



بسم الله الرحمن الرحيم

Sudan University of Science and Technology



Faculty of Engineering

Aeronautical Engineering Department

Automatic Dependent Surveillance – Broadcast Transmitter Analysis (Message Generation)

**Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of Bachelor of Science. (BSc Honor)**

By:

1. Abeer Osman Abdallah
2. Ebtahaj Dafallah Ahmed Mohammed
3. Hajer Mohammed Gasimalla.
4. May Mohieldin Mohamed

Supervised By:

Dr. Omer Abdel Razag Sharif

October, 2016

الآية

بسم الله الرحمن الرحيم

قال تعالى: (اللهُ نُورُ السَّمَاوَاتِ وَالْأَرْضِ مِثْلُ نُورِهِ كَمِشْكَاةٍ فِيهَا مِصْبَاحٌ الْمِصْبَاحُ فِي زُجَاجَةٍ
الزُّجَاجَةُ كَأَنَّهَا كَوْكَبٌ دُرِّيٌّ يُوقَدُ مِنْ شَجَرَةٍ مُبَارَكَةٍ زَيْتُونَةٍ لَا شَرْقِيَّةٍ وَلَا غَرْبِيَّةٍ يَكَادُ زَيْتُهَا
يُضِيءُ وَلَوْ لَمْ تَمْسَسْهُ نَارٌ نُورٌ عَلَى نُورٍ يَهْدِي اللهُ لِنُورِهِ مَنْ يَشَاءُ وَيَضْرِبُ اللهُ الْأَمْثَالَ لِلنَّاسِ
وَاللَّهُ بِكُلِّ شَيْءٍ عَلِيمٌ)

[سورة النور:35]

ABSTRACT

This project deals with the acquaint for new aircraft surveillance system: Automatic Dependent Surveillance-Broadcast (ADS-B), which is a technology that can replace early navigation systems which cannot provide surveillance to all area, and may report false target and position as well as poor resolution. ADS-B enhances cockpit situational awareness, and has the potential to enable procedures not possible with current surveillance technology that would increase the capacity of the National Airspace System (ANS) it is being introduced by the Federal Aviation Administration (FAA) with mandated implementation in the United States by the year 2020.

In this project, a successful microcontroller-based method to generate ADS-B messages was conducted using a system comprised of GPS module (Sky Nav SKM-53), Arduino Uno and 16*2 Liquid Crystal Display unit. The proposed system was firstly simulated using Proteus ISIS and then partially implemented. The obtained results from the simulation platform were compared with the real and calculated one which are similar.

التجريد

سيتناول هذا المشروع تعريف عن نظام جديد لمراقبة الطائرات وهو نظام البث الاستطلاعي المستقل التلقائي . هذه التقنية يمكن أن تحل محل أنظمة الملاحة القديمة التي لا تستطيع ان توفر المراقبة لجميع المناطق، كما يمكن أن تقدم تقريرا عن أهداف ومواقع زائفة وضعف القرار. النظام يزيد من الوعي الظرفي في قمر القيادة، كما أن لديه القدرة على تمكين الإجراءات الغير ممكنة مع تكنولوجيا المراقبة الحالية التي من شأنها أن تزيد من قدرة نظام المجال الجوي الوطني. اقترح هذا النظام بصورة رسمية من قبل إدارة الطيران الفيدرالية الامريكية . مع تكليف تنفيذ في الولايات المتحدة بحلول عام 2020

يتناول هذا المشروع طريقة ناجحة قائمة على متحكم لتوليد رسائل النظام باستخدام نظام يتالف من وحدة نظام تحديد المواقع (سكاي التنقل SKM-53)، اردوينو أونو و وحدة العرض شاشة الكريستال السائل 16*2 تم محاكاة النظام المقترح أولا باستخدام Protues ISIS ثم نفذت جزئيا. وتمت مقارنة النتائج المتحصل عليها من منصة المحاكاة مع الحقيقي وتم حساب التي تتشابه..

ACKNOWLEDGEMENT

Praise be to Allah for reconciling us to complete this project. Moreover, project team would like to express a special thanks and gratitude to our supervisors who are gave us the golden opportunity to make this project true. In addition, it is pleasure to thankful those who have share up to achievement of this project: our teachers who support us with require knowledge, Institute of Space Research and Aeronautic Centre (ISRA) for true support and technical assistant, our colleague who provided expertise that greatly assisted in completion the project.

Last but not least, we would like to thank our parents, families, and friends for their unlimited support and motivation.

