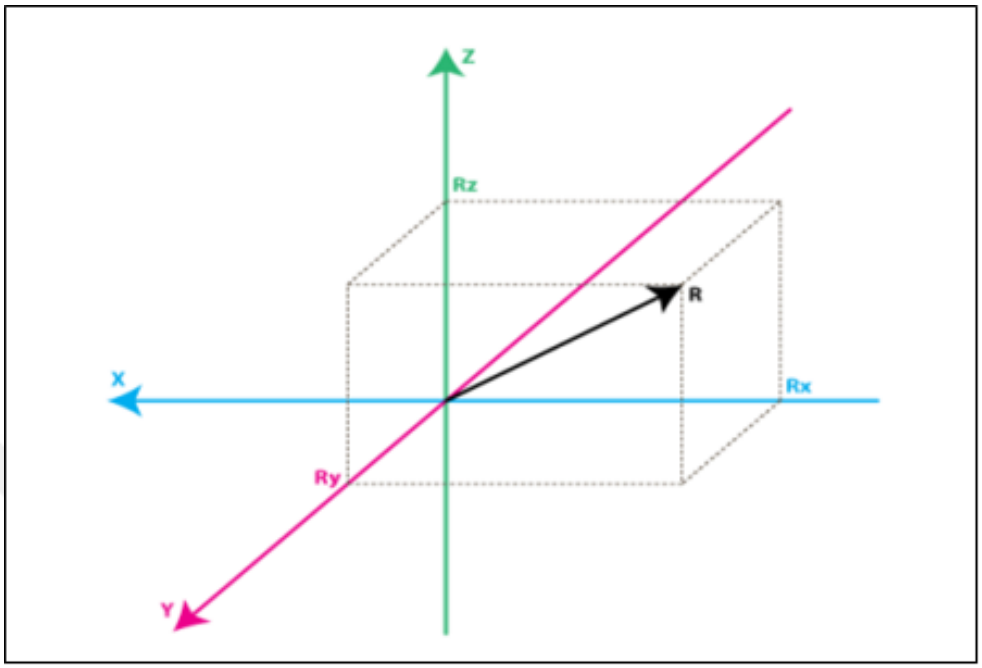
İvme algılayıcısının bağlı olduğu cisim Y ekseni üzerinde 45 derecelik bir açıyla pozitif yönde çevrildiğinde küre tarafından X ve Z eksenlerine 0,71 m x g büyüklüğünde bir kuvvet uygulanacaktır. Şekil 2.6'da söz konusu durumda uygulanan kuvvetler model üzerinde gösterilmiştir.



Şekil 2. 6 Y+ yönünde 45 derecelik dönme sonucunda uygulanan kuvvetler

İvme algılayıcısından elde edilen verilerden yönelimin matematiksel olarak hesaplanabilmesi için Şekil 2.7'de verilen koordinat sistemi modeli kullanılmaktadır. Bu modelde görülen koordinat eksenleri yeryüzü eksen takımını temsil etmektedir. Şekilde verilmiş olan R vektörü ivme algılayıcısından elde edilen ivme vektörüdür. R_x, R_y ve R_z

değerleri ise R vektörünün bileşenlerini gösterir. Kutu ve küre modelinde sağ kısımda gösterilen ivme algılayıcısının verileri R_x, R_y ve R_z bileşen vektörlerini hesaplamak için kullanılmaktadır.



Şekil 2. 7 İvmeölçerden elde edilen ivme vektörünün görünümü

Şekildeki R vektörü cismin hareketi sonucunda meydana gelen ivmenin ve yerçekimsel ivmenin bir bileşkesidir. Cismin yeryüzüne göre yöneliminin bulunabilmesi için cismin hareketinin R vektörü üzerindeki etkisinin giderilmesi gerekir. R vektörünün büyüklüğü Pisagor teoreminin yardımıyla Denklem(2.1)'deki gibi yazılabilir.

$$R^2 = R_x^2 + R_y^2 + R_z^2 \tag{2.1}$$

İvme algılayıcısının bağlı olduğu cisim hareketsiz bir şekilde düz bir alana yerleştirildiğinde elde edilen R vektörü algılayıcıya ait koordinat sistemindeki Z ekseninin dünya eksen takımına göre yönelimini göstermektedir. Cismin yeryüzüne göre yöneliminin değişmesi sonucunda bu vektörün yönü ve büyüklüğü değişmektedir.

2.1.2 İvme Verisi

Üç eksenli ivme algılayıcısı ile yapılan ölçüm sonucunda R vektörünün bileşenlerini oluşturan R_x, R_y ve R_z vektörlerinin büyüklükleri elde edilmektedir. Bu değerler mikrodenetleyiciye analog ya da dijital olarak iletmektedir. Analog veri kullanılması durumunda mikrodenetleyici, vektörlere ait büyüklükleri belirli bir aralıkta değişen

gerilim olarak görülmektedir. Dijital veri iletiminde ise vektörlerin büyüklükleri sayısal veri paketleri olarak mikrodenetleyiciye gönderilmektedir. Ölçülen ivme değerlerini m/s^2 ya da g cinsinden ifade edebilmek için ivme algılayıcıdan iletilen veriler mikrodenetleyici tarafında belli bir hassasiyet değerine bölünmüştür. Bu hassasiyet değeri algılayıcının veri föyünde belirtilmiştir [20]. Aşağıdaki denklemlerde her eksen için ivmeölçerden ölçülen verilerin ivme değerlerine dönüştürülme yöntemi verilmiştir.

$$R_x = DV_R/S \tag{2.2}$$

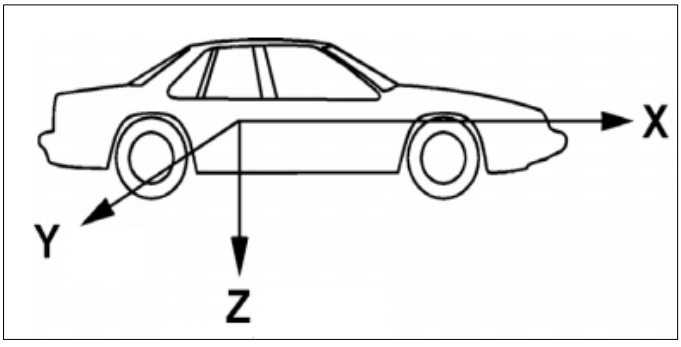
$$R_y = DV_R/S \tag{2.3}$$

$$R_z = DV_R/S \tag{2.3}$$

Yukarıdaki denklemlerde algılayıcıdan sağlanan veri "DV" olarak tanımlanmıştır. Hassasiyet değeri ise "S" şeklinde gösterilmiştir. Bu denklemler kullanılarak her eksen için ivme değeri g cinsinden hesaplanmaktadır.

2.2.3 Kaza Şartları

Sağlıklı ivme ölçümünde en önemli ayrıntı algılayıcının doğru bir şekilde konumlandırılmasıdır. Araç üzerinde oluşan olağandışı fiziksel değişimlerinin doğru işlenebilmesi için 3 eksende ölçüm kapasitesine sahip olan algılayıcı Şekil 2.8'deki gibi konumlandırılmıştır.



Şekil 2. 8 Aracın koordinat diyagramı

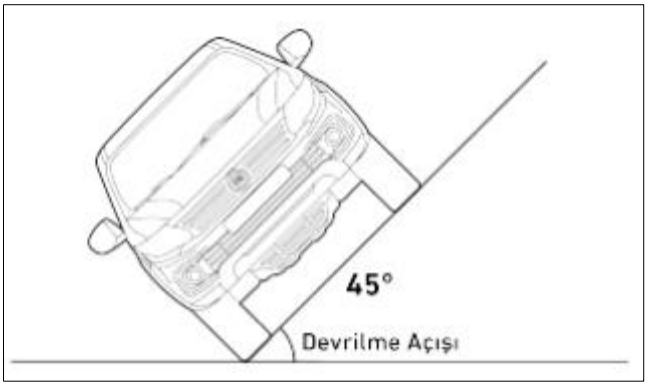
Düz bir yolda sabit hızla giden bir aracın bileşke ivme değeri yerçekimi ivmesi olan 1G'ye eşit olup eğim açısı 0 derece olarak gözlenmektedir. Aracın hızlanmasından, frenlemesinden ve eğimli yollardaki dönüşlerinden dolayı ivme ve eğim değerleri değişiklik göstermektedir. Çalışmada, araçların limit dışı ivme ve eğim değerinin değerlendirilmesi ile kaza tespiti yapılmıştır.

Çarpma ve çarpışma anında aracın ivme değeri aniden yüksek değerlere ulaşmaktadır. Kaza anındaki ivme değerlerinin belirlenmesi kapsamında yapılan çalışmalarda çarpışma anındaki minimum ivme değeri 3.7G olarak belirlenmiştir. Her marka ve model göre ivme değerleri aracın ağırlık, boyut, fren sistemi, amortisör yapısı vb. gibi fiziksel ve teknik özelliklerine göre değişiklik göstermektedir. Hava yastıklarının patlama kriterlerinin üzerine yapılan çalışma neticesinde belirlenmiş olan araç modellerine göre orta ölçekli kaza sonrasında araçların ivme değişimleri Çizelge 2.1'de gösterilmiştir [16]. Bu çalışmada 3.7G ve üzeri hissedilen ivme kaza oluşumu olarak belirlenmiştir.

Çizelge 2. 1 Araçların Fiziksel Özellikleri ve Kaza Anındaki Hız, Mesafe ve İvme Değeri

Model Yılı	Model	Sınıf	Araç Ağırlığı	Hız Farkı		İvme		Durma Mesafesi	
			(Ibs)	(mph)	(mph)	(g's)	(g's)	(in)	(in)
2010	Ford Fusion	Orta	3640	8.0	14.0	4.7	8.2	7.0	12.3
2010	Lexus Rex350	SUV	4747	8.0	14.0	5.0	8.7	6.6	11.5
2009	Ford Escape	SUV	4198	8.0	14.0	5.0	8.8	6.6	11.5
2010	Honda Insight	HB	3115	8.0	14.0	5.1	9.0	6.4	11.2
2010	Toyota Prius	Orta	3499	8.0	14.0	5.8	10.2	5.7	9.9
2007	Chevrolet Corvette	Spor	3132	8.0	14.0	6.1	10.6	5.4	9.5
2007	Chevrolet Equinox	SUV	3172	8.0	14.0	7.5	13.2	4.4	7.6

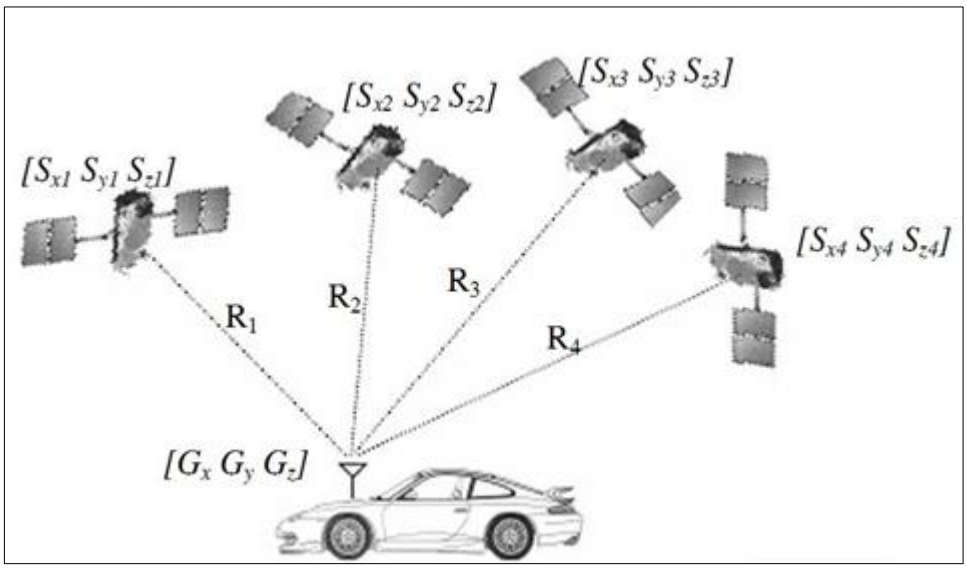
Çarpışma anında oluşan ivmeye ek olarak, araçların takla atması halinde yatay eksendeki eğim açısının tasarım limitleri dışında olması halinde de kaza tespiti yapılmaktadır. Ağırlık merkezi ideal bir araç için yatay eksende maksimum eğim açısının 45 derece olması beklenmektedir (Şekil 2.9). Ancak aracın imalatına ilişkin bir özelik var ise bu değer değişiklik göstermektedir. Çalışmada, 45 derece ve üzeri eğim açısına sahip bir aracın takla atmak suretiyle kaza yaptığı değerlendirilmiştir.



Şekil 2. 9 Aracın sağ/sol eksendeki maksimum devrilme açısı

2.2 Kaza Yerinin Tespit Edilmesi

Kaza konumun belirlenmesi, yapay uydularla konum belirleme yöntemlerinin başında gelen Küresel Uydu Seyrüsefer Sistemi (Global Navigation Satellite System-GNSS) ile sağlanmıştır (Şekil 2.10). GNSS, hali hazırda aktif olarak çalışmakta olan Amerika Birleşik Devletleri tarafından işletilen GPS (Global Positioning System), Rusya Federasyonu tarafından işletilmekte olan GLONASS (Global Navigation Satellite System) ve Avrupa ülkeleri tarafından kurulmakta olan ve ilk uyduları fırlatılan GALILEO uydu konumlama sistemlerinden oluşmaktadır.



Şekil 2. 10 Uydu üzerinden konum tespiti

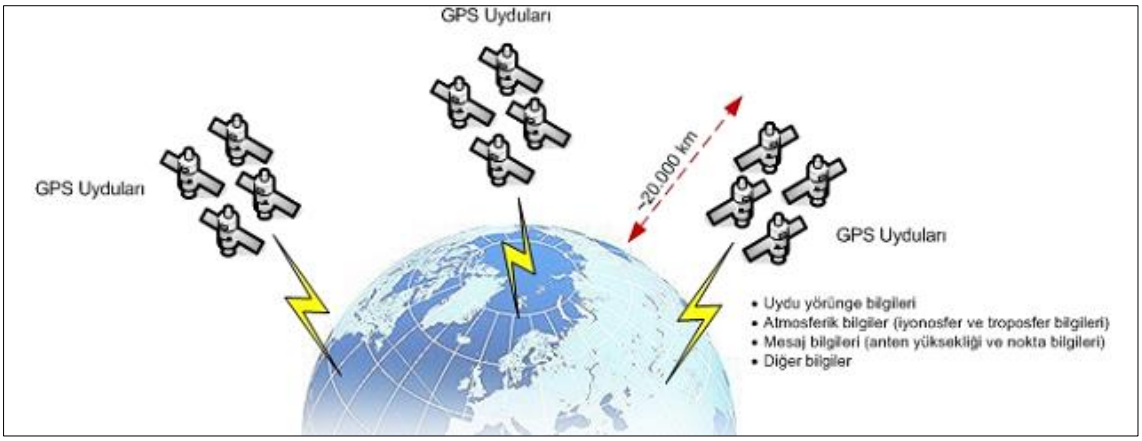
Konum belirleme amaçlı uydu sistemleri içerisinde en yaygın kullanıma sahip olan GPS uydusudur. GPS, düzenli olarak kodlanmış veri yollayan bir uydu ağı olup uydularla arasındaki mesafeyi ölçerek Dünya üzerindeki kesin yeri teşhis eder ve ayrıca kullanımı

ücretsizdir [21]. Bu sebepler doğrultusunda, çalışmada GPS'in kullanılması en efektif çözüm olmuştur.

2.2.1 Küresel Konumlama Sistemi (GPS)

Küresel Konumlandırma Sistemi (Global Positioning System – GPS) uydular ile arasındaki mesafeyi ölçerek dünya üzerindeki konumu herhangi bir zaman, yer ve her hava şartında belirlemek için tasarlanan ve sürekli olarak kodlanmış veri yollayan bir uydu ağıdır [22]. İlk çalışmalarına 1973 yılında askeri amaçla kullanmak amacıyla ABD Savunma Bakanlığınca başlanan NAVSTAR (NAVigation System Using Time and Ranging), GPS uydulardan yayınlanan radyo sinyalleri yardımıyla her türlü hava koşullarında, gece ve gündüz, süratli, hassas ve ekonomik olarak, noktalar arası görüş ihtiyacı olmaksızın üç boyutta konum belirleme sistemidir. Navigasyon ihtiyacına binaen tasarlanan sistem, konumlamanın yanı sıra zaman ve hız belirleme imkanı sunmaktadır [23].

Yön bulmak, amfibi harekatı ve füze atışları gibi tamamen askeri amaçlı kurulmuş olan bu sistem 1980'li yıllarda sivil kullanıma açılmıştır [24]. Şekil 2.11'de GPS uydularının dünya üzerinde yer alan temsili konumları gösterilmiştir.



Şekil 2. 11 GPS uyduları

2.2.1.1 Küresel Konumlama Sisteminin Ana Bölümleri

Küresel konumlama sistemi üç ana bölümden oluşmaktadır.

- **Uzay Bölümü:** Sistemin uzay bölümünü, her noktadan ve her daim uygun pozisyonda en az dört uydunun görülebileceği şekilde planlanmış 24 uydudan

oluşmaktadır. Bu bölümündeki uydular modülasyonu L1 ve L2 frekanslarında sinyal yayınlayınlamaktadır. Uydular, 10.23 Mhz temel frekansının 154 ve 120 tam katı olan L1 ve L2 taşıyıcı sinyalleri üzerine kodlar yardımıyla uygun yörünge ve saat bilgilerini modüle ederek C/A (Coarse/Acquisition) kodu, P kodu (Precision code) ve navigasyon mesajı olmak üzere üç farklı veri üretip yayınlamaktadır [25].

Her uydunun özel bir elektronik donanım ile belli bir matematik modele göre ürettiği bu kodlara "pseudorandomnoise - PRN" adı verilir. Uyduların tanımlanmasında PRN no. ları kullanımı tercih etmektedir [26].

C/A kod veya standart kod (S kod) adı verilen, normal olarak yalnızca L1 üzerinden 1.023 Mhz saat hızında yayınlanan 300 m dalga boyuna sahip sinyal, tüm kullanıcıların kullanımına açıktır. Kod denilen L1 ve L2 üzerinden 10.23 Mhz ile yayınlanan yaklaşık 30 m dalga boyuna sahip sinyal ise esas itibariyle askeri amaçlara yönelik kullanılmaktadır.

Her bir uydu, senkronize zaman sinyallerini, tüm uydulara ait konum bilgilerini ve yörünge parametrelerine ilişkin bilgileri iki taşıyıcı frekans (L1,L2) üzerinden yayınlamaktadır. Üç farklı GPS uydusu mevcut olup bunlar; BlockI, BlockII ve BlockII-R olarak sınıflandırılmışlardır.

- **Kullanıcı Bölümü:** GPS'in kullanıcı bölümünü, yeryüzünde kullanılan alıcı gruplarından oluşturmaktadır. Bir alıcı grubunda, anten, alıcı ve güç kaynağı bulunmaktadır. Çalışmada kullanılan kullanıcı bölümü ile bilgiler Bölüm 3'te verilmiştir.

- **Kontrol Bölümü:** GPS kontrol bölümü, yer kontrol istasyonları, ana kontrol istasyonu ve izleme istasyonları olmak üzere üç kısma ayrılmaktadır. Kontrol bölümünün görevleri, yörünge ve saat bilgilerinin belirlenmesi için uyduların izlenmek ve her bir uydudaki mesaj bilgilerinin güncellemektedir. Yeryüzünde uygun olarak dağılmış toplam 5 adet istasyon bulunmaktadır. Bunlar; Colorado Springs, ana kontrol istasyonu ve izleme istasyonu, Kwajalein, Diego Garcia ve Ascension Island yer kontrol ve izleme istasyonu, Hawai Falcon ise yalnızca izleme istasyonu olarak görev yapmaktadır. İzleme istasyonlarından bütün uydular sürekli olarak izlenmekte olup uyduların gönderdiği sinyaller kaydedilmektedir. İstasyonlarda kaydedilen meteorolojik verilerle birlikte çevrimiçi olarak Colorado Springs ana kontrol istasyonuna

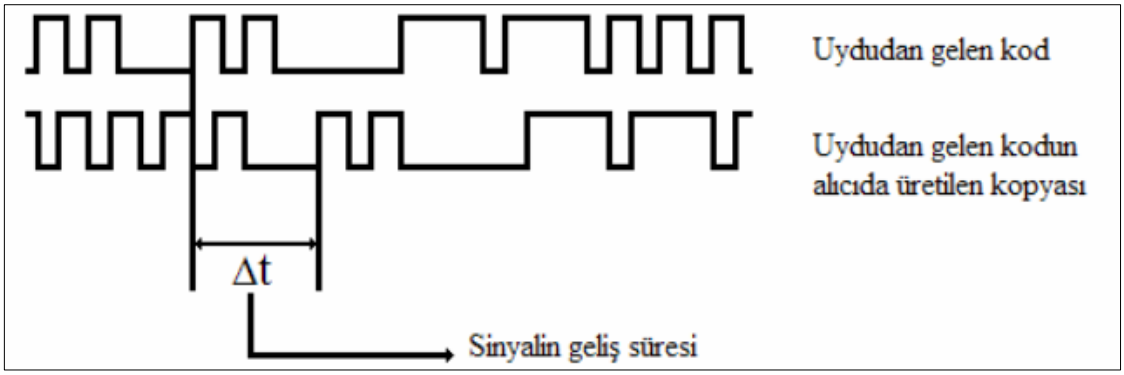
aktadırlmaktadır. Uyduların yeni yörünge ve saat bilgileri, meteorolojik veriler ile kontrol istasyonunda hesaplandıktan sonra yine çevrimiçi olarak yer kontrol istasyonlarına gönderilmektedir. Tüm bu bilgilere yer antenleri yardımıyla, S Band dalgalarıyla yer kontrol istasyonlarından her gün düzenli olarak yüklenmektedir. Yüklenen her bir mesaj 14 günlük bir süre için geçerli olup bu süre zarfında yaşanan değişiklikler uydunun konum doğruluğunda 10-200 m arasında bir sapmaya neden olmaktadır [27].