DEDICATION

To our families with Love and respect

TABLE OF CONSTANCE

الآية	II
ABSTRACT.....	III
التجريد	IV
ACKNOWLEDGEMENT	V
DEDICATION.....	VI
TABLE OF CONSTANCE.....	VII
LIST OF FIGURE.....	IX
LIST OF TABLE	XI
ABBREVIATIONS	XII
CHAPTER ONE.....	1
INTRODUCTION	1
1.1 Overview.....	1
1.2 Aim and Objectives.....	3
1.3 Problem Statement	3
1.4 Proposed Solution	3
1.5 Methodology.....	3
1.6 Thesis Out line	4
CHAPTER TWO	5
Navigation and ADS-B systems	5
2.1 Introduction.....	5
2.1.1 Air Traffic Control (ATC)	5
2.1.2 Radar	5
2.1.3 Mode S	8
2.1.4 Traffic Alert and Collision Avoidance System (TCAS).....	10
2.1.5 The Global Positioning System (GPS)	11
2.1.6 Automatic Dependent Surveillance (ADS).....	16
2.2 ADS-B Message generation techniques.....	23
CHAPTER THREE	24
Proposed ADS-B Message Generation System	24

3.1 Introduction.....	24
3.2 GPS Module (SKM -53).....	24
3.3 Arduino UNO.....	26
3.4 Liquid Crystal Display 16*2.....	28
3.5 Proteus Design Suite8.0.....	28
3.5.1Proteus simulation for GPS with Arduino	29
3.5.2 Proteus simulation for LCD with Arduino.....	29
3.5.3 Proteus simulation for LCD with Arduino and GPS	30
CHAPTER FOUR.....	32
ADS-B Message Generator.....	32
4.1 Introduction.....	32
4.1.1 Down-Link Format (DF) Type	33
4.1.2 Aircraft Identification Number (ID) Generation.....	33
4.1.3 Altitude Encoding	34
4.1.4 Latitude and Longitude Encoding.....	36
4.1.5 Cyclic Redundancy Check (CRC) Generation	41
4.2Results.....	42
4.2.1GPS output from proteus simulation.....	42
4.2.2The Aircraft Identification Encoding.....	43
4.2.3 The Altitude encoding.....	44
4.2.4 The latitude encoding.....	45
4.2.5 The longitude encoding.....	48
4.2.6 Cyclic Redundancy Check for odd message.....	49
4.2.7 The complete Message.....	51
CHAPTER FIVE	52
CONCLUSION AND RECOMMENDATIONS	52
5.1 Introduction.....	52
5.2 Conclusion	52
5.3 Recommendations.....	52
5.4 Future work.....	52
Reference	53

LIST OF FIGURE

Figure 1.1	Automatic Dependent Surveillance- Broadcast System.....	2
Figure 2.1	Graphical representation of SSR&PSR.....	6
Figure 2.2	Primary radar	7
Figure 2.3	Secondary radar	8
Figure 2.4	Mode S interrogation.....	9
Figure 2.5	Mode S- replay	10
Figure 2.6	Global position system- satellites	11
Figure 2.7	Global navigation satellites system	13
Figure 2.8	GPS constellation	15
Figure 2.9	Automatic Dependent Surveillance -contract.....	18
Figure 2.10	Major components of the ADS-B system.....	20
Figure 2.11	Broadcasting of ADS-B Squitter message	21
Figure 2.12	DF11 Frame.....	21
Figure 2.13	DF 17 Frame (ASD-B filed message)	21
Figure 2.14	ADS-B OUT and IN.....	22
Figure 3.1	ADS-B Message Generation component.....	24
Figure 3.2	Arduino Uno structures	27
Figure 3.3	Liquid Crystal Display 16*2	28
Figure 3.4	Proteus simulation for GPS with Arduino.....	29
Figure 3.5	Proteus simulation for LCD with Arduino	30
Figure 3.6	Proteus hardware simulation for LCD with Arduino and GPS	31
Figure 4.1	Latitude and Longitude Measurements	32
Figure 4.2	Aircraft Identification Number (ID) Generation	33
Figure 4.3	Arduino code for Aircraft Identification Number (ID) Generation.....	33
Figure 4.4	Altitude Encoding.....	35
Figure 4.5	Arduino code for Generate the Altitude value	36
Figure 4.6	Latitude Encoding	37
Figure 4.7	Arduino code for Generate the Latitude value (Odd message)	38
Figure 4.8	Arduino code for Generate the Latitude value (even message)	38
Figure 4.9	Longitude Encoding	39
Figure 4.10	Arduino code to Generate the Longitude value (Odd message).....	40
Figure 4.11	Arduino code to Generate the Longitude value (even message).....	40
Figure 4.12	Arduino code for CRC.....	41
Figure 4.13	ADS-B message format. Modified from	41
Figure 4.14	GPS output from proteus simulation	42
Figure 4.15	The Aircraft identification.....	44
Figure 4.16	The altitude for odd and even message	45
Figure 4.17	The latitude for Odd message.....	45
Figure 4.18	The Rlat for odd messege.....	46
Figure 4.19	The Rlat for even messege	47

Figure 4.20	The longitude for odd message.....	48
Figure 4.21	The longitude for even message	49
Figure 4.22	Cyclic Redundancy Check for odd message	50
Figure 4.23	Cyclic Redundancy Check for even message.....	50
Figure 4.24	The complete odd message.....	51
Figure 4.25	The complete even message	51

LIST OF TABLE

Table 3-1 GPS Module (SKM -53) Pin Description.....	25
Table 3-2 NMEA-0183 Output Messages	26
Table 4-1 NMEA-0183 Output Messages Decoding.....	43

ABBREVIATIONS

FAA	Federal Aviation Administration
RADR	Radio Detection And Ranging
ADS-B	Automatic Dependent Surveillance-Broadcast
GPS	Global Positioning System
ATC	Air Traffic Control
VHF	Very High Frequency A
VDL	Voice Data Link
UAT	Universal Access transceiver
ATCRBS	Air Traffic Control Radar Beacon System
NMEA	National Marine Electronics Association
PPI	Plan Position Indicator
SSR	Secondary Surveillance Radar
PSR	Primary Surveillance Radar
DoD	Department of Defense
DPSK	Differential Phase Shift Keying
ASK	Amplitude Shift Keying
TACAS	Traffic Alert and Collision Avoidance System
ICAO	International Civil Aviation Organization
GNSS	Global Navigation Satellite System
EGPWS	Enhanced Ground Proximity Warning System
ADSC	Automatic Dependent Surveillance-Contract.
CDU	Control Display Unit
PPM	Pulse Position Modulation
FMS	Flight Management System
NRA	Non Radar Area
ATSA	Air Traffic Situational Awareness
SIS	Signal In Space
USRP	Universal Software Radio Peripheral

TIS-B	Traffic information service –broad cast
FIS-B	Flight information service –broad cast
LCD	Liquid Crystal Display
UTC	Universal Time Control
CRC	Cyclic Redundancy Check

CHAPTER ONE

INTRODUCTION

1.1 Overview

With the mechanics of flight secured, early aviators began the task of improving operational safety and functionality of flight these were developed in large part through the use of reliable communication and navigation system in an effort to increase the safety, efficiency and capacity of air transport operations. This develop of air navigation systems started from simple system that meet the needs to organized the motion of air transport at that time but due to increasing of number of aircraft in space and wide area of travelling so a new system was created to corresponded this.

The Federal Aviation Administration (FAA) proposes a comprehensive reform of the surveillance radar (Next Gen). For this, the FAA deploys a relatively new technology called Automatic Dependent Surveillance-Broadcast (ADS-B). This technology allows aircraft equipped with a Global positioning system (GPS) to periodically send their position and other information to ground stations and other aircraft equipped with ADS-B that are present in the area. Although the FAA expects keep the primary radar for defense, many of today's secondary surveillance radars will not be used in the future [1].

It is a cooperative surveillance technology in which an aircraft determines its position via satellite navigation and periodically broadcasts it, enabling it to be tracked. The information can be received by air traffic control ground stations as a replacement for secondary radar. It can also be received by other aircraft to provide situational awareness and allow salve separation. ADS-B indicate to: A: automatic in that it requires no pilot or external input, D: it is dependent in that it depends on data from the aircraft's navigation system, S: is for surveillance, because it provides radar -like surveillance services, much like radar, and B: is for broadcast, in the sense that it continuously broadcast aircraft position and other data to any aircraft or ground station equipped to receive ADS-B.

Furthermore, ADS-B consists of two different services: ADS-B Out, and ADS-B In. ADS-B Out provides a means of automated aircraft parameter transmission between the aircraft and the Air Traffic Control (ATC), and ADS-B In: provides automated aircraft parameter transmission

between the aircraft themselves. The system relies on two avionics components: a high-integrity Global Position System (GPS) navigation source and a data link (ADS-B unit), to determine the aircraft position, velocity, altitude, identification and other relevant information to potential ground station and other aircraft.

There are several types of certified ADS-B data links, but the most common ones operate at 1090 MHz, essentially a modified Mode S transponder, or at 978 MHz. The FAA would like to see aircraft that exclusively operate below 18,000 feet (5,500 m) use the 978 MHz link since this will help alleviate further congestion of the 1090 MHz frequency. To obtain ADS-B Out capability at 1090 MHz, one can install a new transponder or modify an existing transponder if the manufacturer offers an ADS-B upgrade, plus install a certified GPS position source if one is not already present.