2.2.2 Küresel Konumlandırma Sisteminin Çalışma Prensibi

Sistemin çalışma prensibi kabaca şu şekilde özetlenebilir: Alıcı uydudan gelen mesafe kodlarını kullanarak sinyalin geliş süresini belirler. Sinyaller ışık hızında yayınlanmakta olup geçen sürenin ışık hızıyla çarpılması sonucu uydu ile alıcı istasyon arasındaki mesafe hesaplanmaktadır. Alıcının üç boyutlu konumu için üç bilinmeyen olduğundan üç farklı uydudan gelen sinyallerle üç farklı ölçümün yapılması gerekir. Sinyallerin geliş süresinin ölçümündeki çok küçük hatalar dahi hesaplanan konumda büyük hataların oluşmasına sebep olacaktır. Örneğin sürenin ölçümündeki 1ms hata, konumda takribi 300 km'lik bir hassasiyet kaybına karşılık gelmektedir. Bu yüzden dördüncü bir ölçüm yapılmaktadır. Hatanın da bilinmeyen olarak toplam dört bilinmeyen bu dört ölçümle hesaplanır. Böylece konumun doğru bir şekilde hesaplanabilmesi için en az dört farklı uydunun görülebilir olması gerekmektedir [28].

2.2.2.1 Sinyallerin Geliş Sürelerinin Ölçümü

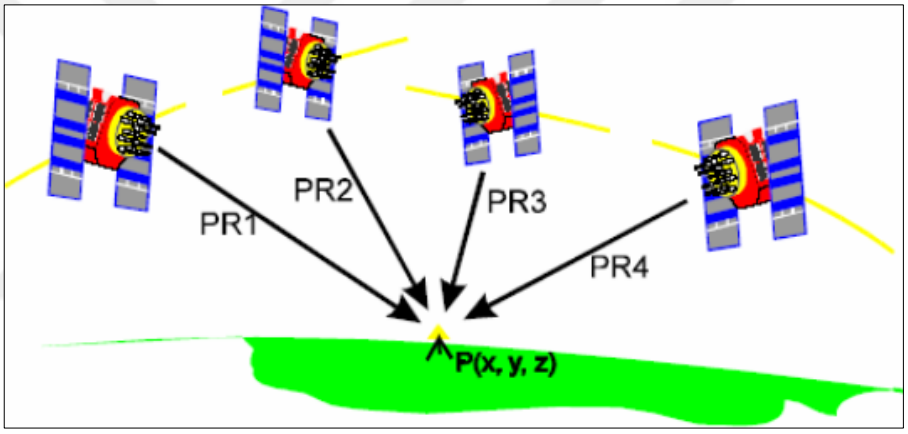
Uydudan gelen sinyallerin geliş süresinin ölçümünde mesafe tayin kodları kullanılmaktadır. Mesafe tayin kodu belli bir 1 0 bit dizisinden oluşan periyodik bir sinyaldir. Alıcı bu kodun bir kopyasını oluşturur ve uydudan gelen kodla karşılaştırır. Her iki kod eşzamanlı olacak şekilde kodun kopyasını zaman içinde kaydırır. Aradaki fark sinyalin geliş süresini verir [28]. Şekil 2.12, sinyallerin geliş sürelerinin ölçümünü göstermektedir.



Şekil 2. 12 Uydulardan gelen sinyallerin geliş süresinin ölçümü

2.2.2.2 Konumun Hesaplanması

Konumun hesaplanması için dört farklı uyduya olan mesafenin ölçülmesi gerekmektedir (Şekil 2.13).



Şekil 2. 13 GPS ile konum hesaplanması

Her uyduya olan mesafe,

$$PR_i = c * \Delta t_i \tag{2.5}$$

eşitliği ile hesaplanır [28]. Eşitlikte c, ışık hızını, PR_i ve Δt_i i = 1,2,3,4 ise uyduya olan uzaklık ve söz konusu uydudan gelen sinyalin geliş süresini göstermektedir.

Mesafe ölçümünde oluşan hata dT, uydu konumları X_i, Y_i ve Z_i ile gösterilirse alıcının üç boyutlu konumu X,Y ve Z, dördüncü bilinmeyen dT ile birlikte aşağıdaki denklem sisteminden hesaplanabilir [28].

$$(X_1 - X)^2 + (Y_1 - Y)^2 + (Z_1 - Z)^2 + dT = {PR_1}^2 \tag{2.6}$$

$$(X_2 - X)^2 + (Y_2 - Y)^2 + (Z_2 - Z)^2 + dT = {PR_2}^2 \tag{2.7}$$

$$(X_3 - X)^2 + (Y_3 - Y)^2 + (Z_3 - Z)^2 + dT = {PR_3}^2 \quad (2.8)$$

$$(X_4 - X)^2 + (Y_4 - Y)^2 + (Z_4 - Z)^2 + dT = {PR_4}^2 \quad (2.9)$$

2.2.3 GPS Verisi

GPS alıcıları, veri iletimi için standart NMEA protokolünü kullanmaktadır. Ayrıca, bazı GPS alıcılarında kendilerine özel protokoller de kullanılamaktadır. Alıcılar, konum bilgilerini NMEA cümleleri olarak iletmektedir. Bu cümlelerin bazıları hız ve yükseklik, bazıları düzeltme verisi, bazıları da uydularla ilgili bilgiler vb. verilerini içermektedir. Sadece enlem, boylam, yükseklik, konum hesaplandığı zaman ve verinin geçerli olup olmadığının anlaşılmasını sağlayan bilgileri içeren cümle en basit olandır. Her cümle "$" karakteri ile başlar, daha sonraki iki karakter veriyi gönderen aygıtı, sonraki üç karakter ise cümle türünü belirtmektedir. Bundan sonra bilgiler virgülle ayrılmış bir şekilde sıralanmakta ve cümle sonunda "*" karakteri ve iki baytlık hata değerlendirme işlemi için kullanılan doğrulama (checksum) verisi bulunmaktadır [29].

Konum bilgilerini içeren en basit cümle GLL cümlesidir. Bu cümle sadece enlem, boylam ve konumun hesaplandığı zaman bilgilerinden oluşmaktadır. Bu bilgiler şu şekildedir:

- Derece ve dakika cinsinden enlem,
- Enlem yönü (N: Kuzey, S: Güney),
- Derece ve dakika cinsinden boylam,
- Boylam yönü (E: Doğu, W: Batı),
- Greenwich zamanına göre saat, dakika ve saniye cinsinden konumun belirlendiği zaman,
- Verilerin geçerliliği,
- Doğrulama (Checksum) verisi.

Aşağıda görülen örnek GLL cümlesinde,

$GPGLL, 3736.2249,N, 02907.5900,E,185310,V,S*

Enlem 37 derece 36,2249 dakika kuzey, boylam 29 derece 7,59 dakika doğu, zaman ise Greenwich zamanına göre 18:53:10 olarak görülmektedir.

2.3 Kaza Yerinin Bildirilmesi

Kaza yerinin bildirilmesi maksadıyla kablosuz haberleşme yöntemlerinin mobil kullanımı konseptinde en yaygın olarak kullanım yer olan “Global Systems for Mobile Communication (GSM)” tercih edilmiştir. Mobil iletişim, hareketsiz ya da hareketli kullanıcıların, mekandan bağımsız (evde, işte ya da otomobilde), başka bir yerde olan bir diğer telli ya da mobil telefon abonesi ile radyo dalgaları aracılığı ile kurduğu iletişim tipidir. Projede mobil iletişim modemi üzerinden Kısa Mesaj Servisi (SMS) gönderimi vasıtasıyla kaza bildirimi sağlanmıştır. SMS ikinci kuşak cep telefonlarıyla başlayan, GSM abonelerine en çok 160 karakterden oluşan, metin mesajların gönderilip alınabilmesine imkân veren, kullanımı oldukça kolay bir servistir [30].

Bu servis ile GSM çalışma bandında yayın yapan tüm telefonlara erişim sağlanabilmektedir. SMS bu bağlamda geniş bir kullanıcı kitlesine sahiptir. Mobil olarak alıcı kaza anında konum bilgisine erişebilecektir. Çalışmada GSM’in kullanılmasındaki esas amaç, alıcının mobil olarak konum bilgisine ulaşabilir olmasıdır.

2.3.1 Mobil İşletişim Sistemi

Mobil iletişim teknolojisi alanında yapılan en büyük hamle, hücresel sistemlerin ortaya çıkmasıyla gerçekleşmiştir. Hücresel bir sistem kurma düşüncesi ilk önce Bell Labs tarafından üretilmiş ve ilk hücresel sistem olan AMPS (Advanced Mobile Phone Service) 1979 yılında Chicago'da oluşturulmuştur. Kuzey Avrupa'da ise İskandinav ülkeleri özellinde NMT (Nordic Mobile Telephone) sistemi geliştirilmiştir. AMPS'in bir uzantısı olan TACS (Total Access Communication System) İngiltere'de 1985'te hizmete girmiştir. Bu sistemler birinci nesil (1G) hücresel haberleşme sistemleri olarak kabul edilmiştir. Mobil haberleşme hizmetine duyulan yüksek talebe, mevcut analog sistemler yetersiz kalmış ve ayrıca kullanılan farklı sistemlerin uyumsuz çalışması haberleşme zafiyetine neden olmuştur. Ancak artan talebin karşılanması amacıyla geliştirilecek yeni sistemlerin maliyetinin hiçbir ülkenin tek başına yüklenemeyeceği kadar yüksek olduğu değerlendirilmiştir. Mobil haberleşmedeki artan talebe cevap verilmesi adına, bütün Avrupa ülkelerinin ortak tasarlayıp kullanacakları sayısal ve hücresel bir mobil sistem olan GSM ortaya çıkmıştır [31].

1990 yılında, 1800 MHz'de frekans bandında çalışan GSM modeli DCS 1800 (Digital Cellular System 1800) sisteminin standartlarının belirlenme çalışmaları başlamış olup çalışmaların tamamlanmasının ardından 2x75 MHz'lik band genişliğinin sağladıığı kapasite artışı elde edlimiştir.

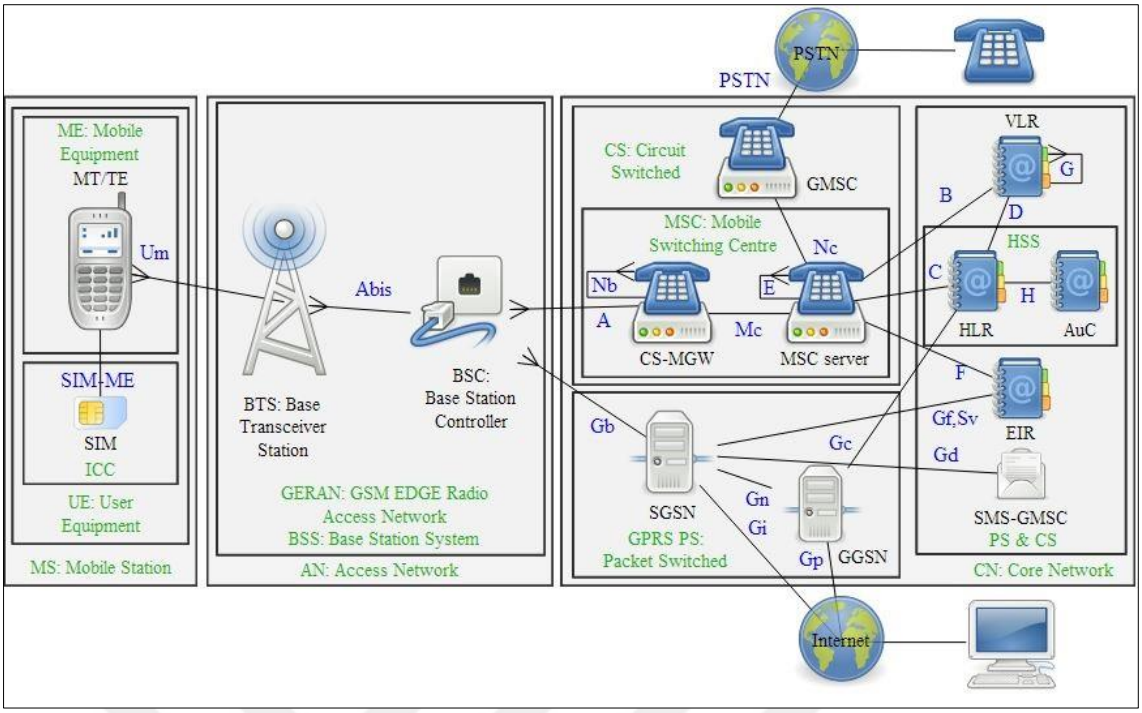
İkinci nesil (2G) hücresel sistem olarak kabul edilen GSM, sayısal olması ve radyo erişim metodlarını kullanması yönünden birçok yenilikler getirmesine karşın erişilen son nokta olmamıştır. Üçüncü nesile geçişe imkan sağlayan bir basamak olarak kabul edilmiştir.

GSM sistemlerindeki gelişmeleri, daha hızlı veri iletimine sahip HSCSD (High Speed Circuit Switched Data), GPRS (General Packet Radio Service) ve EDGE (Enhanced Data Rate for GSM Evolution) sistemleri izlemiş ve bu sistemler 2.5G olarak kabul edilmiştir.

GPRS yalnız mobil olarak internete erişimi sağlamakla kalmamış, bu erişimi sabit hat kapasitelerinin çok daha üzerinde veri iletim hızlarıyla sağlayarak üçüncü nesil sistemlere (3G) geçiş doğrultusunda, evrim çizgisinin de önemli bir basamağı olmuştur.

GSM sisteminde, yüksek hızlara ulaşmak çoklu yayılım periyodu (time-slot) tekniği ile mümkün olmuştur. GSM haberleşmesinde, her biri 200 kHz band genişliğine sahip kanalların, 8 zaman aralığına bölündüğü TDMA (Zaman Bölmeli Çoklu Erişim - Time Division Multiple Access) tekniği kullanılmış olup bu yöntem devre anahtarlamalı (Circuit Switched-CS) bağlantı olarak kabul edilmiştir. Devre anahtarlamalı bağlantı, 9.6kbps veri iletim hızına sahip olup bağlantının kurulduğu bir kullanıcıdan diğerine sabit bir trafik kanalı tanımlanmaktadır. Konuşma veya veri transferi bu kanal üzerinden yapılmaktadır. GSM şebekesinde tek bir trafik kanalının sadece bir kullanıcıya tahsis edilenmesi, şebekenin en yoğun olduğu saatlerde trafiğin de yüklü olmasına neden olamaktadır.

Şekil 2.14'de görüldüğü gibi bir GSM şebekesi temel olarak 3 ana bölümden oluşmaktadır. Bunlardan birincisi "Mobil Terminaller" (Cep Telefonu vb.), ikincisi "Baz İstasyonları Altyapısı" ve üçüncüsü "Şebeke Yönetim Altyapısı" ile buna bağlı olarak çalışan "Abone ve Terminal Veritabanı"dır.



Şekil 2. 14 GSM mimari yapısı [32]

2.3.1.1 Mobil Terminal

Mobil Terminal bir mobil telefonu veya GSM/GPRS modemdir. Bu cihazlar bir SIM (Subscriber Identity Module) kart yerleştirilerek kullanılmaktadır. SIM kart bir GSM operatörüne abone olunması sonucunda temin edilmekte olup o operatöre ait sisteme tümleştik olmayı sağlamaktadır. Bu sayede GSM operatörünün sunmuş olduğu kablosuz haberleşme servisleri kullanılmaktadır. Mobil cihazın IMEI (International Mobile Equipment Identity) adında kendine özgü bir kimlik tanımlaması mevcuttur. SIM kartta ise IMSI (International Subscriber Identity) adında, aboneyi mobil sisteme tanıtan bir numara bulunmaktadır. Ayrıca, GSM operatörü tarafından aboneye tanımladıığı yetkilendirme bilgileri de SIM kartta yer almaktadır [32].

2.3.1.2 Baz İstasyonları Altyapısı

Baz istasyonlarının alt yapısı iki farklı bölümden oluşmaktadır. Bunlar BTS'ler (Base Transciever Station) ve BSC'lerdir (Base Station Controller). Bu yapılar, sistem bileşenleri arasındaki iletişimi sağlamaktadır. BTS'nin ana görevi; uç nokta olarak tanımlanan mobil terminalden gelen mesajları alma-verme işlemleridir. BSC ise BTS'lerin radyolink imkanlarını denetleyerek, ilgili hücrelerdeki mobil terminali BTS'ye

irtibatlandırılmasını, hareketli abonelerin yer değişimlerini izleyerek aboneyi kendi BTS'leri arasında yada başka bir BSC'nin denetlediği BTS'ye aktarılmasını sağlamaktadır. Böylece, kullanıcının bulunduğu mesafeyi ve kullanımı arasındaki orantıyı kontrol ederek mobil terminalin ve BTS'nin tüketeceği gücü en aza indirmektedir [32].

2.3.1.3 Şebeke Yönetim Altyapısı

Şebeke Yönetim Altyapısı, MSC (Mobile Switching Center) ve buna bağlı birimlerden oluşmaktadır. MSC, çağrıları yönlendirme ve diğer haberleşme sistemleri ile irtibat kurma görevlerini yerine getirmektedir. Bu bölümde aynı zamanda operatörün bakım ve yönetim işlemleri yürütülmektedir. GSM servis alanı içinde bulunduğu sürece, mobil terminalin yerinin bulunması, çağrı yapması ve gelen çağrıyı yanıtlanması gibi işmler bu birim üzerinden gerçekleştirmektedir [33]. Bu işlemleri için sistemde kullanılan veri tabanları aşağıda belirtilmiştir:

Dahili adres kaydedici, (HLR-Home Location Register) çağrı yapılırken kullanılan abone bilgilerini içermektedir. Bu bilgiler, abonenin o anda hangi MSC alanında bulunduğu ve hangi hizmetlere kayıtlı olduğunu tanımlayan verileridir. Aboneni sahip olduğu numarada, bu bilgilerin hangi HLR'de kayıtlı bulunduğu tanımlanmaktadır.

Misafir Adres Kaydedicisi (VLR-Visitor Location Register) her MSC için bir VLR mevcut olup MSC'ye ait bölgede o an itibariyle var olan mobil terminallerin bazı HLR bilgilerini içermektedir.

Cihaz Kimlik Kaydedicisi (EIR-Equipment Idendity Register) mobil terminalin türünü ve IMEI numarasını kontrol etmek için gerekli verileri içermektedir. Kayıt dışı cihazlar, sorun çıkaran mobil terminaller ve çalıntı cihazlar veriler ile tespit edilmektedir. Dolayısıyla, mobil terminalin kullanılmasına izin vermek veya kullanım dışında bırakmak bu bölümden gelen onay koduyla sağlanmaktdır.

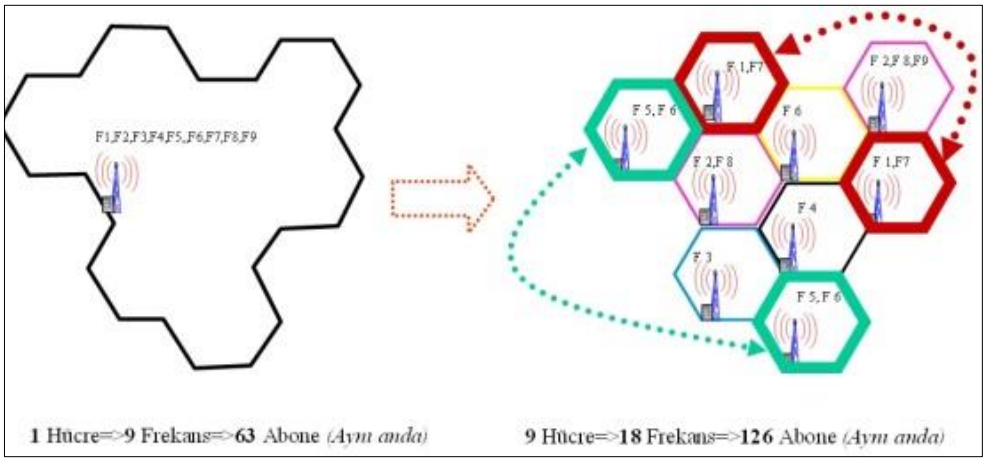
Doğrulama Merkezi (AUC-Authentication Center) frekans atlatma ve abone şifreleme yöntemlerine ilişkin ana bilgileri taşımaktadır. AUC, abonelere tanımlanmış bilgilerin doğrulanmasını sağlamaktadır. Bu sayede abonenin GSM servislerini kullanırken güvenliği güvence altına alınmaktadır [31].

İşletim ve bakım merkezi (OMC- Operationand Maintenance Center) ve iletişim ağı yönetim merkezi (NMC- Network Management Center), tüm GSM sistemine ait yapıların bir haberleşme ağı olarak çalışması ve yönetimi yürütülmektedir. OMC, cihazların her noktadaki çalışabilirliğini tarif ve takip etmekte olup arızalı cihazların kullanım dışında kalmasına sağlamaktadır. NMC ise tüm ağın en efektif şekilde kullanılması için genel strateji belirleyerek ve tüm ağın faaliyetlerini takip etmektedir.

2.3.1.4 GSM'de hücresel yapı

GSM için sınırlayıcı unsurlardan bir tanesi mobil iletişim için tahsis edilen frekans bandının oldukça dar olmasıdır. Şehirlerde eş zamanlı yapılan görüşmelerde ihtiyaç duyulan kapasite ancak şebekenin bazı bölümlerinde aynı radyo frekanslarının tekrar kullanılmasıyla sağlanmaktadır [34].