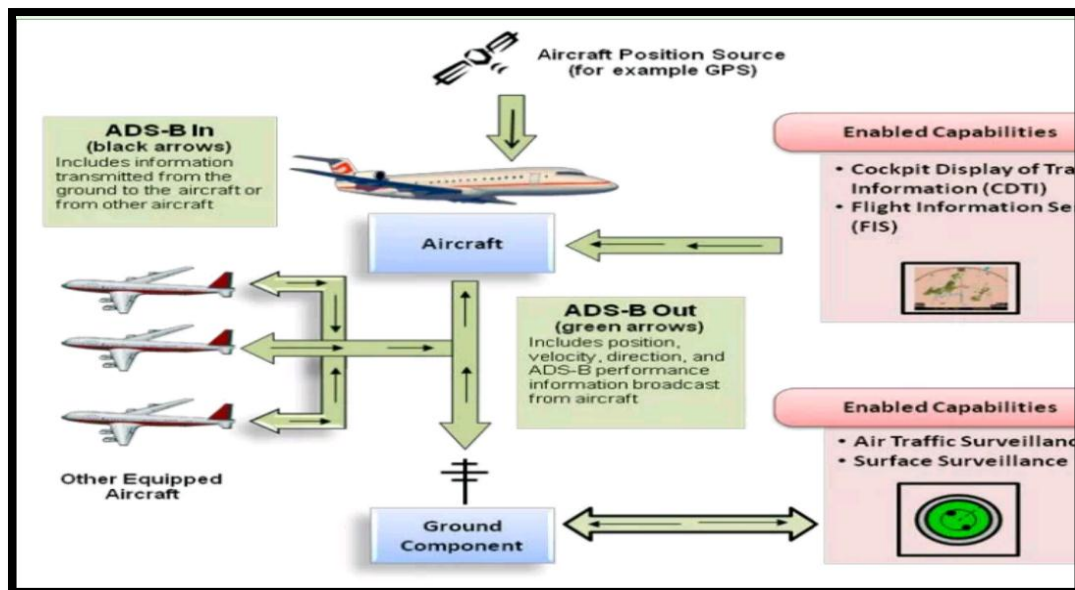


Figure 1.1 Automatic Dependent Surveillance- Broadcast System

The ADS-B system has three main components: ground infrastructure, airborne component and operating procedures. A transmitting subsystem that includes message generation and transmission functions at the source. The transport protocol: VHF (VDL mode 2 or 4), 1090ES, or 978 MHz Universal Access transceiver (UAT). A receiving subsystem that includes message reception and report assembly functions at the receiving destination [2].

1.2 Aim and Objectives

The main aim of this project is to generate ADS-B message depend on the data that received from the GPS. This aim includes the following objectives:

Proposed low-cost platform to generate ADS-B.

Develop a static ADS-B generator using real data.

1.3 Problem Statement

How extract message format from GPS and used it to construct the ADS-B message.

1.4 Proposed Solution

Use SKYLAB-SKM53 GPS module to received GPS signal, Use the Arduino Uno to make interface between the GPS and computer. the GPS signal(NMEA) then decoded to determine the altitude, longitude and UTC time of the located aircraft then Use this information to generated the ADS-B message.

1.5 Methodology

To achieve the project, aim and objectives, the following methodology was utilized:

1. Selecting a low-cost and suitable microcontroller-based platform. The Arduino Uno platform has selected where it is based on ATMEGA microcontroller, the register size is 32bits according to the defined variable, easily to interface with the other components such as Global Positioning System (GPS) and Liquid Crystal Display (16*2 LCD) through the available peripherals and ports. A micro C programming language was used to develop the functions and procedures of generating ADS-B message and connecting the GPS and LCD units.
2. The SKYNAV-SKM53(name of GPS module) was selected where is it cheap and easy to interface with Arduino Uno. GPS signal has received using SKYNAV-SKM53 GPS module through the Arduino Uno kit. The received GPS signal (NMEA) has decoded to determine the altitude, longitude and Universal Time Control (UTC) time of the located aircraft.
3. Through multiple encoding equations (following the ADS-B format), the data had been processed and the ADS-B message has generated.

4. To ensure a successful ADS-B message generation, all the processing stages have displayed on the LCD which is connected to the Arduino Uno platform and checked through conventional numerical calculations.

5. The proposed ADS-B message generation system was simulated using Proteus ISIS software and partially implemented.

1.6 Thesis Out line

In Chapter Two some literature was reviewed which includes a brief introduction about the navigation aids, and its classifications. Some literature about ADS-B generating message were reviewed. In Chapter Three, the proposed ADS-B system based on Arduino Uno was developed, where the system components, its connection and testing procedures were illustrated. The required calculations to generate ADS-B message were developed using micro C language, All ADS-B message generation stages were traced, depicted and discussed Chapter four. The conclusion and recommendations were drawn in Chapter Five.