Bu amaçla sistem, bal peteğine benzer yapıdan bitişik çalışan hücrelerden tasarlanımış olup her hücre, düşük çıkış gücü ve kısa mesafeli radyo sinyallerine sahip baz istasyonu ile çalışmaktadır (Şekil 2.15). Bu sayede aynı frekanslar seviyesi farklı hücrelerde tekrarlanmakta ve aynı frekans daha fazla sayıdaki GSM servisi için kullanılmaktadır. Eş zamanlı birçok görüşmenin yapıldığı yoğun bölgelerde bu hücreler daha küçük ve daha sık yapıdadır.



Şekil 2. 15 GSM hücresel yapısı

2.3.1.5 GSM'de Erişim ve Frekans Kanalları

GSM, havada arabirim olarak TDMA ve FDMA'nın (Frequency Division Multiple Access) kombinasyonunu kullanmaktadır. GSM900 bandında, tahsislenen 25 MHz'lik iletişim

bandı, FDMA tekniği yardımyla 200 kHz'lik kanallar şeklinde, toplamda 124 taşıyıcı frekansı ile kullanılmaktadır. Bu taşıyıcı frekanslardan bir veya daha fazlası ana istasyonlar tarafından kullanılması için ayrılmıştır. Her bir taşıyıcı frekans, TDMA kullanılarak alt bölümlere ayrılmaktadır. TDMA'nın ana bileşeni, 0,577 ms'lik veri yayılım periyodudur (time-slot) [32]. 8 yayılım periyodu bir çerçeve içinde gruplandırılmakta ve yaklaşık 4,615 ms'lik bir TDMA çerçevesi elde edilmektedir. Bu çerçeve formatı mantıksal kanalların temelini oluşturmaktadır.

Bu kanallar fiziksel olarak TDMA çerçevesindeki bir yayılım periyoduna karşılık gelmektedir. Kanalı tanımlamak için yayılım periyoduna karşılık gelen pozisyon ve sayı kullanılmaktadır. Bu tanımlama işlemleri her 3 saatte bir yapılmaktadır.

2.3.1.6 GSM'de kimlik bilgileri

Şebeke içerisinde bir arama, aktivasyon vb. işlemlerinin gerçekleşebilmesi için çeşitli numara ve adresler kullanılmaktadır [33]. Bu bilgileri aşağıda tanımlanmıştır:

"Mobile Station ISDN NUMBER-MSISDN", bir mobil terminalin aranması için kullanılan numaradır. MSISDN uzunluğu standartlara göre prefiks hariç maksimum 15 rakamdan oluşmaktadır.

Yapısı: MSISDN = CC + NDC + SN şeklindedir.
CC = Ülke Kodu (Country Code),
NDC = Alan Kodu (National Destination Code),
SN = Abone Numarası (Subscriber Number).
Örnek: 90 532 3456789

International Mobile Subscriber Identity (IMSI), bir mobil terminalin BTS'den başlayarak tüm GSM şebekesi içerisinde, sistem tarafından tanınmasını sağlayan numaradır. Bu numara SIM kartta, HLR' de ve VLR' de tanımlanmıştır.

Yapısı: IMSI = MCC + MNC + MSIN şeklindedir.
MCC = Mobil Ülke Kodu (Mobile Country Code, 3 dijit),
MNC = Mobil Şebeke Kodu (Mobile Network Code, 2 dijit),
MSIN = Mobil Abone Numarası (Mobile Subscriber Identification Number,

Maksimum10 dijit).

Örnek: 286 01 XXXXXXXXXX

Temporary Mobile Subscriber Identity (TMSI) IMSI, kullanıcının SIM kartında mevcut kalıcı numarasıdır. TMSI ise o kullanıcının şebekeyi kullanırken güvenlik amacıyla ve IMSI nin yerine geçici olarak tahsis edilen bir kimlik numarasıdır. Bu numara yalnız o kullanıcının o anda bulunduğu MSC/VLR sahasında geçerlidir. Ayrıca kullanımı ve idaresi VLR tarafından sağlanmakta olup en fazla 8 rakam olabilir ve bir standart tanımlaması olmadığı için yapısı operatör tarafından belirlenmektedir.

IMEI, mobil terminalin tanımlanması için kullanılmaktadır.

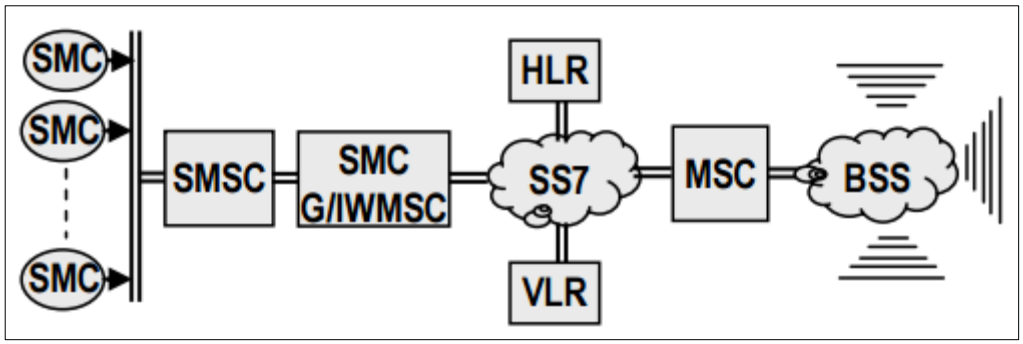
Yapısı: IMEI = TAC + FAC + SNR + sp şeklindedir.
TAC = Tip Kodu (Type Approval Code) (6 rakam),
FAC = Montaj Kodu (Final Assembly Code) (2 rakam),
SNR = Seri Numarası (Serial Number) (6 rakam),
sp = yedek (1 rakam).
Örnek: 449221 10 810166 0.
Cell Global Identity (CGI), bir yerel alan içinde hücrenin tanımlanması için kullanılmaktadır.

Yapısı: CGI = MCC + MNC + LAC + CID şeklindedir. LAI ye hücre kimlik numarası eklenmektedir.
MCC = Mobil Ülke Kodu (Mobile Country Code) (3 rakam),
MNC = Mobil Şebeke Kodu (Mobile Network Code) (2 rakam),
LAC = Yerleşim Bölgesi Kodu (Location Area Code) (Maksimum 5 rakam).
CID = Hücre Kimlik Numarası (Cell IDentity) (Maksimum 5 rakam).
Örnek : 286-01-78006-96048

2.3.2 SMS (Short Message Service)

Kısa mesaj servisi (SMS), GSM altyapısının sunmuş olduğu iki yönlü “text” verisi gönderme servisidir. Kısa mesaj servisinde, bir text mesajı maksimum 160 adet alfa numerik karakter uzunluğunda olup herhangi bir trafik kanalı kullanmadan şebeke üzerinden noktadan noktaya iletilmektedir. Abone bir SMS gönderdiğinde, bu mesaj önce ilgili şebekenin SMS merkezine (SMSC) ulaşmaktadır. SMSC, abonenin musait

olmasında mesajı alıcıya anında gönderir eğer alıcıya ulaşılamıyorsa, sonra iletmek üzere sıraya koymaktadır [30]. SMS'in GSM şebekesi üzerindeki iletim şeması Şekil 2.16'da gösterilmiştir.



Şekil 2. 16 GSM şebekesi üzerinde SMS iletimi blok şeması

Mobil telefonundan gönderilen kısa mesaj, önce numaranın bağlı olduğu kısa mesaj servis merkezine (SMSC) gider. Kısa mesaj ağ geçidi (SMSGateaway), SMSC"den mesajı alır, mesajın yönlendirilmesi için gerekli bilgileri Merkezi Konum Kaydı (HLR)'dan edinerek mesajı, alıcının o anda bağlantıda olduğu Mobil anahtarlama merkezi (MSC)'ye gönderir. MSC; mesajı, alıcının bağlı olduğu baz istasyonuna yönlendirir. Baz istasyonundan da mesaj alıcıya ulaştırılır

Genel olarak, bir GSM modemden mobil telefona kısa mesaj göndermenin iki yolu vardır. Birincisi, bilgisayar ile GSM/GPRS modem arasında bir bağlantı kurulması sonrasında, bu bağlantı üzerinden AT komutlarını (ATtention Command) kullanarak, diğer yol ise SMS Merkezi veya SMS Servis Sağlayıcısına bağlanarak, belirli olan protokol üzerinden(İnternete bağlanarak) mesaj gönderilmesidir [35]. Projede, AT komutları ile SMS gönderimi sağlanmış olup detayları Bölüm 4'de yer almaktadır.

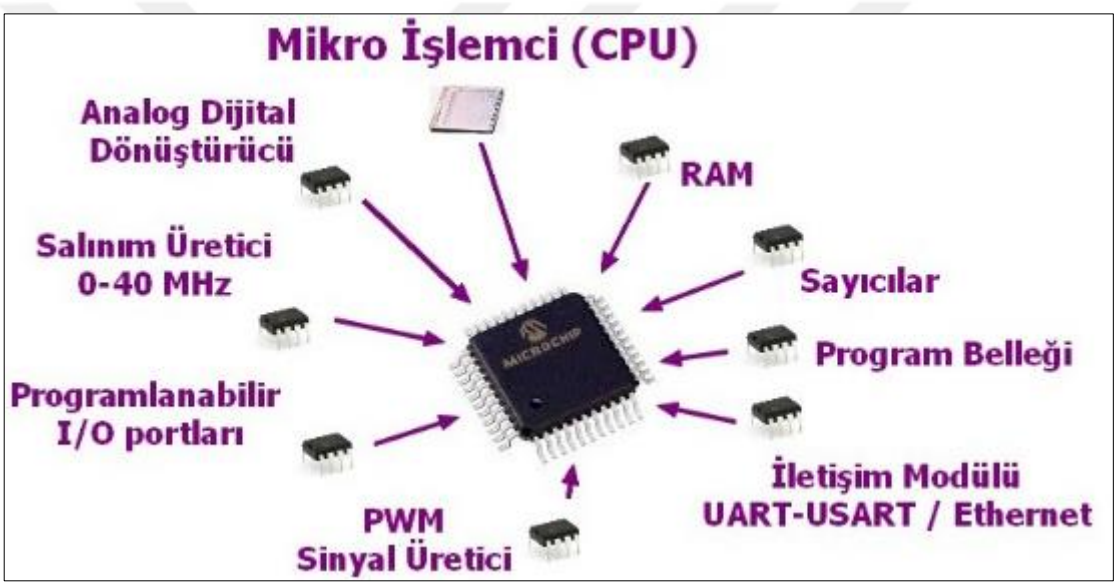
2.4 Sistemin Kontrolü ve Yönetimi

Kaza tespit ve gösterim sisteminin yönetimi mikrodenetleyici ile sağlanmıştır. Mikrodenetleyicilerin düşük maliyetli, küçük ve teknolojinin gelişmesiyle yüksek performanslı olmasında dolayı bu projede kullanılmıştır.

Mikrodenetleyiciler tek bir silikon yonga üstünde birleştirilmiş bir mikroişlemci, veri ve program belleği, sayısal giriş ve çıkışları, analojik girişler ve daha fazla güç veren ve işlev

katan diğer çevre birimleri (zamanlayıcılar, sayaçlar, kesiciler, analog-dijital çeviriciler), barındıran mikrobilgisayarlardır.

Bir mikrodenetleyici, bilgisayarın (MİB, hafıza ve giriş-çıkışlar) bir bütünleşik devre üzerinde üretilmiş olarak tanımlanmaktadır. Sınırlı miktarda olmakla birlikte yeterli sayıda hafıza birimlerine ve giriş–çıkış uçlarına sahip olmaları sayesinde tek başlarına çalışabildikleri gibi donanımı oluşturan diğer elektronik devrelerle birlikte işlem yapabilmektedirler. Mikrodenetleyicilerin programlanabilir yapıda olması, kullanıcıya tasarım geliştirme imkanı sunmaktadır. Analog-dijital çevirici gibi tümleşik devrelere sahip olmaları sayesinde algılayıcılardan veya harici sinyal kaynaklarından veri toplanması ve işlenmesi projelerinde kullanılabilmektedirler [36].



Şekil 2. 17 Mikrodenetleyici ve birimleri

Kaza tespit ve gösterim sisteminin kaza tespit algoritması, ivme algılayıcısından gelen 3 eksendeki ivme değerlerinin mikrodenetleyici tarafından işlenmesine dayanmaktadır. Bildirim algoritması ise küresel konumlandırma alıcısı üzerinden anlık olarak alınan konum bilgisinin mikrodenetleyici üzerinde saklanması ve konum bilgisinin SMS ile gönderilmesi olarak belirlenmiştir.

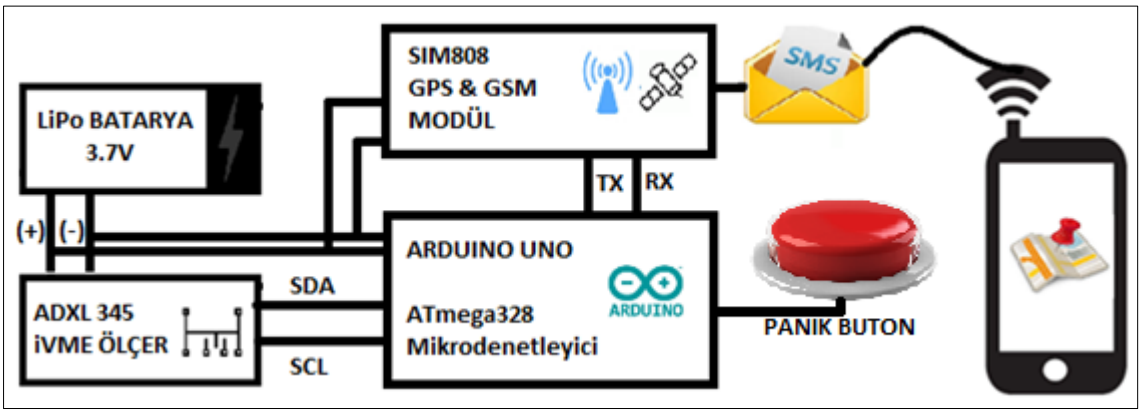
Piyasada çok fazla sayı ve özellikler de mikrodenetleyici bulunmaktadır. Bu durum itibariyle mikrodenetleyici seçimlerini belirleyen en temel faktör fiyat performans kriterinin optimum olmasıdır. Projede kullanılan mikrodenetleyici de bu kriter ile belirlenmiş olup, özellikleri hakkındaki bilgilere üçüncü bölümde yer verilmiştir.

BÖLÜM 3

KAZA TESPİT VE KAZA YERİ BİLDİRİM SİSTEMİ DONANIMI

3.1 Giriş

Kaza tespit ve gösterim sisteminin donanımı, üzerinde mikrodenetleyici bulunan donanım geliştirme kartı, ivme algılayıcısı ve konumlandırma birimine sahip haberleşme kartından oluşmaktadır. Donanımda yer alan ivme algılayıcında elde edilen veriler ile kazanın tespit edilmesi, tümleşik yapıdaki konumlandırma ve haberleşme modül ile konum bilgisinin SMS yoluyla gönderilmesi sağlanmıştır. Ayrıca, sistemde güvenlik ve sağlık açısında oluşabilecek güç durumlarda konum bilgisinin gönderilmesi amacıyla panik buton yer almakta olup sisteme ait blok diyagram Şekil 3.1'de gösterilmiştir.



Şekil 3. 1 Kaza Donanıma ait blok diyagram

3.2 Denetleme ve Kontrol Sistemi Donanımı

Kaza tespit ve gösterim sisteminin denetleme ve kontrolü mekanizması mikrodenetleyici ile sağlanmıştır. Bu kapsamda, projede Atmega328

mikrodenetleyicisine sahip Arduino Uno mikrodenetleyici donanım geliştirme kartı kullanılmıştır. Bu kartın üzerinde 14 adet dijital giriş/çıkış ucu (6 adedi PWM özellikli), 6 analog giriş, 16Mhz kristal, reset tuşu, USB soketi, ICSP konektörü ve güç girişi bulundurmakta olup, mikrodenetleyicinin çalışmasında gerekli olan devre elemanları kart üzerinde yer almaktadır. Arduino Uno mikrodenetleyici donanım geliştirme kartının teknik özellikleri Çizelge 3.1'de gösterilmiştir.

Çizelge 3. 1 Arduino Uno Donanım Geliştirme Kartının Teknik Özellikleri

Mikrodenetleyici	ATmega328
Çalışma Gerilimi	5 V
Giriş Gerilimi (önerilen)	7-12 V
Giriş Gerilimi (limit)	6-20 V
Dijital I/O Uçları	14 (6 adedi PWM çıkışı) adet
Analog Giriş Uçları	6 adedi
Her I/O için Akım	40 mA
3.3V Çıkış için Akım	50 mA
Flash Hafıza	32 KB (ATmega328) 0.5 Kb kadarı bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Saat Hızı	16 MHz
Uzunluk	68.6 mm
Genişlik	53.4 mm
Ağırlık	25 g

Mikrodenetleyicili donanım geliştirme kartının detaylı uç bağlantı noktaları EK-A'da sunulmuş olup kartının özelliklerine ilişkin bilgiler aşağıdaki başlıklar altında verilmiştir.

- **Besleme(Güç):** Arduino Uno'nun çalışması gerekli olan besleme USB port üzerinden veya harici güç kaynağından sağlanmaktadır. Harici güç kaynağı olarak AC-DC adaptör kullanılabileceği gibi DC batarya da kullanılabiliniz. Kart üzerinde adaptör

kullanımı için kart üzerindeki 2.1mm merkez-pozitif güç soketi bulunmaktadır. Batarya ise kart üzerindeki GND ve V_{in} uçlarına bağlanması gerekmektedir.

Harici güç kaynağı kullanımı için besleme voltajı 6-20V arası olmalıdır; ancak bu değerler limit değerler olup, tavsiye edilen harici besleme voltajı 7-12V arasıdır. Çünkü kart üzerinde bir adet regülatör bulunmaktadır bu regülatör 7V altındaki değerlerde stabil çalışmamaktadır ve ayrıca 12V üstündeki değerlerde de aşırı ısınmaktadır.

Uno kartının üzerindeki mikrodenetleyicinin çalışma gerilim 5V'dur. V_{in} ucu veya güç soketi üzerinden verilen 7-12V arası gerilim kart üzerinde bulunan voltaj regülatörü ile 5V'a düşürülerek mikrodenetleyici beslenmektedir.

Güç uçları aşağıdaki gibidir:

- **V_{IN}:** Harici güç kaynağı kullanılırken 7-12V arası gerilim veren giriş ucudur.
- **5V:** Kart üzerinde yer alan regülatörden çıkan 5V değerindeki gerilim çıkışıdır. Eğer kart sadece USB port (5V) üzerinden çalışıyorsa USB port üzerinden gelen 5V doğrudan bu uç üzerine verilmiştir. Eğer kart harici kaynaktan beslenmekteyse, 5V doğrudan bu uç üzerinden çıkış yapmaktadır.
- **3V3:** Kart üzerinde 3.3V regülatörün çıkış ucu mevcuttur. Bu uç en fazla 50mA çıkış verebilmektedir.
- **GND:** Toprak ucudur.
- **Hafıza:** Kart üzerinde yer alan mikrodenetleyici Atmega328, 32 KB'lık flash belleğe sahiptir (0.5 KB kadarı BOOTLOADER tarafından kullanılmaktadır.). Ayrıca, 2 KB SRAM ve 1 KB EEPROM'u bulunmaktadır.
- **Giriş ve Çıkış:** Arduino Uno üzerinde mevcut 14 adet dijital ucunun hepsi giriş veya çıkış olarak kullanılmaktadır. 6 adet analog giriş ucu de ayrıca yer almaktadır. Bu analog giriş uçları dijital giriş ve çıkış olarak da kullanılabilmektedir. Toplamda 20 adet dijital giriş çıkış ucu bulunmaktadır. Bu uçların tamamı sayısal 5V seviyesindedir. Her uç maksimum 40mA giriş ve çıkış akımı ile çalışmaktadır; Ek olarak, bazı uçların farklı özellikleri mevcut olup özellikli uçlara ait bilgilere aşağıda yer verilmiştir.
- **Seri Haberleşme(Pin0 (RX) ve Pin1 (TX)):** Seri olarak veri alıp (RX), vermek (TX) için kullanılmaktadır. Bu uçlar doğrudan kart üzerinde mevcut Atmega16u2 USB-seri

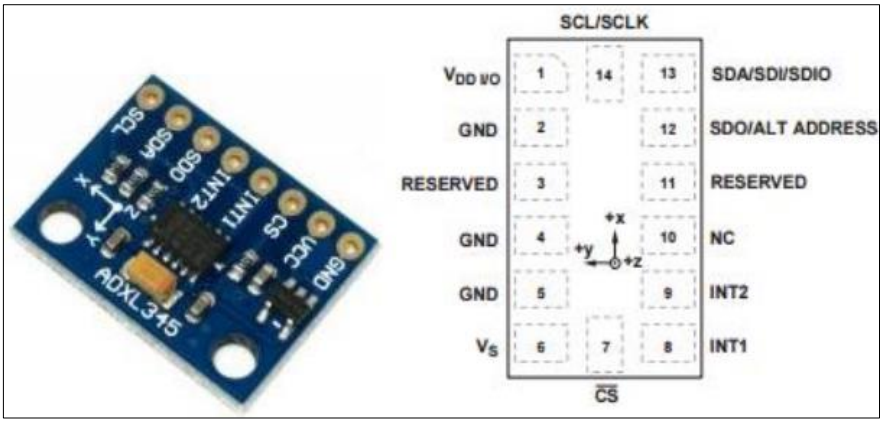
dönüştürücüsüne bağlıdır. Bir başka değişle, bilgisayar ile kart arasındaki veri haberleşmesinde de bu uçlar kullanılmaktadır. Yazılım yükleme işlemi de bu veri hattı üzerinden gerçekleştirilmektedir.