CHAPTER TWO

Navigation and ADS-B systems

2.1 Introduction

To achieved better performances, organized air transport and ensure safety the navigation system had been passed through various development started from Air Traffic Control (ATC), Primary Surveillance Radar (PSR), Secondary Surveillance Radar (SSR), Mode Select (mode S) reached to Automatic Dependent Surveillance (ADS). In the following, these techniques will have introduced briefly.

2.1.1 Air Traffic Control (ATC)

It is a service provided by ground based controllers who direct aircraft on the ground and through controlled airspace. The primary purpose of it worldwide is to prevent collisions, organize and expedite the flow of air traffic and provide information and other support for pilot [3].

ATC is responsible for the separation and efficient movement of aircraft and vehicles operating on the taxiways and runways of the airport. Surveillance displays include a map of the area, position of the aircrafts and data include aircraft identification, speed, altitude and other information [4]. Traffic flow is broadly divided into departures arrivals and over flights. As aircraft move in and out of terminal airspace, they are handed off to the next appropriate control facility. terminal control is responsible for ensuring that aircraft are at an appropriate altitude when they are handed off, and that aircraft arrive a suitable rate for landing [3]. The problems faced by ATC are primarily related to the volume of the air traffic demand placed on the system and weather [5].

2.1.2 Radar

Radar refer to radio detection and ranging was developed during World War II. Radar is a technology which detects the range and azimuth of an aircraft based upon the difference in time between transmission of pulses to the aircraft and the receipt of energy from the aircraft. Typically, the technology uses a large rotating antenna and associated machinery works through the emission of electromagnetic (radio frequency) waves by a directional or rotating antenna dish. Those radio waves then reflect off an object and return back to the source where the signal is gathered by a receiver and the target is placed on the Plan Position Indicator (PPI), the display within the ATC

tower. A typical radar system is comprised of Secondary Surveillance Radar (SSR) and Primary Surveillance Radar(PSR)[5].

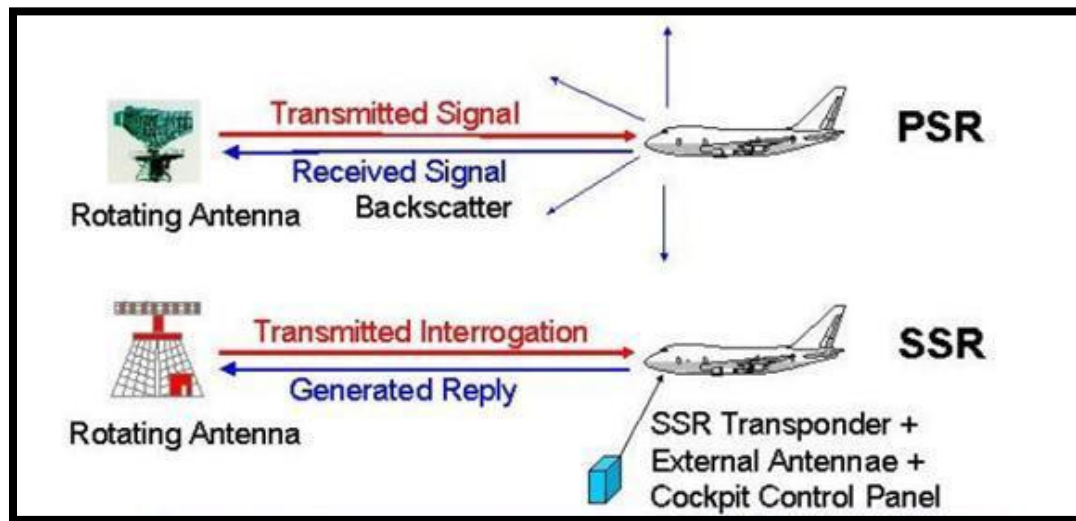


Figure 2.1 Graphical representation of SSR&PSR

2.1.2.1 primary surveillance radar (PSR)

These radars that it worked on the “Battle of Britain” principle in which the radar transmitter sends out a pulses in radio energy, of which small proration is reflected from the surface or structure of the target from the aircraft back to the radar receiver, the azimuth of the radar antenna provides the bearing of the aircraft from the ground station, and the time taken for the pulse to reach the target and return provides a measure of the distance of the target from the ground station.

The bearing and distance of the target can then be converted to a ground position for display to the air traffic controller. Target elevation (altitude) is not normally measured by ATC. Primary radar PSR has been augmented with Air Traffic Control Radar Beacon System (ATCRBS), also known as secondary surveillance radar (SSR).