- **Harici Kesme(Pin2 (interrupt 0) ve Pin3 (interrupt 1)):** Bu uçlar yükselen kenar, düşen kenar veya değişiklik kesmesi uçları olarak kullanılmaktadır.
- **PWM(Pin3,5,6,9,10,11):** 8-bit çözünürlükteki PWM çıkış uçları olarak kullanılmaktadır.
- **SPI (Pin10 (SS)), Pin11 (MOSI), Pin12 (MISO), Pin13 (SCK)):** Bu uçlar SPI haberleşmesi için kullanılmaktadır.
- **LED (Pin13):** Kart üzerinde bu uca bağlı bir led bulunmaktadır. Uç HIGH yapıldığında led yanmakta olup, LOW yapıldığında led sönmektedir.
- **Analog (PinA0-A5):** 6 adet 10-bit çözünürlüğünde analog giriş ucu mevcuttur. Bu uçlar dijital giriş ve çıkış içinde kullanılmaktadır. Uçların voltaj aralığı 0-5V'dur. Kart üzerinde yer alan AREF ucu ve "analogReference()" fonksiyonu kullanılarak alt limit yükseltilip, üst limit düşürülmektedir.
- **I^2C (PinA4-SDA, PinA5-SCL):** Bu uçlar I^2C haberleşmesi için kullanılmaktadır.
- **AREF**: Analog giriş için referans ucudur.
- **Reset**: Uç LOW yapıldığından mikrodenetleyici resetlenmektedir**.**

3.3 Algılayıcı Donanımı

Tasarlanan sistemde ivme algılayıcısı olarak "Analog Devices" firmasının ADXL345 isimli algılayıcısı kullanılmıştır. ADXL345 küçük, düşük güç tüketimine sahip 3 eksende çıkış verebilen bir açısal ivmeölçer algılayıcıdır. Algılayıcı ±2g, ±4g, ±8g, ±16g'lik ivme araklılarında isteğe bağlı olarak I^2C ve SPI hattı üzerinden ayarlanabilir ivme bilgisi çıkısı yapılabilmektedir [19].

ADXL345 ivmeölçer, Şekil 3.1'de gösterilen bir tümleşik devre kartı üzerine monte edilmiştir. Proje kullanılan bu algılayıcı kart, küçük ve ince, oldukça düşük güç sarfiyatına sahip yüksek çözünürlüklü bir 3-eksenli ivmeölçer özelliğine sahiptir. Algılayıcı tümleşik devresi, 16-bit SPI (3 ya da 4 tel ile) ya da I^2C hattı üzerinden dijital çıkış verisi üretmektedir.



Şekil 3. 2 ADXL345 uç konfigürasyonu ve fiziksel görünümü

Teknik özelliklerinin yanı sıra ADXL345'in mobil aygıt uygulamaları için çok uygun olması sebebiyle projede kullanılmıştır. Algılayıcı sarsıntı ölçme uygulamalarında yer çekimi ivmesini ölçebilirken, hareket veya şok kaynaklı dinamik ivmeleri de ölçebilmektedir. Algılayıcının yüksek çözünürlüğü (3.9 mg/LSB) sayesinde 1.0^{o}'den düşük eğimler dahi hissedilmektedir.

Algılayıcının farklı saptama fonksiyonları bulunmaktadır. Aktivite inaktivite saptama fonksiyonu kullanıcı tarafından seçilen belirli eşiklerle karşılaştırma yapılarak aktivite olup olmadığına karar verir. Algılayıcı, her yönde tek ve çift tıklamaları da saptayabilmektedir ve ek olarak serbest düşme fonksiyonu da aletin düşüp düşmediğini saptamak için kullanılabilir. Bu fonksiyonların her biri modülün iki kesme çıkışından her hangi birine bağlanabilmektedir.

ADXL 345 İvme algılayıcınsın Genel Özellikleri;

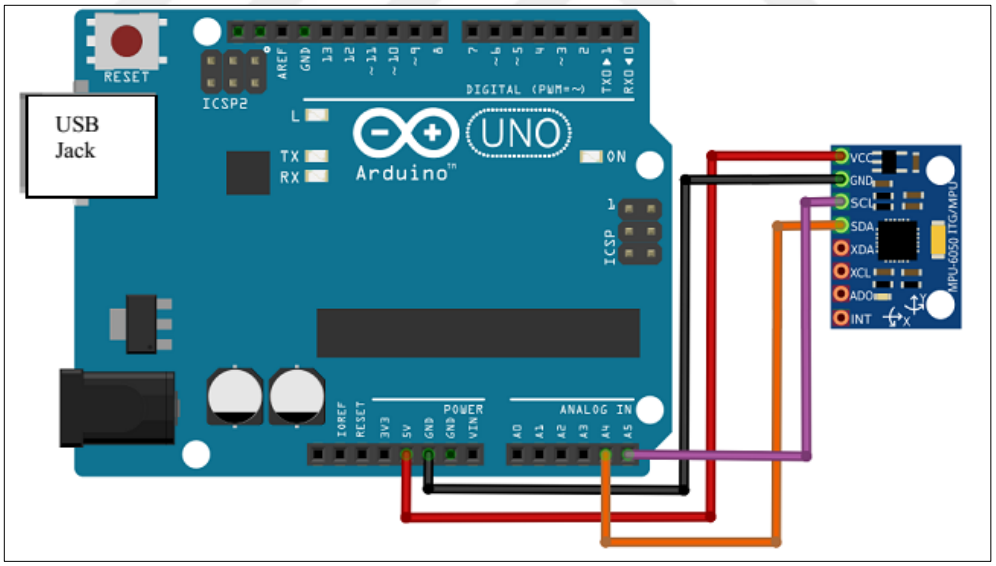
- Tek/çift tıklama saptama,
- Aktivite/inaktivite saptama,
- Serbest düşme saptama,
- 10,000g şok dayanıklılık,
- SPI ve I^2C dijital haberleşme ara yüzü,
- Seri komutlarla ayarlanabilir ölçüm menzili,
- Seri komutlarla seçilebilir band genişliği,

ADXL 345 ivme algılayıcınsın teknik özellikleri;

- Çalışma Voltajı: 4.0V – 6.0V,
- I/O Voltaj Aralığı: 1.7V – 3.6V,
- Haberleşme: SPI ve I^2C,
- Çalışma Sıcaklığı: -40^oC - 85^oC,
- Boyut: 3mm x 5mm x 1mm.

Algılayıcı üzerinde dijital veri iletimi için iki farklı iletişim protokolü bulunur. Bu protokoller elektronik haberleşmede Serial Periperhal Interface (SPI) ve Inter Integrated Circuit (I^2C) olarak adlandırılır. Kaza tespit ve bildirim sisteminde I^2C isimli protokol kullanılmıştır. Bu konuda detaylı bilgi Bölüm 3.4.3'teverilmiştir.

ADXL 345 ivme algılayıcısı projede kaza tespiti için sarsıntıyı/vuruntuyu ve araçların devrilmesi halinde eğim açılarını tespit ve hesaplamak amacıyla kullanılmıştır. Algılayıcı mikrodenetleyici üzerinden beslenmekte olup Şekil 3.3'de gösterildiği mikrodenetleyiciye irtibatlandırılmıştır.

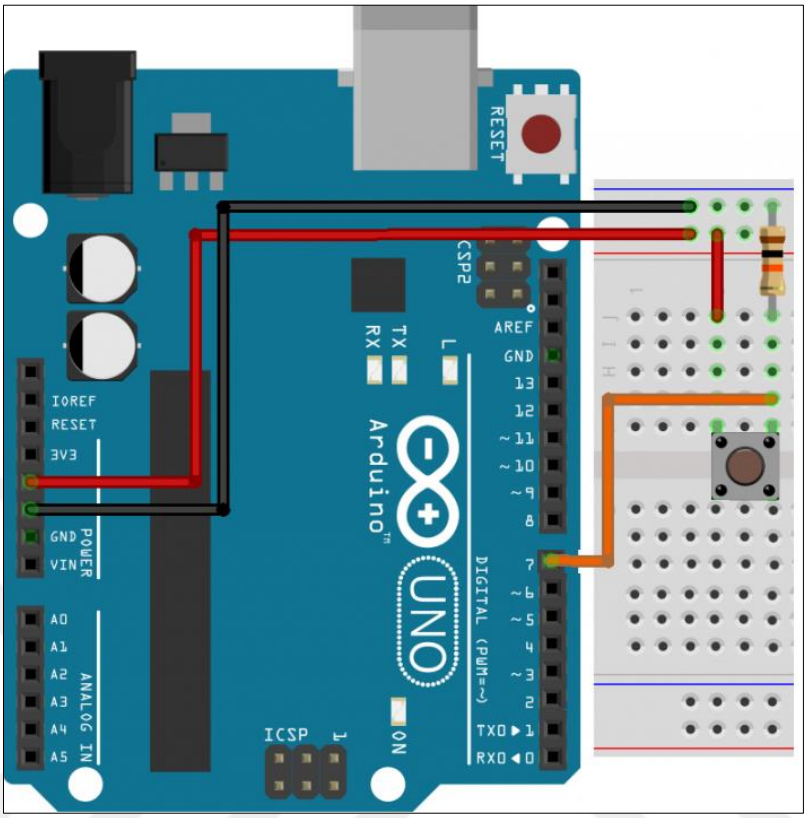


Şekil 3. 3 ADXL345 İvme Algılayıcısı ile Arduino Uno mikrodenetleyici donanım geliştirme kartının bağlantısı

3.4 Panik Buton Donanımı

Çalışmada, güvenlik ve sağlık açısında oluşabilecek güç durumlarda konum bilgisinin gönderilmesi maksadıyla panik buton kullanılmıştır.

Panik buton için 2 uçlu “push-buton” kullanılmış olup, Şekil 3.4’de gösterildiği şekilde Arduino Uno mikrodenetleyici donanım geliştirme kartına bağlanmıştır.



Şekil 3. 4 “Push-Buton“ donanımın Arduino Uno mikrodenetleyici geliştirme kartına bağlantısı

Buton, 10 kΩ “pull-down” direnci ile bağlanmıştır. “Pull-down direnci, dijital uçları giriş olarak kullandığımızda sinyalin bozulmamasını sağlamaktadır. Projede buton basılı değilken dijital uç üzerinden okunan değer 0V yani lojik LOW seviyesidir. “Pull-down” direnci, buton basılıp değer HIGH’a çekilmediği sürece bu uçtaki gerilimin 0V’ta sabit kalmasını sağlamaktadır.

3.5 Konumlandırma ve Haberleşme Donanımı

Bildirim sistemin konumlandırma ve haberleşme donanımı “SIMCom” firmasının SIM808 entegresi barındıran tümleşik bir devre kartından oluşmaktadır. SIM808 modülü uydu navigasyonu için GPS teknolojisini birleştiren eksiksiz bir Quad-Band GSM / GPRS modülüdür [37]. Yüzey montaj teknolojisi (Surface Mount Technology-SMT) ile GSM ve GPS altyapısını barındıran kompakt tasarıma sahip olmasından ve AT

komutlarıyla mikrodenetleyici tarafından kontrol edilebilmesinden dolayı projede SIM808 tercih edilmiştir.

SIM 808'un GPS Özellikleri:

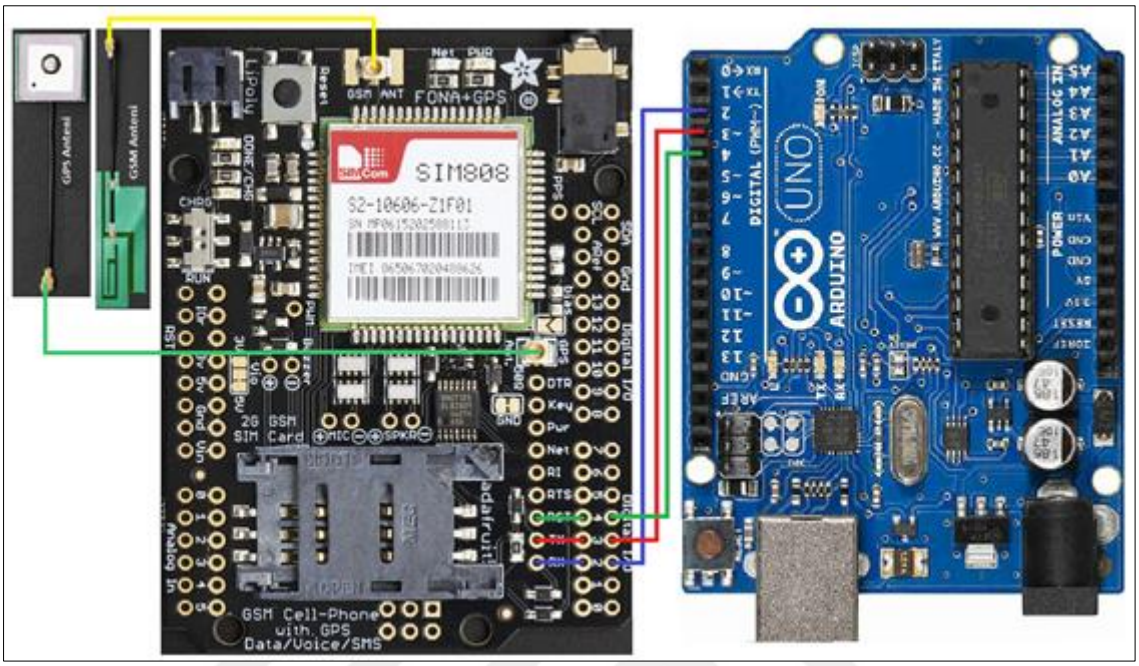
- 22 takip/66 algılama kanal,
- GPS L1 C/A kod,
- Hassasiyet: İzleme(tracking) -165dBm, soğuk başlangıç -148dBm,
- Konum bulma süresi(tipiksel): Soğuk başlangıç 32sn, sıcak başlangıç 1sn, ılık başlangıç 5sn
- Doğruluk: ~2.5m

SIM 808'un GSM Özellikleri:

- Quad-band850/900/1800/1900MHz desteği, dünya çapında tüm 2G şebekeler ile uyumluluk,
- Dahili MT3337 GPS alıcısı, -165dBm hassasiyet, aynı seri port üzerinden kontrol imkanı,
- Kart üzerindeki kulaklık/mikrofon çıkışı veya harici 32 ohm hoparlör + elektret mikrofon ile sesli görüşme desteği,
- SMS gönderip alabilme özelliği,
- GPRS veri gönderip alabilme özelliği (TCP/IP, HTTP vb.),
- PWM/Buzzer titreşim motoru kontrolü,
- AT komut satırı destekli, otomatik baud rate ayarlı UART haberleşme,

SIM 808 entegresi, Arduino Uno mikrodenetleyici donanım geliştirme kartına uyumlu olan bir devre kartı (Adafruit FONA 808 Shield) üzerine monte edilmiştir. Bu kart üzerindeki konektörler ile kontrol kartına ilave olarak konumlandırma ve şebeke anteni takılmaktadır. Modül üzerinde ayrıca JST konektör bağlantılı güç devresi mevcuttur. Bu konektör sayesinde batarya üzerinde sistem beslenmekte olup kartın sahip olduğu şarj devresi ile batarya şarj da edilebilmektedir.

SIM 808 tümleşik devre kartı, projede kaza konumun belirlenmesi ve konumun bildirilmesi amacıyla kullanılmıştır. Devre batarya üzerinden beslenmekte olup UART haberleşme hattı ile Şekil 3.5'te gösterildiği mikrodenetleyiciye irtibatlandırılmıştır.



Şekil 3. 5 İvmeölçerin SIM 808 tümleşik GSM/GPS devre kartının Arduino Uno mikrodenetleyici donanım geliştirme kartıyla bağlantısı

3.6 Antenler

Konumlandırma uydularıyla ve GSM şebekesiyle bağlantı kurmak için SIM 808 tümleşik devre kartına bağlı 2 adet anten kullanılmıştır.

3.6.1 Şebeke Anteni

Projede SIM808 tümleşik devre kartına uyumlu olması maksadıyla u.FL tipi konektöre sahip 850/900/1800/1900/2100 MHz frekanslar ile uyumlu Şekil 3.6'da gösterilmiş anten kullanılmıştır. Arkası yapışkanlı olan 2 mm kalınlıktaki anten, yüzeye kolayca sabitleyebilmekte olup 3dB kazanca sahiptir.

3.6.2 Konumlandırma Anteni

Konumlandırma anteni, GPS uydularıyla haberleşme amacıyla SIM808 tümleşik devre kartına bağlıdır. Anten olarak u.FL konektör yapısına sahip, hafif ve küçük olan

Şekil 3.6’de yer alan anten tercih edilmiştir. Antenin 1 dB kazanca sahip olup pasif yapıdadır.



Şekil 3. 6 GPS ve GSM Antenlerinin Fiziksel Görünümü

BÖLÜM 4

KAZA TESPİT VE KAZA YERİ BİLDİRİM SİSTEMİ YAZILIMI

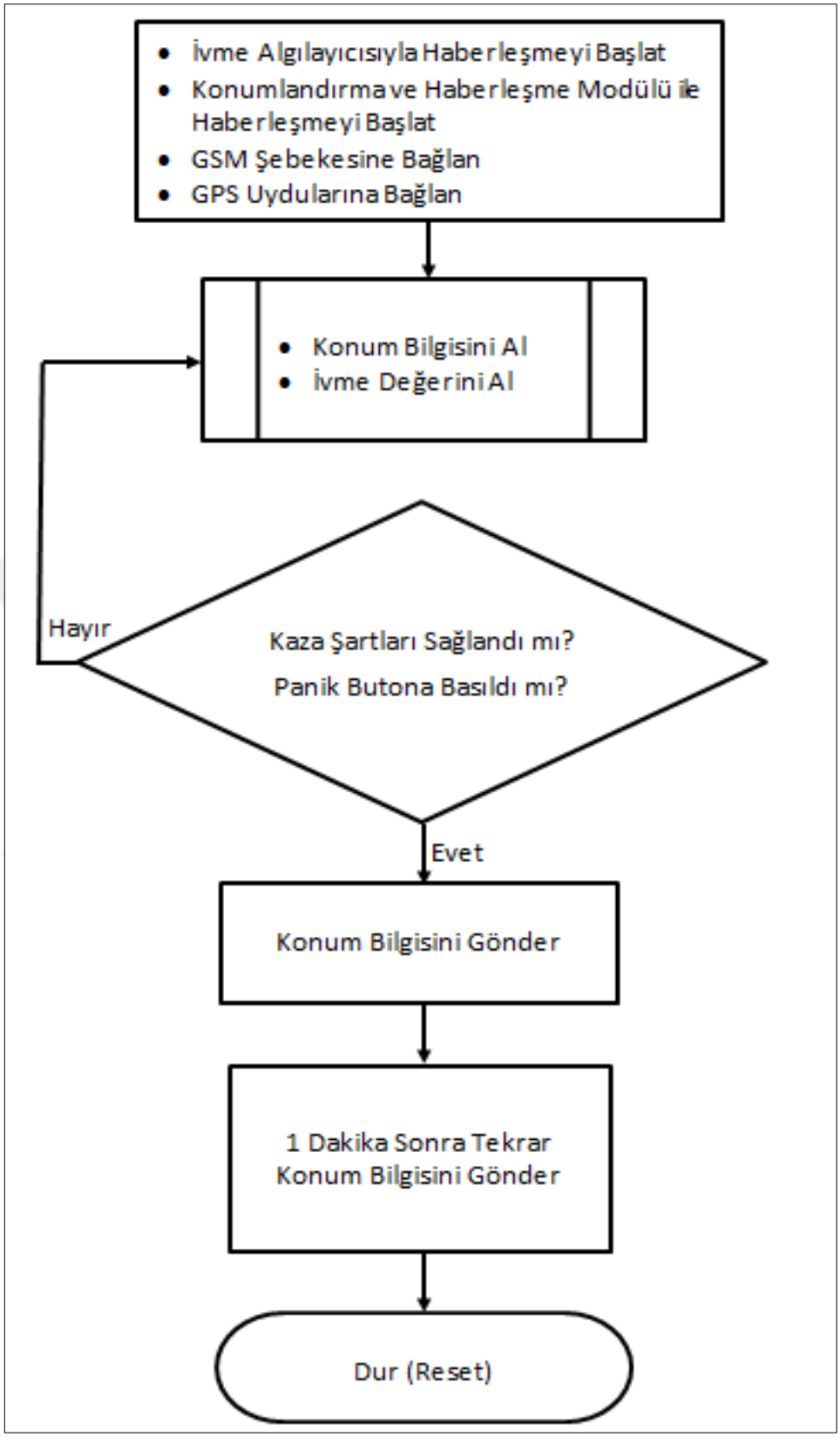
4.1 Giriş

Kaza tespit ve gösterim sisteminin yazılımı, ana kontrol yazılımıyla ivme algılayıcısı (ADXL345) ve konumlandırma ve haberleşme modülü (SIM808) kütüphanelerinden oluşmaktadır. Yazılıma ait akış diyagram Şekil 4.1'de belirtilmiş olup, EK-B'de yazılımların detaylarına yer verilmiştir.

Yazılım ile öncelilkli olarak ivme algılayıcısı ve konumlandırma/haberleşme modülü iletişimin kurulması gerçekleştirilmiş olup, GSM şebekesine ve GPS uydularına erişim sağlanmıştır. Gerçek zamanlı olarak konum ve ivme değerleri yazılım ile mikrodenetleyiciyle aktarılmıştır. Kaza tespit algoritması, yazılım aracıyla mikrodenetleyiciye kontrol ettrilmiş ve son olarak kaza tepiti sonrasında SMS ile konum bildirimi sağlanmıştır. İlave olarak panik ikazını kontrolü ve bildirimi yazılım ile gerçekleştirilmiştir.

Sisteme ait tüm yazılım Arduino Uno mikrodenetleyici donanım geliştirme kartı üzerinde yer alan ATmega328 mikrodenetleyicisi yer almaktadır.

Atmega328, 0 (RX) ve 1 (TX) uçları üzerinden UART TTL (5V) seri haberleşme imkanı sunmaktadır. Kart üzerinde bulunan Atmega16u2 USB port-seri dönüştürücü vasıtasıyla bilgisayarda sanal bir haberleşme portu oluşturmaktadır. Bu port ile Atmega328 ile bilgisayar arasında bir köprü kurulmaktadır. Bu köprü üzerinden sisteme ait tasarlanan yazılım, açık kaynaklı "Arduino IDE" programında derlenerek mikrodenetleyiciye yüklenmektedir.



Şekil 4. 1 Kaza tespit ve gösterim sistemi yazılımına ait akış şeması

4.2 Mikrodenetleyici Yazılımı

Arduino firmasına ait mikrodenetleyici kartları firmanın resmi internet sitesinde yer alan bir ara yüz programı aracılığıyla programlanmaktadır. Bu program hem derleyici hem de kod yükleyici olarak kullanılmaktadır. Kullanıcı ara yüz programında programlama dili olarak mikrodenetleyiciler için özel fonksiyonlar barındıran basitleştirilmiş bir C++ dili kullanılmaktadır. Bu ek fonksiyonlar sayesinde mikrodenetleyiciye ait birçok işlem tek satırlık kodlarla tanımlanmaktadır[38]. Yapılan çalışmada kullanılan ek fonksiyonların isimleri ve görevleri aşağıda verilmiştir.

#include: Harici işlemlerin yapıldığı kütüphane dosyasının tanımlanması için kullanılır.
#define: Mikrodenetleyiciye ait uçların tanımlanması için kullanılır.
pinMode(): Uçları giriş veya çıkış olarak yapılandırma işlemi yapar.
digitalRead():Belirtilen dijital uç değerini okur.
delay(): Belli bir milisaniye için kodun çalışmasını durdurur.
memcpy(): Array kopyalama için kullanılır.
atof(): String değişkeni Float değişkene çevirmek için kullanılır.
strtok(): Varolan dizgiyi dizgiciklere ayırmak için kullanılır.
snprintf(): Array birleştirmek için kullanılır.
dtostrf(): Tam sayıyı, ondalıklı sayıya dönüştürmek için kulanır.
sqrt(): Karekök işlemi için kullanılır.
pow(): Bir sayının karesini almak için kullanılır.
Abs(): Mutlak değer işlemi için kullanılır.

Mikrodenetleyici kodu “setup” ve “loop” olmak üzere iki temel fonksiyondan meydana gelir. Mikrodenetleyici çalışmaya başlarken gerçekleştireceği işlemler “setup” fonksiyonu içinde belirtilir. Buna göre karta enerji verildikten sonra Mikrodenetleyici bir seferlik bu kısımda bulunan kodda belirtilen görevleri yerine getirir. Sonrasında ise Mikrodenetleyici “loop” fonksiyonunda bulunan koda göre çalışmaya devam eder. Burada bulunan kod ise Mikrodenetleyici aktif olduğu sürece sürekli kendini tekrarlamaktadır.

Yapılan çalışmada kullanılan ADXL 345 ve SIM808 modülleri Arduino Uno mikrodenetleyici donanım geliştirme kartının seri iletişim uçlarına bağlanmış olup,

aralarındaki haberleşme aşağıda belirtilen seri iletişime ait kodlar kullanılarak gerçekleştirilmiştir.

Serial.print() ve Serial.println(): Seri iletişim üzerinden bilgi alışverişi için standart Amerikan kodu (ASCII) yapısında veri iletir.

Wire.begin(): I^2C protokolü üzerinden iletişimi başlatır (ADXL345 ile haberleşme).

Serial.begin(): Seri iletişimi başlatır. Bilgi akış hızı (baud rate) burada tanımlanır (SIM808 modülü ile haberleşme).

Önceden de belirtildiği gibi Arduino Uno mikrodenetleyici donanım geliştirme kartlarının farklı algılayıcılar veya cihazlar ile çalışmasında kolaylık sağlayan çeşitli kütüphane dosyaları mevcuttur. Projede ADXL345 ivme algılayıcısı SIM808 konumlandırma ve haberleşme ile haberleşmede “fona” ve “Wire” isimli iki ayrı kütüphane dosyası kullanılmıştır.

“ADXL345” isimli kütüphane dosyası aracılığıyla herhangi bir ek kontrolör kullanmadan I^2C haberleşmesi ile ivme algılayıcısından 3 ayrı eksendeki ivme değerleri alınmaktadır. Gelen ivme değerlerinin kıymetlendirilmesi ana kontrol yazılımıyla sağlanmaktadır. “Adafruit_FONA” isimli kütüphane dosyası üzerinden konumlandırma ve haberleşme modülü AT komutlarıyla otomatik baud rate ayarlı UART seri haberleşmesi ile haberleşme konum bilgisi alınmakta ve SMS gönderimi sağlanmaktadır.

AT komut seti, "Hayes" adında bir telekomünikasyon firması tarafından oluşturulmuş bir standarttır. İsmi, her komutun "AT" karakterleri ile başlamasından gelmektedir. GSM için kullanıldığı gibi faks makineleri, modemler vs. cihazlar için de kullanılan bir standarttır. Bir cep telefonunda rehberden bir kişiyi seçilip, arama tuşuna basıldığında; telefonun işlemcisi, GSM modeme seri porttan ATD05xxxxxxxxx; gibi bir kod yollanmasıyla arama gerçekleşmektedir. GSM için en önemli AT komutları; SMS gönderme, gelen SMS’e işlem yapma, arama yapma, gelen aramaya cevap verme, GPRS veya 3G üzerinden internete girip veri alıp gönderme işlemleri için kullanılandır. GSM modemler, standart AT komutları ile birlikte genişletilmiş AT komutlarını da desteklemektedir. Genişletilmiş AT komutları ile sinyal gücünü tespit etme, enerji tasarrufu sağlama, uçak moduna geçiş gibi birçok işlem gerçekleştirilmektedir. AT komutları için birkaç temel örnek Çizelge 4.1’de verilmiştir. [39]

Çizelge 4. 1 AT Komutları

AT+CBC	Bataya durumu
AT+CGMI	Üretici firma
AT+CGMM	Model
AT+CPIN?	Sim kart modeli
AT+CREG?	Şebeke kontrolü
ATD050XXXXXXXX	Arama yapma
ATH	Aramayı cevaplama
ATA	Gelen aramayı cevaplama
AT+CMGF=1	Gelen ya da okonan SMS'i text olarak formatlama
AT+CMGS="+901234567890" R"+"text"+"r"	Belirtilen numraya text yazısını SMS olarak gönderme
AT+CMGR=1	Gelen mesajı okuma
AT+CGSN:	IMEI sorulama
AT+CPIN="pin"	Pin kodu girme
AT+CPMS?	Kaç SMS geldiğini öğrenme
AT+CMGL="ALL"	Tüm SMS'leri sil

4.3 Ana Kontrol Yazılım

Ana kontrol yazılımı ivme algılayıcısı ve konumlandırma/haberleşme modülü ile seri haberleşmeyi, kaza değerlendirmesini, kazayı tespit ettiğinde konumun SMS ile gönderilmesini sağlamaktadır. Yazılımı oluşturan unsurlar alt başlıklarda anlatılmıştır.

4.3.1 Tanımlanma

Yazılımda, öncelikli olarak kullanılan haberleşmelere ve mikrodenetleyiciye bağlı modüllere ait kütüphaneler tanıtılmıştır (Şekil 4.2).

```
#include "ADXL345.h"//ADXL345 sensörü kütüphanesi
#include <Wire.h> //I2C haberleşme kütüphanesi
#include "Adafruit_FONA.h"//Konumlandırma ve haberleşme modülü kütüphanesi
#include <SoftwareSerial.h>//Seri Port Haberleşme Kütüphanesi
#include <EEPROM.h> //EEPROM Kütüphanesi
```

Şekil 4. 2 Kütüphane tanımlaması

Tanımlama bölümünde ayrıca, kullanılacak değişkenler tanımlanmıştır. (Şekil 4.3)

```
double Bileske_ivme,  Max_bileske_ivme, yatayegim, Max_yatayegim;//İvme algılayıcısı için değişkenlerin tanımlanması
int SMSgonderme=0; // Kaza şartının oluşumunu anahtarlayan değişken
int acildurum=0;// Kaza veya Acil Durumu anahtarlayan değişken
char numara[]="05053857887";//tanımlanmış numara
char mesaj[140];//mesaj metni için değişken ataması
char enlem[100];//Enlem için değişken tanımlaması
char boylam[100];//Boylam için değişken tanımlaması
char m1[]="XXX YY QQQ plaka numarali araciniz kaza yapti... Konum     ";//mesaj metni tanımlaması
char m2[]="XXX YY QQQ plaka numarali araciniz icin acil durum...Konum     ";//mesaj metni tanımlaması
char m3[]="https://www.google.com.tr/maps/@";//akıllı telefonlarda konum oluşturma linki
int butonPin=7;//Panik Buton için pin tanımlaması
int butonDurum=0; //Panik Buton durum tanımlaması
char gpsverisi[120];//GPS verisi için tanımlanan array
```

Şekil 4. 3 Değişkenlerin tanımlaması

4.3.2 Kurulum

Yazılım içerisinde bir kez uygulanması gereken fonksiyonlar ve atama işlemleri “setup” bölümünde gerçekleştirilir. Bu bölüm, mikrodenetleyicinin her açılışında uygulanacaktır. Genel olarak;

- Haberleşme portlarının aktif edilmesi,
- İvme algılayıcısının kalibre edilmesi,,
- SIM kartın şebekeye bağlanması,
- Panik buton için uç ataması,

bu bölümde gerçekleştirilmiştir (Şekil 4.4).

```
void setup() {

  Serial.begin(115200);//Seri port haberleşmesini başlatma

  Wire.begin();//Sensör ile haberleşmeyi başlatma
  accel.init(-4, -4, 4);//Sensörün başlangıç değerlerinin tanımlaması için OFFSET değerleri
  accel.setSoftwareOffset(-0.01, -0.015, -0.011);//Kalibrasyon OFFSET değerleri
  accel.printCalibrationValues(40);//(Kalibrasyon 4 saniye boyunca değer alınmaktadır.)

  fonaSerial->begin(9600);//SIM808 Konumlandırma ve Haberleşme Modülü haberleşme hızı
  fona.begin(*fonaSerial);//SIM808 Konumlandırma ve Haberleşme Modülü haberlşemesini başlatma

  fona.enableGPS(true);//Konumlandırmayı aç
  char PIN[5]="2019";//PIN numarası
  fona.unlockSIM(PIN);//Haberleşme modulunu aç
  pinMode(butonPin,INPUT);//Panik buton için pin atanması
}
```

Şekil 4. 4 Kurulum işlemleri

4.3.3 İşlem

Yazılımda tüm işlemlerin belirlenen sıralamaya göre icra edilmesi "loop" fonksiyonunda sağlanmıştır. Bu bölümde kaza tespit ve gösterim işlemleri sonsuz bir döngü içerisinde gerçekleştirilmiştir. Bu sonsuz döngünün içerisinde;

- İvme değerlerinin alınması (Şekil 4.5)

```
void loop() {

AccelG accelG;//Kütüphane fonksiyonundan yeni fonksiyon tanımlanması
accelG = accel.readAccelG(); //Sensörden G biriminde ivme okunması
```

Şekil 4. 5 İvme değerini okunması

- Bileşke ivme değerinin hesaplanması (Şekil 4.6)

Bu değer Denklem (2.1) ile hesaplanmıştır.

```
Bileske_ivme = sqrt(pow(accelG.x,2)+pow(accelG.y,2)+pow(accelG.z,2));//bileşke ivme değerinin hesaplanması
```

Şekil 4. 6 Bileşke ivmenin hesaplanması

- Yatay eğim açısının hesaplanması

Eğim, Denklem 3.1'deki belirtilen anlık ivme verileri ile hesaplanmıştır (Şekil 4.7).

$$Eğim = \arctan\left(\frac{R_x}{\sqrt{{R_z}^2+{R_y}^2}}\right) \quad (3.1)$$

```
yatayegim  = (atan2(accelG.y,(sqrt(accelG.x*accelG.x+accelG.z*accelG.z))) * 180.0) / PI;
```

Şekil 4. 7 Eğimin hesaplanması

- Panik buton durumunun kontrol edilmesi (Şekil 4.8)

```
butonDurum=digitalRead(butonPin);//Panik buton değerinin okunması
```

Şekil 4. 8 Panik butonun kontrol edilmesi

- Konumun bilgisinin (enlem ve boylam) alınması (Şekil 4.9)

```
char *durumm = strtok(gpsverisi, ",");//GPS verisinden GPS haberleşmesinin yapıldığı duruma ait bilginin alınması
char *fixx = strtok(NULL, ",");//Konum atılmasına ait bilginin alınması
char *tarihh = strtok(NULL, ",");//GPS'den tarih bilgisinin alınması
char *enlemm = strtok(NULL, ",");//GPS'den enlem bilgisinin alınması
char *boylamm = strtok(NULL, ",");//GPS'den boylam bilgisini alınması
strcpy(boylam,boylamm);//Array kopyalama işlemi
strcpy(enlem,enlemm);//Array kopyalama işlemi
int fix,durum;
fix = atof(fixx);//sayısal değere çevirme
durum = atof(durumm);//sayısal değere çevirme
```

Şekil 4. 9 Konum bilgisinin alınması

- Konum bilgisinin (enlem ve boylam), EEPROM üzerine kaydedilmesi ve GPS haberleşmesi kesildiğinde EEPROM üzerinden okunması (Şekil 4.10)

```
switch (fix){//GPS haberleşmesi durumununa göre anahtarlama
case 1://Haberleşme var ise konum EEPROM'a yazılır.
   float en,boy;
   en = atof(enlem);
   boy = atof(boylam);
   EEPROM_Write(&en, 1);
   EEPROM_Write(&boy, 32);
   break;

case 0://Haberleşme yok ise konum EEPROM'dan okunur.
   EEPROM_Read(&en, 1);
   EEPROM_Read(&boy, 32);
   dtostrf(en, 2, 6, enlem);
   dtostrf(boy, 2, 6, boylam);
   break;
 }
```

Şekil 4. 10 Konum Bilgisinin kaydedilmesi alınması

- Kaza ve acil durum şartının kontrol edilmesi (Şekil 4.11)

Konum bilgisinin SMS ile gönderilmesi kapsamında iki durum sorgulanmaktadır. Bunlardan birincisi aracın 3.7G ivmeye maruz kalması, diğeri ise aracın yatay eksende 45 dereceden fazla bir eğim açısında bulunmasıdır(normal şartlar altında devrilme öncesi). Bu şartlardan bir tanesinin gerçekleşmesi durumunda konum gönderilmesi sağlanmıştır. Ayrıca acil durum bildiricisi olarak yer alan panik botuna basılması da konum gönderme şartı olarak burada tanımlanmıştır. “acildurum” değişkeni kaza ve panik durumlarını mesaj metninde ayırmak için tanımlanmıştır.

```
switch (SMSgonderme){
  case 0:
    if (Max_bileske_ivme>3.7 || Max_yatayegim>45){//Kaza durum algoritması
        SMSgonderme=1;
        acildurum=1;
      }
    else if (butonDurum==HIGH){//Acil Durum algoritması
        SMSgonderme=1;
        acildurum=2;
      }
    else{
         SMSgonderme=0;
      }
  break;
```

Şekil 4. 11 Kaza ve acil durum sorgulama

- Kısa mesaj gönderilmesi (Şekil 4.12)

Kaza ve acil durum şartlarının oluşması halinde SMS ile konumun gönderimi sağlanmıştır.

```
case 1:
  switch(acildurum){
    case 1://kaza için
      snprintf(mesaj, sizeof mesaj, "%s%s,%s %s%s,%s", m1, enlem, boylam, m3, enlem, boylam);//mesaj metninin oluşturma
    break;

    case 2://Acil Durum için
      snprintf(mesaj, sizeof mesaj, "%s%s,%s %s%s,%s", m2, enlem, boylam, m3, enlem, boylam);//mesaj metninin oluşturma
    break;
  }

  fona.sendSMS(numara, mesaj);//SIM808 kütüphanesinde tanımlı SMS gönderme fonksiyonu
  delay(60000);//1 dakikalık gecikme
break;
```

Şekil 4. 12 SMS gönderme

SMS gönderme şartlarının sağlanması sonrasında takla, sürüklenme vb. nedenler ile aracın konumunun değişebileceğinden, sms gönderme fonksiyonu sonrasında tanımlanan gecikme fonksiyonu ile konum bilgisi bir dakika aralıklarla gönderilmektedir. Bu sayede alıcının devamlı bilgilendirilmesi sağlanmıştır.

4.4 Kütüphaneler

Projede, ivme algılayıcısı ile konumlandırma/haberleşme modülünün ana kontrol yazılımıyla veri alışverişini kurma işlemini gerçekleyen iki adet kütüphane mevcuttur. Bu kütüphaneler aynı zamanda modüllerin yapılandırmayı ve modüller ile haberleşmeyi sağlamaktadır. Ana kontrol yazılımın yer alan fonksiyonlar bu kütüphanelerde yer almaktadır.

4.4.1 ADXL345 Kütüphanesi

Bu kütüphanede ADXL345 ivme algılayıcısının veri föyünde yer alan haberleşme kodları ve bu kodların C dilinde işlenmiş verileri yer almaktadır [19]. Kütüphane dosyaları EK-B'de sunulmuştur. yer alan tanımlamalar ile algılayıcı yapılandırılmıştır.

Kütüphane yazılımında;

- Algılayıcıyla seri haberleşmenin yapılması,
- Algılayıcının başlangıç pozisyonuna göre kalibrasyon ayarının yapılması,
- Algılayıcıdan 3 eksendeki ivme değerlerinin okunması,

- Algılayıcının ofset ayarının yapılarak, Kalman filtresinin[1] uygulanması,
- Algılayıcını çözünürlük değerinin ayarlanması,

gibi sistemin kaza tespit biriminde gerekli olan işlemler tanımlanmıştır.

4.4.2 SIM808 (Adafruit) Kütüphanesi

Bu kütüphanede SIM808 konumlandırma ve haberleşme modülünün veri föyünde yer alan haberleşme kodlarında yer alan AT komut seti ve bu komutların C dilinde işlenmiş verileri yer almaktadır [40]. Kütüphane dosyaları EK-B'de sunulmuştur.

Kütüphane yazılımında,

- Modüle AT komutlarının gönderilmesi ve modül üzerinden AT komutlarının alınması,
- GSM şebekesine bağlanılması ve GPS'in aktif edilmesi,
- NMEA formatındaki konum bilgisinin alınması ve verinin işlenmesi,
- Kısa mesaj gönderimi

gibi sistemin kaza bildirim biriminde gerekli olan işlemler tanımlanmıştır.

[1] Kalman Filtresi, bir optimizasyon algoritmasıdır. Algoritma, gürültülü bir gözlem veri akışı (algılayıcı ölçüm sonuçları) üzerinde gerçek-zamanlı, özyinelemeli çalışarak hatayı en aza indirecek şekilde filtreleme yapar ve sistemin fiziksel karakteristiklerinin modellenmesi ile üretilen gelecek durumun matematiksel tahminine göre optimize eder.

BÖLÜM 5

SONUÇ VE ÖNERİLER

Bu tez çalışmasında, GSM servislerden biri olan SMS yolu ile GPS üzerinden konum bilgisinin iletilmesine olanak sağlayan gerçek zamanlı bir kaza tespit ve gösterim sistemi gerçekleştirilmiştir. Gerçekleştirilen sistem ile kaza yapan araca ait konum bilgisi SMS yoluyla öncedn belirlenen bir kullanıcıya gönderilmesi sağlanmıştır.

Kullanılmakta olan araç takip sistemleri yoğun şehir bölgeleri, kapalı alanlar, tüneller vb. yerlerde, GPS alıcılarının konum alamamasından dolayı izleme işlemlerinde kesintiler ortaya çıkmaktadır. Bu alanlarda yaşanacak kazalarda konum bilgisinin gönderilmesine yönelik olarak konumlandırma uydularından alınan son konum bilgisi, mikrodenetleyicinin EEPROM'larında saklanmaktadır. Bu sistem ile konumlandırma uydularıya iletişim kesildiğinde EEPROM'larda saklanan gerçek son konum bilgisinin gönderildiği tecrübe edilmiştir.

Kaza tespit yönteminde araçların maruz kalabileceği limit dışı ivme ve eğim hesapları baz alınmıştır. Dolayısıyla sisteme ait donanım araç içerisindeki konumundan bağımsız olarak herhangi bir noktadan kaza tespiti yapabilmektedir.

Yapılan performans testlerinde, kaza tespit ve gösterim sistemi kaza şartlarının oluşması halinde kaza yapan araca ait konum bilgisini 0.5sn ile 4.7 sn arasında SMS ile bildirmektedir. Bu sürenin artıp azalması GSM altyapısı ile ilişkilendirilmiştir. Konumlandırma sisteminden ise 2.5 metrelik bir hata payı ile koordinat bilgisi alınmıştır.

Sistemin tasarımında, kişisel güvenlik ve araç güvenliğine yönelik uygulama fonksiyonlarına sahip olması amaçlanmıştır. Bu maksatla, gerçekleştirilen kaza tespit ve

gösterim sistemine bir acil durum butonu eklenerek, sistemin kişisel güvenlik fonksiyonuna sahip olması sağlanmıştır. Bu fonksiyon özellikle taksi sürücülerinin tehlikeye maruz kalmaları durumunda kullanılabilir. Cihazın acil durum butonu araç içerisine gizlenerek ayak yardımıyla kullanılması sağlanabilir. Muhtemel bir tehlike, taksi durağına anında bildirilerek acil müdahale yapılabilir. Yapılan performans testlerinde acil durum bildirim fonksiyonunun sağlıklı biçimde çalıştığı ve anında konum bildirimi yapabildiği görülmüştür.

Kaza tespit ve gösterim sisteminde kaza anında konum gönderiminin yapılması sonrasında aracın olası uçurumda yuvarlanma gibi konum değişiminin veya panik butona basılması sonrasında aracın takip edilebilemesi maksadıyla 1 dakika aralıklar ile konum gönderilmesi sağlanmıştır.