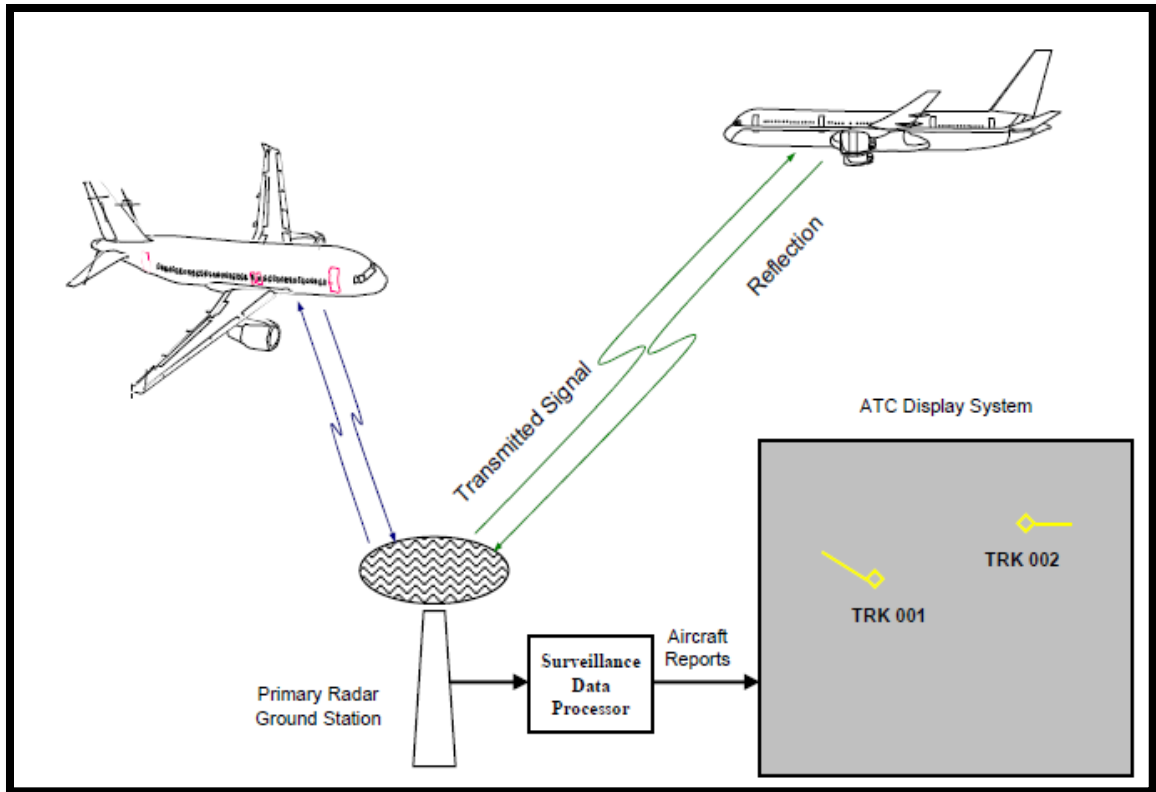


Figure 2.2 Primary radar

2.1.2.2 secondary surveillance radar (SSR)

The disadvantage of PSR outlined above led to the employment of another aspect of wartime radar development, this was the identification friend or foe (IFF) system, which had been developed as means of positively identifying friendly aircraft from enemy's employs special equipment onboard the target aircraft. Interrogation signals are sent from the ground station antenna just as with PSR. Once an aircraft receives the interrogation signal, a transponder aboard the aircraft returns a coded reply signal containing information such as aircraft identification and altitude. Because the aircraft transmits the reply message, SSR provides greater signal strength which translates to greater range and improved performance. SSR also decreases the power needed by the ground station transmission since the signal will not need to be reflected and thus providing considerable economy.

One drawback of SSR is its dependent nature, with dedicated equipment on the ground and on the aircraft. In most cases, PSR and SSR will be coupled together to accommodate both aircraft

with and without transponders. disadvantage of SSR that required a target A/C to Carry an operating transponder, thus SSR is dependent surveillance system for this PSR operating with SSR in certain area [6].