Sistemde yer alan mikrodenetleyici ve GPS/ GSM modülü düşük güç tüketimine sahip olacak şekilde seçilmiştir. Sistemin batarya ile çalıştırıldığında uzun çalışması süresine (batarya kapasitesine göre değişiklik göstermektedir.) sahip olması amaçlanmıştır.

Sistemin, daha önce yapılan çalışmalardan ve piyasada mevcut ticari ürünlerden farkı kaza tespit yönetimi ve algoritmasıdır. Buna ek olarak kapalı alanlarda meydana gelen kazalarda dahili hafıza biriminden gerçek son konum bilgisinin göndermesi, sistemi diğer çalışma ve endüstriyel ürünlerden farklılaştırmıştır. Öte yandan, sadece SMS yoluyla (düşük güç tüketmesi) ve yinelemeli konum bildiriminde bulunması diğerlerinden farklılık oluşturması amacıyla projede çalışılmıştır. Yapılan çalışmanın çalışmalar ile karşılaştırılması Çizelge 5.1'de gösterilmiştir.

Gönderilen SMS içerisinde yer alan enlem boylam bilgileri akılı telefonların haritalarında yazılımlarına girilerek arama kurtarma çalışması yapacak ekip için kaza mevkiine ulaşmada en kısa yolun belirlenmesi sağlanmaktadır. Gerçekleştirilen sistem, insansız hava aracı, uçak, helikopter zeplin ve gibi hava uygulamalarında da kullanılabilir. Bu tür uygulamalarda araçla ilgili konum, yükseklik ve hız bilgilerinin yanı sıra bulunduğu ortama ait sıcaklık, basınç, nem vb. bilgiler de alınabilir.

Çizelge 5. 1 Yapılan Çalışmanın Diğer Çalışmalar ile Karşılaştırma Çizelgesi.

Çalışmalar / Özellikler	**Özgün Çalışma**	Trafik Kazaları İçin Acil Yardım Çağrı Sistemi Tasarımı [4]	Akıllı Telefon ile Kaza Tespit ve Bildirim Sistemi [5]	Araç Takip Cihazı [8]	Çarpışma Algılama Uygulaması SOSMART[11]	Araç Koruma Sistemi GrayBOX [12]	Kaza Tepit Sistemi [13]
Kaza Tepit	X	X	X		X		X
İvme Algılayıcısı	X				X		
Kaza Tespit Algoritması	X		X		X		X
GPS	X	X	X	X	X	X	X
Yaklaşık Konumlandırma Algoritması	X						
Bildirim (SMS)	X	X		X		X	X
Yinelemeli Bildirim	X						
Güç (Batarya)	X		X		X	X	X
Güç (Güç kaynağı)	X	X		X			

Gerçekleştirilen sistemde kullanılan konumlandırma modülünün GPS verileri içerisinde konum belirlenmesini sağlayan uydu sayısı, bu çalışmada gerekli olmayan ve dolayısı ile kullanılmayan birçok NMEA verisi mevcuttur. İleride yapılacak geliştirme çalışmalarında bu bilgiler kolaylıkla kullanılabilir durumdadır.

Öte yandan geniş bir sistem ağı ile ileride kurulacak merkezi bir arama kurtarma biriminin alt yapı çalışmalarında bu tezde tasarlanan sistemin kullanılabileceği ve gelecekte sel, yangın, deprem vb. farklı felaketleri algılayan tespit ve bildirim sistemlerinin hayati önem arz edeceği değerlendirilmektedir.

KAYNAKLAR

[1] Yiğit, E., (2009). GPS Teknolojisi ile Konum Tespit Sistemi Tasarımı, Yüksek Lisans Tezi, Beykent Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

[2] Aygen, M., Fırat, H., Güneş, M., Yamaçlı,S. ve Küçük, E., (2013). SMS Tabanlı Araç Takip Sistemi, Yüksek Lisans Tezi, Mersin Üniversitesi Fen Bilimleri Enstitüsü, Mersin.

[3] Priyadharshini, S., Anupriya, N., Maheswari, S. ve Sellam, S., (2016). "Tracking and Theft Prevention System for Two Wheeler Using GSM and GPS", International Journal for Research in Applied Science&Engineering Technology (IJRASET), 6 (3):324-334.

[4] Demircan, T., Yıldırım, S., Yıldız,M., Ekşi,Z. ve Çakıroğlu,M., (2013). "Trafik Kazaları İçin Bir Acil Yardım Çağrı Sistemi Tasarımı", SAÜ. Fen Bil. Der. 17 (3):315-319.

[5] Thompson, C., Whit, J., Doughenty, B., Abnight, A., Schmidt, D.C., (2010). "Using Smart Phone to Detect Car Accidents and Provide Situational Awareness Emergency Responses", LNICST 48:29-42.

[6] Child Guard Dijital Takip Cihazı, http://weewell.com/urunler/ev-bebek/takip-cihazlari/takip-cihazlari-wct100.html, 09 Mayıs 2017.

[7] STI GL300 Real-Time GPS Tracker, http://www.spytecinc.com/gl-200-real-time-gps-tracker.html, 09 Mayıs 2017.

[8] MINI A8, https://www.istanbultekno.com/mini-arac-sinyal-izleme-ve-ses-kay%c4%b1t-cihazi-mini-a8.html, 09 Mayıs 2017.

[9] ATS 911, http://piranha.com.tr/arac-takip/product.php?product_id=625, 09 Mayıs 2017.

[10] OBDPLUS AT-90, http://obdplus.com.tr/obdplus.html, 09 Mayıs 2017.

[11] SOSMART, http://www.sosmartapp.com/, 09 Mayıs 2017.

[12] GrayBOX, http://www.instructables.com/id/GrayBOX-Accident-Detection-and-

Theft-Protection-Sy/, 09 Mayıs 2017.

[13] Vehicle Accident Detection with GPS and GSM modem, http://www.projectsof8051.com/vehicle-accident-detection-with-gps-and-gsm-modem/, 09 Mayıs 2017.

[14] TUİK Trafik Kaza İstatistikleri, 2015, http://www.tuik.gov.tr/PreHaberBultenleri.do?id=21611, 09 Mayıs 2017.

[15] Roco, M.C., (2001). "A Frontier for Engineering – The Aim of Nanotechnology to Build the Future, Molecule by Molecule", Mechanical Engineering ,123 (1):52-55.

[16] Kendall, J., (2014). Air Bag Deployment Criteria, The Forensic Examiner, Institute of Risk &Safety Analyses, California.

[17] Gad-el-Hak, M., (2001). The MEMS Handbook, CRC Press, USA.

[18] The Starlino Electronics project, A Guide To using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications, http://www.starlino.com/imu_guide.html , 11 Ocak 2017.

[19] Lee, D.B., Chitrakaran, V., Burg, T., Dawson, D. ve Xian, B. (2007). "Control of a remote lyoperated quadroto raerial vehicle and camera unit using a fly-the-camera perspective", Proceedings of the IEEE Conference on Decisionand Control, 6412–6417.

[20] Analog Devices, ADXL345 3-Axis, ±2 g / ±4 g / ±8 g / ±16 g Digital Accelerometer Datasheet.

[21] GPS, https://tr.wikipedia.org/wiki/GPS, 09 Mayıs 2017.

[22] Karaali, C. ve Yıldırım, Ö., (1996). "Global Konum Belirleme Sistemi (GPS)", PAÜ. Mühendislik Bilimleri Der. 2 (2):103-108.

[23] Wells, D.E., Beck, N., Delikaraoğlu, D., Kleusberg, A.,Krakiwsky, E.E., Lachapelle, G., Langey, R.B., Nakiboğlu, M., Schwarz, K.P., Tranquilla, J.M. ve Vanicek, P., (1987). Guide To GPS Positioning, Second Edition, Canadian GPS Associates, New Brunswick, Canada.

[24] Mekik, C., (2010). "Global Uydu Navigasyon Sistemleri ve Uydu Bazlı Alan Büyütme Sistemleri", CeBIT-Eurasia 2010 Coğrafi Bilgi Teknolojileri Çalıştayı, 8 Ekim 2010, Istanbul.

[25] King, E.W., Masters, E.G., Rizos, C., Stoltz, A. ve Collins, C., (1987). Surveying wiht Global Positioning System, Dümmler, Bonn.

[26] Kahveci, M., (1993). Ortometrik Yüksekliklerin Belirlenmesinde GPS Sistemi, Yüksek Lisans Tezi, Fen Bilimleri Enstitüsü, İstanbul.

[27] Seeber, G., (1993). Satellite Geodesy, Hannover.

[28] U.S. Coast Guard Navigation Center, Navstar GPS User Equipment Introduction, http://www.gps.gov/pub/gps/gpsuser/gpsuser.pdf, 09 Mayıs 2017.

[29] O'Sullivan D.ve Igoe T., (2004). "Physical Computing: Sensing and Controllingthe Physical World with Computers", Course Technology, 494.

[30] Cincirop, B., (2009). GSM Kontrollü Akıllı Ev Otomasyonu, Yüksek Lisans Tezi, Sakarya Üniversitesi Fen Bilimleri Enstitüsü, Sakarya.

[31] Kriegl, J., (2000). "Location in cellular Networks" , University of Technology, Institute for Applied Information Processing and Communications, Graz, 15-20.

[32] Willasen, S. Y., (2003). "Forensics and the GSM mobile telephone system", International Journal of Digital Evidence, 2 (1): 1-17.

[33] Noldus, R., (2006). Intelligent networks for the GSM, GPRS and UMTS Networks, John Wiley&Sons Ltd., Washington, 65-88.

[34] Willig, A., (2003). The GSM air interface fundamental sand protocols, University of Potsdam, Hasso-Plattner Institude, Potsdam, 11-17.

[35] Yüksel, M., GSM/GPRS Aygıtlar Üzerinden Çok Dilli SMS Gönderme http://slideplayer.biz.tr/slide/2313715/, 09 Mayıs 2017.

[36] Altınbaşak O., (2000). Mikrodenetleyiciler ve PIC programlama Eylül İstanbul.

[37] Fried, L., (2016). Adafruit FONA 808 Cellular + GPS Breakout, Adafruit Industries Co., New York, USA.

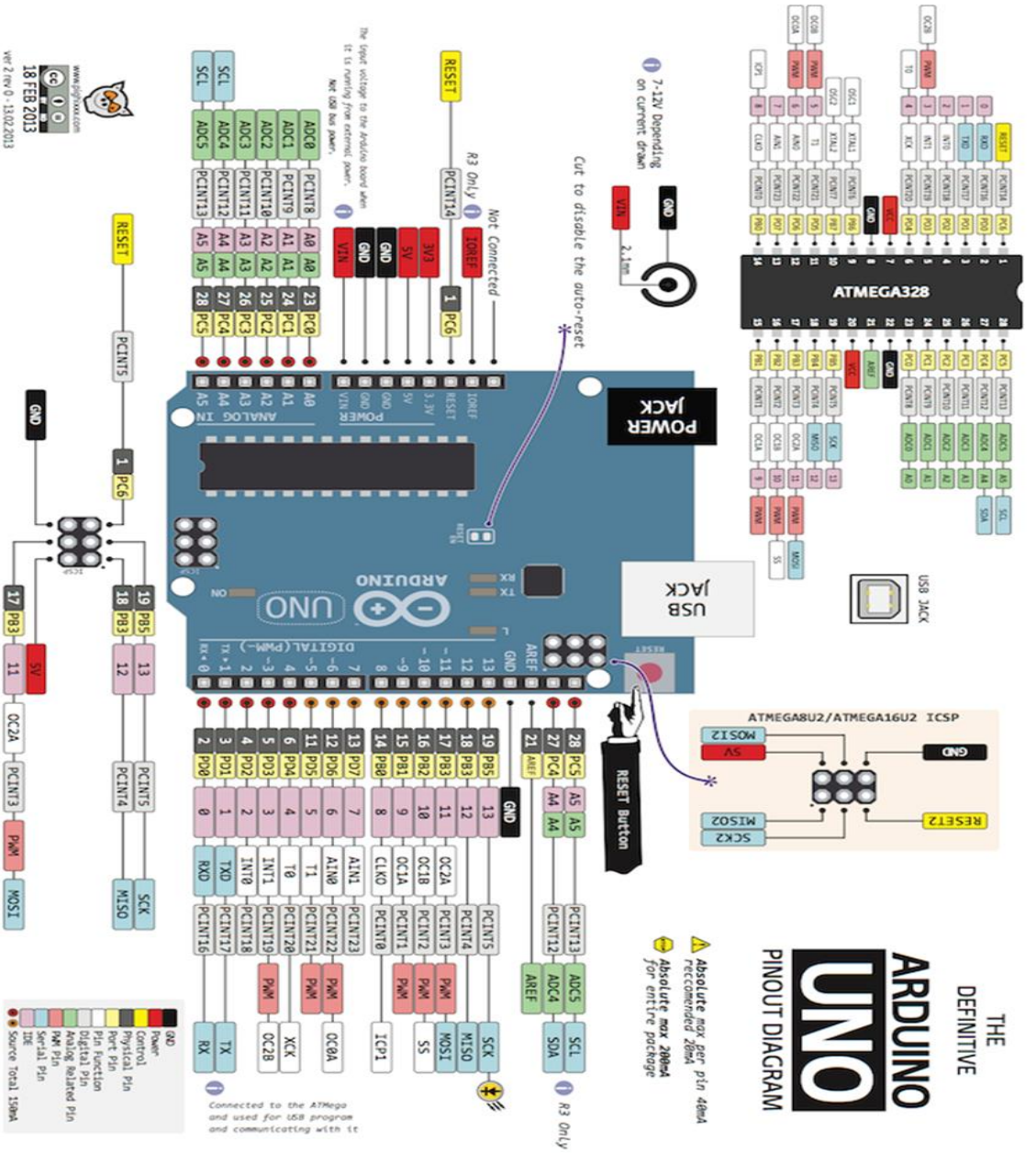
[38] Bükey, M.V., (2012). Kurtarma Robotlarında Hazır Elektronik Kartlar ile İleri Kontrol Uygulamaları, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

[39] AT Komut Seti, http://gokhanbeken.com/at-komut-setigsm/, 09 Mayıs 2017.

[40] Shanghai SIMCom Wireless Solution Ltd., SIM808 Hardware Design V1.00", Shanghai, China, 27 Mart 2014.

EK-A

ARDIUNO UNO BAĞLANTI NOKTALARI



EK-B

KAZA TESPİT VE GÖSTERİM SİSTEMİ YAZILIMI

B-1 Ana Kontrol Yazılım

```
#include "ADXL345.h"//ADXL345 sensörü kütüphanesi
#include <Wire.h> //I2C haberleşme kütüphanesi
#include "Adafruit_FONA.h"//Konumlandırma ve haberleşme modülü kütüphanesi
#include <SoftwareSerial.h>//Seri Port Haberleşme Kütüphanesi
#include <EEPROM.h> //EEPROM Kütüphanesi
ADXL345 accel;//ADXL345 kütüphanesinden ivme okuma fonksiyonun tanıtımı
double Bileske_ivme, Max_bileske_ivme, yatayegim, Max_yatayegim;//İvme algılayıcısı için değişkenlerin tanımlanması
int SMSgonderme=0; // Kaza şartının oluşumunu anahtarlayan değişken
int acildurum=0;// Kaza veya Acil Durumu anahtarlayan değişken
char numara[]="05053857887";//tanımlanmış numara
char mesaj[140];//mesaj metni için değişken ataması
char enlem[100];//Enlem için değişken tanımlaması
char boylam[100];//Boylam için değişken tanımlaması
char m1[]="XXX YY QQQ plaka numarali araciniz kaza yapti... Konum ";//mesaj metni tanımlaması
char m2[]="XXX YY QQQ plaka numarali araciniz icin acil durum...Konum ";//mesaj metni tanımlaması
char m3[]="https://www.google.com.tr/maps/@";//akıllı telefonlarda konum oluşturma linki
int butonPin=7;//Panik Buton için pin tanımlaması
int butonDurum=0; //Panik Buton durum tanımlaması
char gpsverisi[120];//GPS verisi için tanımlanan array
```

```
#define FONA_RX 2
#define FONA_TX 3  //Konumlandırma ve Haberleşme modülü haberleşme portlarının tanımlanması
#define FONA_RST 4
SoftwareSerial fonaSS = SoftwareSerial(FONA_TX, FONA_RX);//Konumlandırma ve Haberleşme modülü haberleşme portlarının tanıtılması
SoftwareSerial *fonaSerial = &fonaSS;
Adafruit_FONA fona = Adafruit_FONA(FONA_RST);

void setup() {

  Serial.begin(115200);//Seri port haberleşmesini başlatma

  Wire.begin();//Sensör ile haberleşmeyi başlatma
  accel.init(-4, -4, 4);//Sensörün başlangıç değerlerinin tanımlaması için OFFSET değerleri
  accel.setSoftwareOffset(-0.01, -0.015, -0.011);//Kalibrasyon OFFSET değerleri
  accel.printCalibrationValues(40);//(Kalibrasyon 4 saniye boyunca değer alınmaktadır.)

  fonaSerial->begin(9600);//SIM808 Konumlandırma ve Haberleşme Modülü haberleşme hızı
  fona.begin(*fonaSerial);//SIM808 Konumlandırma ve Haberleşme Modülü haberlşemesini başlatma

  fona.enableGPS(true);//Konumlandırmayı aç
  char PIN[5]="2019";//PIN numarası
  fona.unlockSIM(PIN);//Haberleşme modulunu aç
  pinMode(butonPin,INPUT);//Panik buton için pin atanması
}

// EEPROM üzerinde konum verisinin kaydedilmesi ve okunması için oluşturulan fonksiyonlar.
  void EEPROM_Write(float *num, int MemPos)//EEPROM üzerine Float değer yazabilmek için yazılan fonksiyon
{
```

```
 byte ByteArray[4];
 memcpy(ByteArray, num, 4);
 for(int x = 0; x < 4; x++)
 {
   EEPROM.write((MemPos * 4) + x, ByteArray[x]);
 }
}

void EEPROM_Read(float *num, int MemPos)//EEPROM üzerine Float değer okuyabilmek için yazılan fonksiyon
{
 byte ByteArray[4];
 for(int x = 0; x < 4; x++)
 {
   ByteArray[x] = EEPROM.read((MemPos * 4) + x);
 }
 memcpy(num, ByteArray, 4);
}

void loop() {

  AccelG accelG;//Kütüphane fonksiyonundan yeni fonksiyon tanımlanması
  accelG = accel.readAccelG(); //Sensörden G biriminde ivme okunması

  Bileske_ivme = sqrt(pow(accelG.x,2)+pow(accelG.y,2)+pow(accelG.z,2));//bileşke ivme değerinin hesaplanması
  yatayegim = (atan2(accelG.y,(sqrt(accelG.x*accelG.x+accelG.z*accelG.z))) * 180.0) / PI;

  if (Bileske_ivme > Max_bileske_ivme)
    {//maksimum bileşke değerinin tespit edilmesi için kurulan denklem
      Max_bileske_ivme=Bileske_ivme;
    }
  if (abs(yatayegim) > abs(Max_yatayegim))
    {//maksimum bileşke değerinin tespit edilmesi için kurulan denklem
```

```
    Max_yatayegim=abs (yatayegim);
   }

  butonDurum=digitalRead(butonPin);//Panik buton değerinin okunması

  uint8_t konum= fona.getGPS(32, gpsverisi, 120);////SIM808 Konumlandırma ve Haberleşme Modülü üzerinden
                                ////GPS verisinin okunması için fonksiyon tanımlanması
  konum; ////GPS verisinin okunması için fonksiyonu çağırma

  char *durumm = strtok(gpsverisi, ",");//GPS verisinden GPS haberleşmesinin yapıldığı duruma ait bilginin alınması
  char *fixx = strtok(NULL, ",");//Konum atılmasına ait bilginin alınması
  char *tarihh = strtok(NULL, ",");//GPS'den tarih bilgisinin alınması
  char *enlemm = strtok(NULL, ",");//GPS'den enlem bilgisinin alınması
  char *boylamm = strtok(NULL, ",");//GPS'den boylam bilgisini alınması
  strcpy(boylam,boylamm);//Array kopyalama işlemi
  strcpy(enlem,enlemm);//Array kopyalama işlemi
  int fix,durum;
  fix = atof(fixx);//sayısal değere çevirme
  durum = atof(durumm);//sayısal değere çevirme

  switch (fix){//GPS haberleşmesi durumununa göre anahtarlama
    case 1://Haberleşme var ise konum EEPROM'a yazılır.
      float en,boy;
      en = atof(enlem);
      boy = atof(boylam);
      EEPROM_Write(&en, 1);
      EEPROM_Write(&boy, 32);
    break;

    case 0://Haberleşme yok ise konum EEPROM'dan okunur.
      EEPROM_Read(&en, 1);
      EEPROM_Read(&boy, 32);
```

```
    dtostrf(en, 2, 6, enlem);
    dtostrf(boy, 2, 6, boylam);
   break;
 }

 switch (SMSgonderme){
  case 0:
   if (Max_bileske_ivme>3.7 || Max_yatayegim>45){//Kaza durum algoritması
      SMSgonderme=1;
      acildurum=1;
     }
   else if (butonDurum==HIGH){//Acil Durum algoritması
      SMSgonderme=1;
      acildurum=2;
     }
   else{
       SMSgonderme=0;
     }
  break;

  case 1:
   switch(acildurum){
     case 1://kaza için
      snprintf(mesaj, sizeof mesaj, "%s%s,%s %s%s,%s", m1, enlem, boylam, m3, enlem, boylam);//mesaj metninin oluşturma
     break;

     case 2://Acil Durum için
      snprintf(mesaj, sizeof mesaj, "%s%s,%s %s%s,%s", m2, enlem, boylam, m3, enlem, boylam);//mesaj metninin oluşturma
     break;
   }

   fona.sendSMS(numara, mesaj);//SIM808 kütüphanesinde tanımlı SMS gönderme fonksiyonu
```

```
    delay(60000);//1 dakikalık gecikme
   break;

 }
}
```