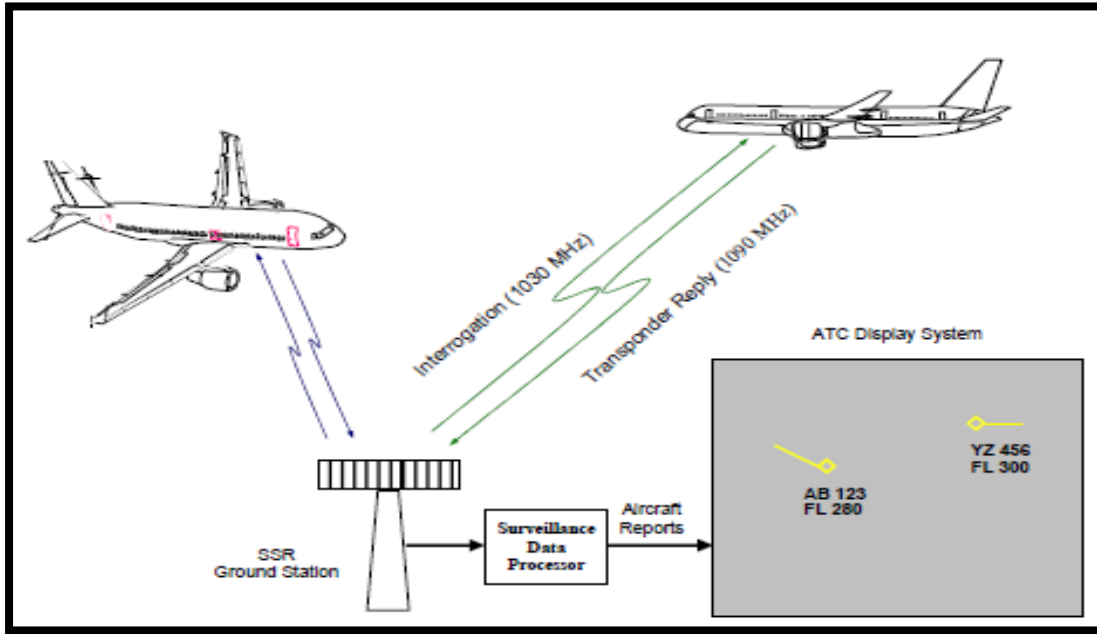


Figure 2. 3 Secondary radar

2.1.3 Mode S

The interrogation of the Mode S transponder is very similar to the interrogation of the Mode A/C transponders. Mode S requires two different types of interrogation an individual call and an all call, In the all call, the P1, P2, and P3 pulses are still present along with a new P4 pulse that begins 2 μ s after the leading edge of the P3 pulse. The P4 pulse lasts for a duration of 1.6 μ s and does not interfere with the interrogation of the Mode A/C transponders. If the P4 pulse is not present, the transponder knows that it must make its reply in the Mode A/C format .

The individual call again consists of the P1 and P2 pulses. The P2 pulse in this case has greater amplitude than the P1 pulse so that Mode A/C transponders will no longer attempt to decode the interrogation. The P2 pulse is followed by P6 pulse that is either 16.25 or 30.25 μ s in length. The P6 pulse contains data in Differential Phase Shift Keying (DPSK) format and either contains 56 or 112 bits of data along with a synchronization pulse. P5 pulse is transmitted in the difference beam of the interrogator and the pulse is generated so that it will coincide with the synchronization pulse in P6. By interfering with the synchronization bit, the P5 pulse keeps

aircraft in the side lobes of the transmission antenna from being able to synchronize and thus decode the DPSK signal

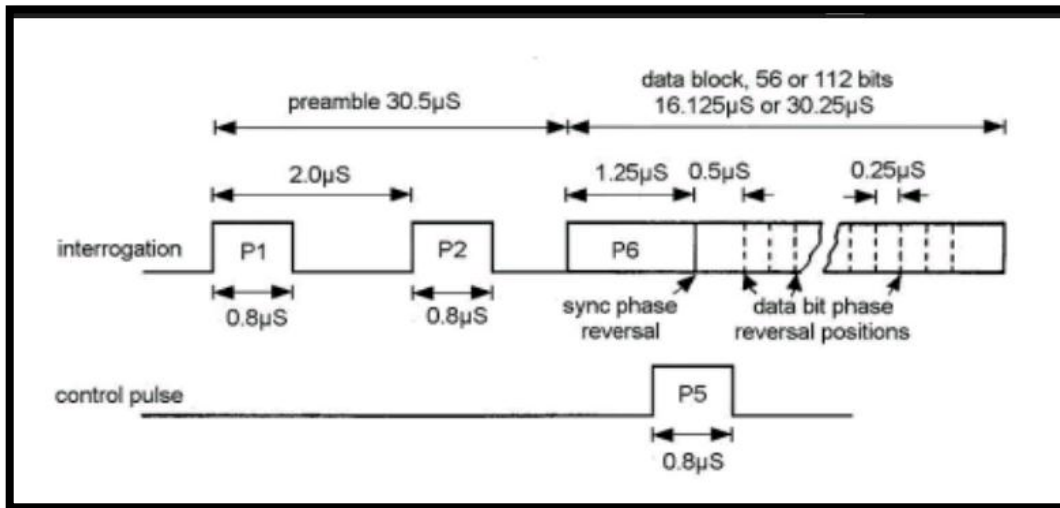


Figure 2.4 Mode S interrogation

- Mode S Reply

First the reply contains four Amplitude Shift Keying ASK bits that are spaced in such a way that no two overlapping Mode A/C responses could generate them. Following these preambles bits is a block of either 56 or 112 data pulses, depending on the reply that is requested. The data pulses are sent in ASK format with Manchester encoding so that each data pulse lasts 1.0 μs but each pulse is constructed of two 0.5 μs pulses, one high and one low.