B-2 ADXL345 Kütüphanesi

- CCP Dosyası:

```
#include "Arduino.h"
#include "ADXL345.h"
#include <math.h>
ADXL345::ADXL345()
{
  xg =0;
  yg=0;
  zg=0;
}

void ADXL345::init(char x_offset, char y_offset, char z_offset)
{
  writeTo(ADXL345_POWER_CTL, 8);

  writeTo(ADXL345_DATA_FORMAT, 0x0B);

  writeTo(ADXL345_OFSX, x_offset);
  writeTo(ADXL345_OFSY, y_offset);
  writeTo(ADXL345_OFSZ, z_offset);
}
void ADXL345::setSoftwareOffset(double x, double y, double z)
{
  _xoffset = x;
  _yoffset = y;
  _zoffset = z;
}
AccelRotation ADXL345::readPitchRoll()
{
  AccelG accel;
  accel = readAccelG();
}
```

```
void ADXL345::printCalibrationValues(int samples)
{
  double x,y,z;
  double xt,yt,zt;
  xt = 0;
  yt = 0;
  zt = 0;

  Serial.print("Calibration in: 3");
  delay(1000);
  Serial.print(" 2");
  delay(1000);
  Serial.println(" 1");
  delay(1000);
  for(int i=0; i<samples; i++)
  {
    AccelG accel = readAccelG();
    xt += accel.x;
    yt += accel.y;
    zt += accel.z;
    delay(100);
  }
  Serial.println("Accel Offset (mg): ");
  Serial.print("X: ");
  Serial.print(xt/float(samples)*1000,5);
  Serial.print(" Y: ");
  Serial.print(yt/float(samples)*1000,5);
  Serial.print(" Z: ");
  Serial.println( zt/float(samples)*1000,5);
  delay(2000);
}
// Writes val to address register on device
void ADXL345::writeTo(byte address, byte val)
{
```

```
  Wire.beginTransmission(ADXL345_DEVICE); // start transmission to device
  Wire.write(address); // send register address
  Wire.write(val); // send value to write
  Wire.endTransmission(); // end transmission
}
AccelG ADXL345::readAccelG()
{
  AccelRaw raw;
  raw = readAccel();
  double fXg, fYg, fZg;
  fXg =raw.x * 0.00390625 + _xoffset;
  fYg =raw.y * 0.00390625 + _yoffset;
  fZg =raw.z * 0.00390625 + _zoffset;
  AccelG res;
  res.x = fXg * ALPHA + (xg * (1.0-ALPHA));
  xg = res.x;
  res.y = fYg * ALPHA + (yg * (1.0-ALPHA));
  yg = res.y;
  res.z = fZg * ALPHA + (zg * (1.0-ALPHA));
  zg = res.z;
  return res;
 }
AccelRaw ADXL345::readAccel()
{
  readFrom(ADXL345_DATAX0, ADXL345_TO_READ, _buff); //ivme bilgisinin okunması

  AccelRaw raw;
  raw.x = (((int)_buff[1]) << 8) | _buff[0];
  raw.y = (((int)_buff[3]) << 8) | _buff[2];
  raw.z = (((int)_buff[5]) << 8) | _buff[4];
  return raw;
}
void ADXL345::readFrom(byte address, int num, byte _buff[])
{
```

```
  Wire.beginTransmission(ADXL345_DEVICE);
  Wire.write(address);
  Wire.endTransmission();
  Wire.beginTransmission(ADXL345_DEVICE);
  Wire.requestFrom(ADXL345_DEVICE, num);
  int i = 0;
  while(Wire.available())  {
    _buff[i] = Wire.read(); // receive a byte
    i++;
  }
  Wire.endTransmission(); // end transmission
}
void ADXL345::printAllRegister()
{
byte _b;
Serial.print("0x00: ");
readFrom(0x00, 1, &_b);
print_byte(_b);
Serial.println("");
int i;
for (i=29;i<=57;i++)
      {
Serial.print("0x");
Serial.print(i, HEX);
Serial.print(": ");
readFrom(i, 1, &_b);
print_byte(_b);
Serial.println("");
}
}
void ADXL345::print_byte(byte val)
{
int i;
Serial.print("B");
```

```
for(i=7; i>=0; i--){
Serial.print(val >> i & 1, BIN);
}
}
```

- H Dosyası:

```
#ifndef ADXL345_h
#define ADXL345_h

#include <Wire.h>
#include "Arduino.h"



#define ADXL345_DEVICE 0x53
#define ADXL345_TO_READ 6

#define ADXL345_POWER_CTL 0x2d
#define ADXL345_DATAX0 0x32
#define ADXL345_DATA_FORMAT 0x31

#define ADXL345_OFSX 0x1E
#define ADXL345_OFSY 0x1F
#define ADXL345_OFSZ 0x20

#define ALPHA 0.5
struct AccelRaw
{
  int x;
  int y;
  int z;
};

struct AccelG
{
  double x;
  double y;
  double z;
};
```

```
class ADXL345
{
  public:
    ADXL345();
    void init(char x_offset=0, char y_offset=0, char z_offset=0);
    void writeTo(byte address, byte val);
    AccelRaw readAccel();
    AccelG readAccelG();
    void readFrom(byte address, int num, byte _buff[]);
    void printAllRegister();
    void print_byte(byte val);
    void printCalibrationValues(int samples);
    AccelRotation readPitchRoll();
    void setSoftwareOffset(double x, double y, double z);

  private:
     byte _buff[6];

     double xg;
     double yg;
     double zg;

     double _xoffset;
     double _yoffset;
     double _zoffset;
};
#endif
```

B-3 Adafurit

- CPP Dosyası:

```
#include "Adafruit_FONA.h"
```

```
Adafruit_FONA::Adafruit_FONA(int8_t rst)
{
  _rstpin = rst;

  apn = F("FONAnet");
  apnusername = 0;
  apnpassword = 0;
  mySerial = 0;
  httpsredirect = false;
  useragent = F("FONA");
  ok_reply = F("OK");
}

uint8_t Adafruit_FONA::type(void) {
  return _type;
}

boolean Adafruit_FONA::begin(Stream &port) {
  mySerial = &port;

  pinMode(_rstpin, OUTPUT);
  digitalWrite(_rstpin, HIGH);
  delay(10);
  digitalWrite(_rstpin, LOW);
  delay(100);
  digitalWrite(_rstpin, HIGH);

  DEBUG_PRINTLN(F("Attempting to open comm with ATs"));
  // give 7 seconds to reboot
  int16_t timeout = 700;

  while (timeout > 0) {
    while (mySerial->available()) mySerial->read();
    if (sendCheckReply(F("AT"), ok_reply))
```

```
    break;
   while (mySerial->available()) mySerial->read();
   if (sendCheckReply(F("AT"), F("AT")))
    break;
   delay(5);
   timeout-=30;
  }

  if (timeout <= 0) {
#ifdef ADAFRUIT_FONA_DEBUG
   DEBUG_PRINTLN(F("Timeout: No response to AT... last ditch attempt."));
#endif
   sendCheckReply(F("AT"), ok_reply);
   delay(50);
   sendCheckReply(F("AT"), ok_reply);
   delay(50);
   sendCheckReply(F("AT"), ok_reply);
   delay(50);
  }

  // turn off Echo!
  sendCheckReply(F("ATE0"), ok_reply);
  delay(100);

  if (! sendCheckReply(F("ATE0"), ok_reply)) {
   return false;
  }

  // turn on hangupitude
  sendCheckReply(F("AT+CVHU=0"), ok_reply);

  delay(10);
  flushInput();
```

```
DEBUG_PRINT(F("\t---> ")); DEBUG_PRINTLN("ATI");

mySerial->println("ATI");
readline(500, true);

DEBUG_PRINT (F("\t<--- ")); DEBUG_PRINTLN(replybuffer);

if (prog_char_strstr(replybuffer, (prog_char *)F("SIM808 R14")) != 0) {
  _type = FONA808_V2;
} else if (prog_char_strstr(replybuffer, (prog_char *)F("SIM808 R13")) != 0) {
  _type = FONA808_V1;
} else if (prog_char_strstr(replybuffer, (prog_char *)F("SIM800 R13")) != 0) {
  _type = FONA800L;
} else if (prog_char_strstr(replybuffer, (prog_char *)F("SIMCOM_SIM5320A")) != 0) {
  _type = FONA3G_A;
} else if (prog_char_strstr(replybuffer, (prog_char *)F("SIMCOM_SIM5320E")) != 0) {
  _type = FONA3G_E;
}

if (_type == FONA800L) {
  // determine if L or H

DEBUG_PRINT(F("\t---> ")); DEBUG_PRINTLN("AT+GMM");

  mySerial->println("AT+GMM");
  readline(500, true);

DEBUG_PRINT (F("\t<--- ")); DEBUG_PRINTLN(replybuffer);

  if (prog_char_strstr(replybuffer, (prog_char *)F("SIM800H")) != 0) {
    _type = FONA800H;
  }
}
```

```
#if defined(FONA_PREF_SMS_STORAGE)
  sendCheckReply(F("AT+CPMS=\"" FONA_PREF_SMS_STORAGE "\""), ok_reply);
#endif

  return true;
}

/********* Serial port ****************************************/
boolean Adafruit_FONA::setBaudrate(uint16_t baud) {
  return sendCheckReply(F("AT+IPREX="), baud, ok_reply);
}

/********* Real Time Clock ************************************/

boolean Adafruit_FONA::readRTC(uint8_t *year, uint8_t *month, uint8_t *date,
uint8_t *hr, uint8_t *min, uint8_t *sec) {
  uint16_t v;
  sendParseReply(F("AT+CCLK?"), F("+CCLK: "), &v, '/', 0);
  *year = v;

  DEBUG_PRINTLN(*year);
}

boolean Adafruit_FONA::enableRTC(uint8_t i) {
  if (! sendCheckReply(F("AT+CLTS="), i, ok_reply))
    return false;
  return sendCheckReply(F("AT&W"), ok_reply);
}

/********* SIM
*************************************************************/

uint8_t Adafruit_FONA::unlockSIM(char *pin)
```

```
{
  char sendbuff[14] = "AT+CPIN=";
  sendbuff[8] = pin[0];
  sendbuff[9] = pin[1];
  sendbuff[10] = pin[2];
  sendbuff[11] = pin[3];
  sendbuff[12] = '\0';

  return sendCheckReply(sendbuff, ok_reply);
}

uint8_t Adafruit_FONA::getSIMCCID(char *ccid) {
  getReply(F("AT+CCID"));// up to 28 chars for reply, 20 char total ccid
  if (replybuffer[0] == '+') {// fona 3g?
    strncpy(ccid, replybuffer+8, 20);
  } else {// fona 800 or 800
    strncpy(ccid, replybuffer, 20);
  }
  ccid[20] = 0;
  readline(); // eat 'OK'
  return strlen(ccid);
}

/********* IMEI
**********************************************************/

uint8_t Adafruit_FONA::getIMEI(char *imei) {
  getReply(F("AT+GSN")); // up to 15 chars
  strncpy(imei, replybuffer, 15);
  imei[15] = 0;
  readline(); // eat 'OK'
  return strlen(imei);
}
```

```
/********* NETWORK
*******************************************************/

uint8_t Adafruit_FONA::getNetworkStatus(void) {
  uint16_t status;
  if (! sendParseReply(F("AT+CREG?"), F("+CREG: "), &status, ',', 1)) return 0;
  return status;
}

uint8_t Adafruit_FONA::getRSSI(void) {
  uint16_t reply;
  if (! sendParseReply(F("AT+CSQ"), F("+CSQ: "), &reply) ) return 0;
  return reply;
}

/********* SMS
*******************************************************/

uint8_t Adafruit_FONA::getSMSInterrupt(void) {
  uint16_t reply;

  if (! sendParseReply(F("AT+CFGRI?"), F("+CFGRI: "), &reply) ) return 0;
  return reply;
}
boolean Adafruit_FONA::setSMSInterrupt(uint8_t i) {
  return sendCheckReply(F("AT+CFGRI="), i, ok_reply);
}
int8_t Adafruit_FONA::getNumSMS(void) {
  uint16_t numsms; // get into text mode
  if (! sendCheckReply(F("AT+CMGF=1"), ok_reply))return -1;// ask how many sms are
stored
  if (sendParseReply(F("AT+CPMS?"), F("\"SM\","), &numsms))
    return numsms;
  if (sendParseReply(F("AT+CPMS?"), F("\"SM_P\","), &numsms))
    return numsms;
```

```
  return -1;
}

boolean Adafruit_FONA::readSMS(uint8_t i, char *smsbuff,
      uint16_t maxlen, uint16_t *readlen) { // text mode
  if (! sendCheckReply(F("AT+CMGF=1"), ok_reply)) return false;// show all text mode
parameters
  if (! sendCheckReply(F("AT+CSDH=1"), ok_reply)) return false;// parse out the SMS len
  uint16_t thesmslen = 0;

  DEBUG_PRINT(F("AT+CMGR="));
  DEBUG_PRINTLN(i);
  mySerial->print(F("AT+CMGR="));
  mySerial->println(i);
  readline(1000); // timeout

  DEBUG_PRINTLN(replybuffer);

  if (! parseReply(F("+CMGR:"), &thesmslen, ',', 11)) {
    *readlen = 0;
    return false;
  }
  readRaw(thesmslen);
  flushInput();
  uint16_t thelen = min(maxlen, strlen(replybuffer));
  strncpy(smsbuff, replybuffer, thelen);
  smsbuff[thelen] = 0; // end the string
  DEBUG_PRINTLN(replybuffer);
  *readlen = thelen;
  return true;
}

boolean Adafruit_FONA::getSMSSender(uint8_t i, char *sender, int senderlen) {
// Ensure text mode and all text mode parameters are sent.
```

```
  if (! sendCheckReply(F("AT+CMGF=1"), ok_reply))return false;
  if (! sendCheckReply(F("AT+CSDH=1"), ok_reply)) return false;

  DEBUG_PRINT(F("AT+CMGR="));
  DEBUG_PRINTLN(i);
  // Send command to retrieve SMS message and parse a line of response.
  mySerial->print(F("AT+CMGR="));
  mySerial->println(i);
  readline(1000);

  DEBUG_PRINTLN(replybuffer);  // Parse the second field in the response.
  boolean result = parseReplyQuoted(F("+CMGR:"), sender, senderlen, ',', 1); // Drop
any remaining data from the response.
  flushInput();
  return result;
}

boolean Adafruit_FONA::sendSMS(char *smsaddr, char *smsmsg) {
  if (! sendCheckReply(F("AT+CMGF=1"), ok_reply)) return false;
  char sendcmd[30] = "AT+CMGS=\"";
  strncpy(sendcmd+9, smsaddr, 30-9-2);  // 9 bytes beginning, 2 bytes for close quote +
null
  sendcmd[strlen(sendcmd)] = '\"';
  if (! sendCheckReply(sendcmd, F("> "))) return false;
  DEBUG_PRINT(F("> ")); DEBUG_PRINTLN(smsmsg);
  mySerial->println(smsmsg);
  mySerial->println();
  mySerial->write(0x1A);
  DEBUG_PRINTLN("^Z");
  if ( (_type == FONA3G_A) || (_type == FONA3G_E) ) {// Eat two sets of CRLF
    readline(200);
    readline(200);
  }
  readline(10000); // read the +CMGS reply, wait up to 10 seconds!!!
```

```
  if (strstr(replybuffer, "+CMGS") == 0) {
    return false;
  }
  readline(1000); // read OK

  if (strcmp(replybuffer, "OK") != 0) {
    return false;
  }

  return true;
}

boolean Adafruit_FONA::deleteSMS(uint8_t i) {
  if (! sendCheckReply(F("AT+CMGF=1"), ok_reply)) return false;  // read an sms
  char sendbuff[12] = "AT+CMGD=000";
  sendbuff[8] = (i / 100) + '0';
  i %= 100;
  sendbuff[9] = (i / 10) + '0';
  i %= 10;
  sendbuff[10] = i + '0';
  return sendCheckReply(sendbuff, ok_reply, 2000);
}

/********* TIME
************************************************************/

boolean Adafruit_FONA::enableNetworkTimeSync(boolean onoff) {
  if (onoff) {
    if (! sendCheckReply(F("AT+CLTS=1"), ok_reply))
      return false;
  } else {
    if (! sendCheckReply(F("AT+CLTS=0"), ok_reply))
      return false;
```

```
  }
  flushInput(); // eat any 'Unsolicted Result Code'
  return true;
}

boolean Adafruit_FONA::enableNTPTimeSync(boolean onoff, FONAFlashStringPtr
ntpserver) {
  if (onoff) {
    if (! sendCheckReply(F("AT+CNTPCID=1"), ok_reply))
      return false;
    mySerial->print(F("AT+CNTP=\""));
    if (ntpserver != 0) {
      mySerial->print(ntpserver);
    } else {
      mySerial->print(F("pool.ntp.org"));
    }
    mySerial->println(F("\",0"));
    readline(FONA_DEFAULT_TIMEOUT_MS);
    if (strcmp(replybuffer, "OK") != 0)
      return false;
    if (! sendCheckReply(F("AT+CNTP"), ok_reply, 10000))
      return false;

    uint16_t status;
    readline(10000);
    if (! parseReply(F("+CNTP:"), &status))
      return false;
  } else {
    if (! sendCheckReply(F("AT+CNTPCID=0"), ok_reply))
      return false;
  }
  return true;
}
```

```
boolean Adafruit_FONA::getTime(char *buff, uint16_t maxlen) {

  getReply(F("AT+CCLK?"), (uint16_t) 10000);
  if (strncmp(replybuffer, "+CCLK: ", 7) != 0)
    return false;
  char *p = replybuffer+7;
  uint16_t lentocopy = min(maxlen-1, strlen(p));
  strncpy(buff, p, lentocopy+1);
  buff[lentocopy] = 0;
  readline(); // eat OK
  return true;
}

/********* GPS
**********************************************************/

boolean Adafruit_FONA::enableGPS(boolean onoff) {
  uint16_t state; // first check if its already on or off

  if (_type == FONA808_V2) {
    if (! sendParseReply(F("AT+CGNSPWR?"), F("+CGNSPWR: "), &state) )
      return false;
  } else {
    if (! sendParseReply(F("AT+CGPSPWR?"), F("+CGPSPWR: "), &state))
      return false;
  }
  if (onoff && !state) {
    if (_type == FONA808_V2) {
      if (! sendCheckReply(F("AT+CGNSPWR=1"), ok_reply))  // try GNS command
return false;
    } else {
      if (! sendCheckReply(F("AT+CGPSPWR=1"), ok_reply))
return false;
    }
```

```
  } else if (!onoff && state) {
    if (_type == FONA808_V2) {
      if (! sendCheckReply(F("AT+CGNSPWR=0"), ok_reply)) // try GNS command
return false;
    } else {
      if (! sendCheckReply(F("AT+CGPSPWR=0"), ok_reply))
return false;
    }
  }
  return true;
}

boolean Adafruit_FONA_3G::enableGPS(boolean onoff) {
  uint16_t state;

  // first check if its already on or off
  if (! Adafruit_FONA::sendParseReply(F("AT+CGPS?"), F("+CGPS: "), &state) )
    return false;

  if (onoff && !state) {
    if (! sendCheckReply(F("AT+CGPS=1"), ok_reply))
      return false;
  } else if (!onoff && state) {
    if (! sendCheckReply(F("AT+CGPS=0"), ok_reply))
      return false;// this takes a little time
    readline(2000); // eat '+CGPS: 0'
  }
  return true;
}

int8_t Adafruit_FONA::GPSstatus(void) {
  if (_type == FONA808_V2) {
    getReply(F("AT+CGNSINF"));
    char *p = prog_char_strstr(replybuffer, (prog_char*)F("+CGNSINF: "));
```

```
  if (p == 0) return -1;
  p+=10;
  readline(); // eat 'OK'
  if (p[0] == '0') return 0; // GPS is not even on!

  p+=2; // Skip to second value, fix status.
  // Assume if the fix status is '1' then we have a 3D fix, otherwise no fix.
  if (p[0] == '1') return 3;
  else return 1;
 }
 if (_type == FONA3G_A || _type == FONA3G_E) {
  getReply(F("AT+CGPSINFO"));
  char *p = prog_char_strstr(replybuffer, (prog_char*)F("+CGPSINFO:"));
  if (p == 0) return -1;
  if (p[10] != ',') return 3; // if you get anything, its 3D fix
  return 0;
 }
 else {// 808 V1 looks for specific 2D or 3D fix state.
  getReply(F("AT+CGPSSTATUS?"));
  char *p = prog_char_strstr(replybuffer, (prog_char*)F("SSTATUS: Location "));
  if (p == 0) return -1;
  p+=18;
  readline(); // eat 'OK'
  //DEBUG_PRINTLN(p);
  if (p[0] == 'U') return 0;
  if (p[0] == 'N') return 1;
  if (p[0] == '2') return 2;
  if (p[0] == '3') return 3;
 }
 return 0;
}

uint8_t Adafruit_FONA::getGPS(uint8_t arg, char *buffer, uint8_t maxbuff) {
 int32_t x = arg;
```

```
  if ( (_type == FONA3G_A) || (_type == FONA3G_E) ) {
    getReply(F("AT+CGPSINFO"));
  } else if (_type == FONA808_V1) {
    getReply(F("AT+CGPSINF="), x);
  } else {
   getReply(F("AT+CGNSINF"));
  }

  char *p = prog_char_strstr(replybuffer, (prog_char*)F("SINF"));
  if (p == 0) {
    buffer[0] = 0;
    return 0;
  }
  p+=6;
  uint8_t len = max(maxbuff-1, strlen(p));
  strncpy(buffer, p, len);
  buffer[len] = 0;
  readline(); // eat 'OK'
  return len;
}

boolean Adafruit_FONA::enableGPSNMEA(uint8_t i) {

  char sendbuff[15] = "AT+CGPSOUT=000";
  sendbuff[11] = (i / 100) + '0';
  i %= 100;
  sendbuff[12] = (i / 10) + '0';
  i %= 10;
  sendbuff[13] = i + '0';

  if (_type == FONA808_V2) {
    if (i)
      return sendCheckReply(F("AT+CGNSTST=1"), ok_reply);
```

```
    else
      return sendCheckReply(F("AT+CGNSTST=0"), ok_reply);
  } else {
    return sendCheckReply(sendbuff, ok_reply, 2000);
  }
}

/********* LOW LEVEL *******************************************/

inline int Adafruit_FONA::available(void) {
  return mySerial->available();
}

inline size_t Adafruit_FONA::write(uint8_t x) {
  return mySerial->write(x);
}

inline int Adafruit_FONA::read(void) {
  return mySerial->read();
}

inline int Adafruit_FONA::peek(void) {
  return mySerial->peek();
}

inline void Adafruit_FONA::flush() {
  mySerial->flush();
}

void Adafruit_FONA::flushInput() {
    // Read all available serial input to flush pending data.
    uint16_t timeoutloop = 0;
    while (timeoutloop++ < 40) {
       while(available()) {
```

```
          read();
          timeoutloop = 0;  // If char was received reset the timer
        }
        delay(1);
    }
}

uint16_t Adafruit_FONA::readRaw(uint16_t b) {
  uint16_t idx = 0;

  while (b && (idx < sizeof(replybuffer)-1)) {
    if (mySerial->available()) {
      replybuffer[idx] = mySerial->read();
      idx++;
      b--;
    }
  }
  replybuffer[idx] = 0;

  return idx;
}

uint8_t Adafruit_FONA::readline(uint16_t timeout, boolean multiline) {
  uint16_t replyidx = 0;

  while (timeout--) {
    if (replyidx >= 254) {
      break;
    }

    while(mySerial->available()) {
      char c =  mySerial->read();
      if (c == '\r') continue;
      if (c == 0xA) {
```

```
      if (replyidx == 0) // the first 0x0A is ignored
        continue;

      if (!multiline) {
        timeout = 0; // the second 0x0A is the end of the line
        break;
      }
    }
    replybuffer[replyidx] = c;
    replyidx++;
  }

  if (timeout == 0) {
    break;
  }
  delay(1);
 }
 replybuffer[replyidx] = 0;  // null term
 return replyidx;
}

uint8_t Adafruit_FONA::getReply(char *send, uint16_t timeout) {
 flushInput();
 DEBUG_PRINT(F("\t---> ")); DEBUG_PRINTLN(send);
 mySerial->println(send);
 uint8_t l = readline(timeout);
 DEBUG_PRINT (F("\t<--- ")); DEBUG_PRINTLN(replybuffer);
 return l;
}

uint8_t Adafruit_FONA::getReply(FONAFlashStringPtr send, uint16_t timeout) {
 flushInput();
 DEBUG_PRINT(F("\t---> ")); DEBUG_PRINTLN(send);
 mySerial->println(send);
```

```
  uint8_t l = readline(timeout);
  DEBUG_PRINT (F("\t<--- ")); DEBUG_PRINTLN(replybuffer);
  return l;
}
// Send prefix, suffix, and newline. Return response (and also set replybuffer with response).
uint8_t Adafruit_FONA::getReply(FONAFlashStringPtr prefix, char *suffix, uint16_t timeout) {
  flushInput();
  DEBUG_PRINT(F("\t---> ")); DEBUG_PRINT(prefix); DEBUG_PRINTLN(suffix);
  mySerial->print(prefix);
  mySerial->println(suffix);
  uint8_t l = readline(timeout);
  DEBUG_PRINT (F("\t<--- ")); DEBUG_PRINTLN(replybuffer);
  return l;
}
// Send prefix, suffix, and newline. Return response (and also set replybuffer with response).
uint8_t Adafruit_FONA::getReply(FONAFlashStringPtr prefix, int32_t suffix, uint16_t timeout) {
  flushInput();
  DEBUG_PRINT(F("\t---> ")); DEBUG_PRINT(prefix); DEBUG_PRINTLN(suffix, DEC);
  mySerial->print(prefix);
  mySerial->println(suffix, DEC);
  uint8_t l = readline(timeout);
  DEBUG_PRINT (F("\t<--- ")); DEBUG_PRINTLN(replybuffer);
  return l;
}


// Send prefix, suffix, suffix2, and newline. Return response (and also set replybuffer with response).
uint8_t Adafruit_FONA::getReply(FONAFlashStringPtr prefix, int32_t suffix1, int32_t suffix2, uint16_t timeout) {


  flushInput();
```

```
  DEBUG_PRINT(F("\t---> ")); DEBUG_PRINT(prefix);
  DEBUG_PRINT(suffix1, DEC); DEBUG_PRINT(','); DEBUG_PRINTLN(suffix2, DEC);
  mySerial->print(prefix);
  mySerial->print(suffix1);
  mySerial->print(',');
  mySerial->println(suffix2, DEC);

  uint8_t l = readline(timeout);

  DEBUG_PRINT (F("\t<--- ")); DEBUG_PRINTLN(replybuffer);
  return l;
}

// Send prefix, ", suffix, ", and newline. Return response (and also set replybuffer with
response).
uint8_t Adafruit_FONA::getReplyQuoted(FONAFlashStringPtr prefix,
FONAFlashStringPtr suffix, uint16_t timeout) {

  flushInput();

  DEBUG_PRINT(F("\t---> ")); DEBUG_PRINT(prefix);
  DEBUG_PRINT('"'); DEBUG_PRINT(suffix); DEBUG_PRINTLN('"');
  mySerial->print(prefix);
  mySerial->print('"');
  mySerial->print(suffix);
  mySerial->println('"');

  uint8_t l = readline(timeout);

  DEBUG_PRINT (F("\t<--- ")); DEBUG_PRINTLN(replybuffer);
  return l;
}

boolean Adafruit_FONA::sendCheckReply(char *send, char *reply, uint16_t timeout) {
```

```
  if (! getReply(send, timeout) )
  return false;
  return (strcmp(replybuffer, reply) == 0);
}

boolean Adafruit_FONA::sendCheckReply(FONAFlashStringPtr send,
FONAFlashStringPtr reply, uint16_t timeout) {
if (! getReply(send, timeout) )
return false;
  return (prog_char_strcmp(replybuffer, (prog_char*)reply) == 0);
}

boolean Adafruit_FONA::sendCheckReply(char* send, FONAFlashStringPtr reply,
uint16_t timeout) {
  if (! getReply(send, timeout) )
  return false;
  return (prog_char_strcmp(replybuffer, (prog_char*)reply) == 0);
}

// Send prefix, suffix, and newline.  Verify FONA response matches reply parameter.
boolean Adafruit_FONA::sendCheckReply(FONAFlashStringPtr prefix, char *suffix,
FONAFlashStringPtr reply, uint16_t timeout) {

  getReply(prefix, suffix, timeout);
  return (prog_char_strcmp(replybuffer, (prog_char*)reply) == 0);
}

// Send prefix, suffix, and newline.  Verify FONA response matches reply parameter.
boolean Adafruit_FONA::sendCheckReply(FONAFlashStringPtr prefix, int32_t suffix,
FONAFlashStringPtr reply, uint16_t timeout) {

  getReply(prefix, suffix, timeout);
  return (prog_char_strcmp(replybuffer, (prog_char*)reply) == 0);
}
```

```
// Send prefix, suffix, suffix2, and newline.  Verify FONA response matches reply
parameter.
boolean Adafruit_FONA::sendCheckReply(FONAFlashStringPtr prefix, int32_t suffix1,
int32_t suffix2, FONAFlashStringPtr reply, uint16_t timeout) {

  getReply(prefix, suffix1, suffix2, timeout);
  return (prog_char_strcmp(replybuffer, (prog_char*)reply) == 0);
}

// Send prefix, '', suffix, '', and newline.  Verify FONA response matches reply
parameter.
boolean Adafruit_FONA::sendCheckReplyQuoted(FONAFlashStringPtr prefix,
FONAFlashStringPtr suffix, FONAFlashStringPtr reply, uint16_t timeout) {

  getReplyQuoted(prefix, suffix, timeout);
  return (prog_char_strcmp(replybuffer, (prog_char*)reply) == 0);
}
boolean Adafruit_FONA::parseReply(FONAFlashStringPtr toreply,
        uint16_t *v, char divider, uint8_t index) {
  char *p = prog_char_strstr(replybuffer, (prog_char*)toreply);  // get the pointer to
the voltage

  if (p == 0) return false;
  p+=prog_char_strlen((prog_char*)toreply);
  for (uint8_t i=0; i<index;i++) { // increment dividers
    p = strchr(p, divider);
    if (!p) return false;
    p++;
     }
  *v = atoi(p);
  return true;
}

boolean Adafruit_FONA::parseReply(FONAFlashStringPtr toreply,
```

```
        char *v, char divider, uint8_t index) {
  uint8_t i=0;
  char *p = prog_char_strstr(replybuffer, (prog_char*)toreply);
  if (p == 0) return false;
  p+=prog_char_strlen((prog_char*)toreply);

  for (i=0; i<index;i++) {  // increment dividers
    p = strchr(p, divider);
    if (!p) return false;
    p++;
  }

  for(i=0; i<strlen(p);i++) {
    if(p[i] == divider)
      break;
    v[i] = p[i];
  }

  v[i] = '\0';
  return true;
}

boolean Adafruit_FONA::parseReplyQuoted(FONAFlashStringPtr toreply,
        char *v, int maxlen, char divider, uint8_t index) {
  uint8_t i=0, j;  // Verify response starts with toreply.
  char *p = prog_char_strstr(replybuffer, (prog_char*)toreply);
  if (p == 0) return false;
  p+=prog_char_strlen((prog_char*)toreply);// Find location of desired response field.
  for (i=0; i<index;i++) {// increment dividers
    p = strchr(p, divider);
    if (!p) return false;
    p++;
  }
```

```
  // Copy characters from response field into result string.
  for(i=0, j=0; j<maxlen && i<strlen(p); ++i) {
    // Stop if a divier is found.
    if(p[i] == divider)
      break;
    // Skip any quotation marks.
    else if(p[i] == '"')
      continue;
    v[j++] = p[i];
  }
  // Add a null terminator if result string buffer was not filled.
  if (j < maxlen)
    v[j] = '\0';
  return true;
}

boolean Adafruit_FONA::sendParseReply(FONAFlashStringPtr tosend,
     FONAFlashStringPtr toreply,
     uint16_t *v, char divider, uint8_t index) {
  getReply(tosend);

  if (! parseReply(toreply, v, divider, index)) return false;
  readline(); // eat 'OK'
  return true;
}
// needed for CBC and others
boolean Adafruit_FONA_3G::sendParseReply(FONAFlashStringPtr tosend,
     FONAFlashStringPtr toreply,
     float *f, char divider, uint8_t index) {
  getReply(tosend);

  if (! parseReply(toreply, f, divider, index)) return false;
  readline(); // eat 'OK'
  return true;
```

```
}


boolean Adafruit_FONA_3G::parseReply(FONAFlashStringPtr toreply,
        float *f, char divider, uint8_t index) {
  char *p = prog_char_strstr(replybuffer, (prog_char*)toreply);  // get the pointer to the voltage
  if (p == 0) return false;
  p+=prog_char_strlen((prog_char*)toreply);

  for (uint8_t i=0; i<index;i++) { // increment dividers
    p = strchr(p, divider);
    if (!p) return false;
    p++;
  }
  *f = atof(p);

  return true;
}
```

- H Dosyası:

```
#ifndef ADAFRUIT_FONA_H
#define ADAFRUIT_FONA_H

#include "includes/FONAConfig.h"
#include "includes/FONAExtIncludes.h"
#include "includes/platform/FONAPlatform.h"

#define FONA800L 1
#define FONA800H 6

#define FONA808_V1 2
#define FONA808_V2 3

#define FONA3G_A 4
#define FONA3G_E 5

// Uncomment to changed the preferred SMS storage
//#define FONA_PREF_SMS_STORAGE "SM"

#define FONA_HEADSETAUDIO 0
#define FONA_EXTAUDIO 1

#define FONA_STTONE_DIALTONE 1
#define FONA_STTONE_BUSY 2
#define FONA_STTONE_CONGESTION 3
#define FONA_STTONE_PATHACK 4
#define FONA_STTONE_DROPPED 5
#define FONA_STTONE_ERROR 6
#define FONA_STTONE_CALLWAIT 7
#define FONA_STTONE_RINGING 8
#define FONA_STTONE_BEEP 16
#define FONA_STTONE_POSTONE 17
```

```
#define FONA_STTONE_ERRTONE 18
#define FONA_STTONE_INDIANDIALTONE 19
#define FONA_STTONE_USADIALTONE 20

#define FONA_DEFAULT_TIMEOUT_MS 500

#define FONA_HTTP_GET   0
#define FONA_HTTP_POST  1
#define FONA_HTTP_HEAD  2

#define FONA_CALL_READY 0
#define FONA_CALL_FAILED 1
#define FONA_CALL_UNKNOWN 2
#define FONA_CALL_RINGING 3
#define FONA_CALL_INPROGRESS 4

class Adafruit_FONA : public FONAStreamType {
 public:
  Adafruit_FONA(int8_t r);
  boolean begin(FONAStreamType &port);
  uint8_t type();

  // Stream
  int available(void);
  size_t write(uint8_t x);
  int read(void);
  int peek(void);
  void flush();

  // FONA 3G requirements
  boolean setBaudrate(uint16_t baud);

  // RTC
  boolean enableRTC(uint8_t i);
```

```
  boolean readRTC(uint8_t *year, uint8_t *month, uint8_t *date, uint8_t *hr, uint8_t
*min, uint8_t *sec);

// SIM query
  uint8_t unlockSIM(char *pin);
  uint8_t getSIMCCID(char *ccid);
  uint8_t getNetworkStatus(void);
  uint8_t getRSSI(void);

  // IMEI
  uint8_t getIMEI(char *imei);

  // SMS handling
  boolean setSMSInterrupt(uint8_t i);
  uint8_t getSMSInterrupt(void);
  int8_t getNumSMS(void);
  boolean readSMS(uint8_t i, char *smsbuff, uint16_t max, uint16_t *readsize);
  boolean sendSMS(char *smsaddr, char *smsmsg);
  boolean deleteSMS(uint8_t i);
  boolean getSMSSender(uint8_t i, char *sender, int senderlen);
  boolean sendUSSD(char *ussdmsg, char *ussdbuff, uint16_t maxlen, uint16_t
*readlen);

  // Time
  boolean enableNetworkTimeSync(boolean onoff);
  boolean enableNTPTimeSync(boolean onoff, FONAFlashStringPtr ntpserver=0);
  boolean getTime(char *buff, uint16_t maxlen);

  // GPS handling
  boolean enableGPS(boolean onoff);
  int8_t GPSstatus(void);
  uint8_t getGPS(uint8_t arg, char *buffer, uint8_t maxbuff);
  boolean getGPS(float *lat, float *lon, float *speed_kph=0, float *heading=0, float
*altitude=0);
  boolean enableGPSNMEA(uint8_t nmea);
```

```
  // TCP raw connections
  boolean TCPconnect(char *server, uint16_t port);
  boolean TCPclose(void);
  boolean TCPconnected(void);
  boolean TCPsend(char *packet, uint8_t len);
  uint16_t TCPavailable(void);
  uint16_t TCPread(uint8_t *buff, uint8_t len);

  // HTTP low level interface (maps directly to SIM800 commands).
  boolean HTTP_init();
  boolean HTTP_term();
  void HTTP_para_start(FONAFlashStringPtr parameter, boolean quoted = true);
  boolean HTTP_para_end(boolean quoted = true);
  boolean HTTP_para(FONAFlashStringPtr parameter, const char *value);
  boolean HTTP_para(FONAFlashStringPtr parameter, FONAFlashStringPtr value);
  boolean HTTP_para(FONAFlashStringPtr parameter, int32_t value);
  boolean HTTP_data(uint32_t size, uint32_t maxTime=10000);
  boolean HTTP_action(uint8_t method, uint16_t *status, uint16_t *datalen, int32_t
timeout = 10000);
  boolean HTTP_readall(uint16_t *datalen);
  boolean HTTP_ssl(boolean onoff);

  // HTTP high level interface (easier to use, less flexible).
  boolean HTTP_GET_start(char *url, uint16_t *status, uint16_t *datalen);
  void HTTP_GET_end(void);
  boolean HTTP_POST_start(char *url, FONAFlashStringPtr contenttype, const uint8_t
*postdata, uint16_t postdatalen,  uint16_t *status, uint16_t *datalen);
  void HTTP_POST_end(void);
  void setUserAgent(FONAFlashStringPtr useragent);

  // HTTPS
  void setHTTPSRedirect(boolean onoff);
```

```
  // PWM (buzzer)
  boolean setPWM(uint16_t period, uint8_t duty = 50);


  // Helper functions to verify responses.
  boolean expectReply(FONAFlashStringPtr reply, uint16_t timeout = 10000);
  boolean sendCheckReply(char *send, char *reply, uint16_t timeout =
FONA_DEFAULT_TIMEOUT_MS);
  boolean sendCheckReply(FONAFlashStringPtr send, FONAFlashStringPtr reply,
uint16_t timeout = FONA_DEFAULT_TIMEOUT_MS);
  boolean sendCheckReply(char* send, FONAFlashStringPtr reply, uint16_t timeout =
FONA_DEFAULT_TIMEOUT_MS);

 protected:
  int8_t _rstpin;
  uint8_t _type;

  char replybuffer[255];
  FONAFlashStringPtr apn;
  FONAFlashStringPtr apnusername;
  FONAFlashStringPtr apnpassword;
  boolean httpsredirect;
  FONAFlashStringPtr useragent;
  FONAFlashStringPtr ok_reply;

  // HTTP helpers
  boolean HTTP_setup(char *url);

  void flushInput();
  uint16_t readRaw(uint16_t b);
  uint8_t readline(uint16_t timeout = FONA_DEFAULT_TIMEOUT_MS, boolean
multiline = false);
  uint8_t getReply(char *send, uint16_t timeout = FONA_DEFAULT_TIMEOUT_MS);
  uint8_t getReply(FONAFlashStringPtr send, uint16_t timeout =
FONA_DEFAULT_TIMEOUT_MS);
```

```
  uint8_t getReply(FONAFlashStringPtr prefix, char *suffix, uint16_t timeout =
FONA_DEFAULT_TIMEOUT_MS);
  uint8_t getReply(FONAFlashStringPtr prefix, int32_t suffix, uint16_t timeout =
FONA_DEFAULT_TIMEOUT_MS);
  uint8_t getReply(FONAFlashStringPtr prefix, int32_t suffix1, int32_t suffix2, uint16_t
timeout); // Don't set default value or else function call is ambiguous.
  uint8_t getReplyQuoted(FONAFlashStringPtr prefix, FONAFlashStringPtr suffix,
uint16_t timeout = FONA_DEFAULT_TIMEOUT_MS);

  boolean sendCheckReply(FONAFlashStringPtr prefix, char *suffix, FONAFlashStringPtr
reply, uint16_t timeout = FONA_DEFAULT_TIMEOUT_MS);
  boolean sendCheckReply(FONAFlashStringPtr prefix, int32_t suffix,
FONAFlashStringPtr reply, uint16_t timeout = FONA_DEFAULT_TIMEOUT_MS);
  boolean sendCheckReply(FONAFlashStringPtr prefix, int32_t suffix, int32_t suffix2,
FONAFlashStringPtr reply, uint16_t timeout = FONA_DEFAULT_TIMEOUT_MS);
  boolean sendCheckReplyQuoted(FONAFlashStringPtr prefix, FONAFlashStringPtr
suffix, FONAFlashStringPtr reply, uint16_t timeout = FONA_DEFAULT_TIMEOUT_MS);


  boolean parseReply(FONAFlashStringPtr toreply,
        uint16_t *v, char divider  = ',', uint8_t index=0);
  boolean parseReply(FONAFlashStringPtr toreply,
        char *v, char divider  = ',', uint8_t index=0);
  boolean parseReplyQuoted(FONAFlashStringPtr toreply,
        char *v, int maxlen, char divider, uint8_t index);

  boolean sendParseReply(FONAFlashStringPtr tosend,
      FONAFlashStringPtr toreply,
      uint16_t *v, char divider = ',', uint8_t index=0);

  static boolean _incomingCall;
  static void onIncomingCall();

  FONAStreamType *mySerial;
};
```

```
class Adafruit_FONA_3G : public Adafruit_FONA {

 public:
  Adafruit_FONA_3G (int8_t r) : Adafruit_FONA(r) { _type = FONA3G_A; }

    boolean getBattVoltage(uint16_t *v);
    boolean playToolkitTone(uint8_t t, uint16_t len);
    boolean hangUp(void);
    boolean pickUp(void);
    boolean enableGPRS(boolean onoff);
    boolean enableGPS(boolean onoff);

 protected:
    boolean parseReply(FONAFlashStringPtr toreply,
       float *f, char divider, uint8_t index);

    boolean sendParseReply(FONAFlashStringPtr tosend,
   FONAFlashStringPtr toreply,
   float *f, char divider = ',', uint8_t index=0);
};

#endif
```

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Atılay ARSLANCAN

Doğum Tarihi ve Yeri : 09.12.1988, Balıkesir

Yabancı Dili : İngilizce

E-posta : atilayarslancan@gmail.com

ÖĞRENİM DURUMU

Derece	**Alan**	**Okul/Üniversite**	**Mezuniyet Yılı**
Yüksek Lisans	Elektronik	Yıldız Teknik Üniversitesi	2017
Lisans	Elektrik ve Elektronik Mühendisliği	FMV Işık Üniversitesi	2011
Lise	Fen Bilimleri	Sırrı Yırcalı Anadolu Lisesi	2006

İŞ TECRÜBESİ

Yıl	**Firma/Kurum**	**Görevi**
2012-2017	Deniz Hava Komutanlığı	Sistem Geliştirme Müdürü
2011-2012	Makersan Makina Otomotiv San. Tic. Ltd. Şti.	Ar-Ge Mühendisi
2010-2011	Tuzeks Endüstri Tesisleri Sanayi ve Ticaret A.Ş.	Proje Asistanı