This helps make the signal very resistant to noise interference and reduces the number of replies needed for Mode S to operate safely. Again the final 24 bits of each reply contain the aircraft's identification number and a parity check of the data bits so that errors caused by noise can be detected and false information is not generated.

The Mode S transponder is also designed to make transmissions without being interrogated. This transmission is called a "squitter." The squitter contains the shorter 56-bit transmission and is made to inform surrounding receivers of the aircraft's identification number. This is done so that individual calls can be made to that aircraft from that point forward.

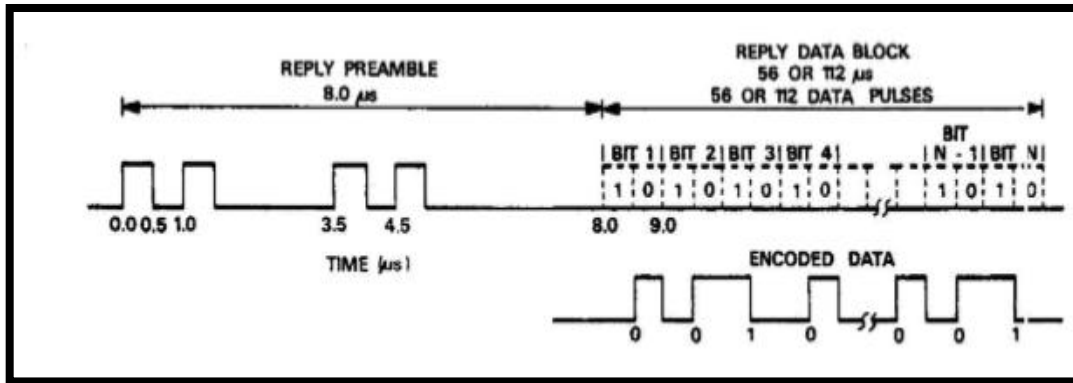


Figure 2.5 Mode S- replay

The Mode S transponder was designed to help reduce the congestion caused in the 1030 and 1090 MHz bands. This was needed because of potential saturation of the SSR systems around busy airports.[7]

2.1.4 Traffic Alert and Collision Avoidance System (TCAS)

The TCAS can provide pilots with airborne collision warning and avoidance information; however, aircraft pilots have not had access to a source of airborne information that can provide real-time continuous situational awareness. While TCAS systems are installed in commercial and corporate aircraft, general aviation has not seen a significant level of fleet equipage. Advances in aircraft avionics capabilities now make it possible for all properly equipped aircraft to broadcast certain information about its location, direction of travel, and other information to other aircraft in the vicinity that are similarly equipped. A reasonable predication is that the other aircraft can then use that information to determine the transmitting aircraft's altitude, position, and direction of flight in relation to their own, providing an immediate increase in the level of flight situational awareness [8].

TCAS modified the way in which TCAS interrogates nearby and distant targets, limiting the number of interrogations in a given area. However, in order to reduce the number of interrogations in a crowded environment, the range of the interrogations is limited to that necessary to avoid a collision. This limited range reduced the effectiveness of TCAS for general traffic situational awareness.

The Mode S transponder was developed as part of the TCAS system in the 1970s and is still an integral part of the TCAS system. ADS-B technology actually enhances the effectiveness of the TCAS system while still maintaining the independence of the TCAS safety backup. [9]

2.1.5 The Global Positioning System (GPS)

Is a space-based navigation system that provides location and time information in all weather conditions, anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites. The system provides critical capabilities to military, civil, and commercial users around the world. GPS provides continuous (24 hours/day), real-time, 3-dimensional positioning, navigation and timing worldwide.

The GPS system consists of three segments: The space segment: the GPS satellites themselves, the control system, operated by the U.S. military, and The user segment, which includes both military and civilian users and their GPS equipment.

- The GPS Constellation

The GPS satellites are powered primarily by sun-seeking solar panels, with NiCad batteries providing secondary power. On board each GPS satellite are four atomic clocks, only one of which is in use at a time. These highly accurate atomic clocks enable GPS to provide the most accurate timing system that exists. Satellite Orbits. There are four satellites in each of 6 orbital planes. Each plane is inclined 55 degrees relative to the equator, which means that satellites cross the equator tilted at a 55-degree angle. The system is designed to maintain full operational capability even if two of the 24 satellites fail.

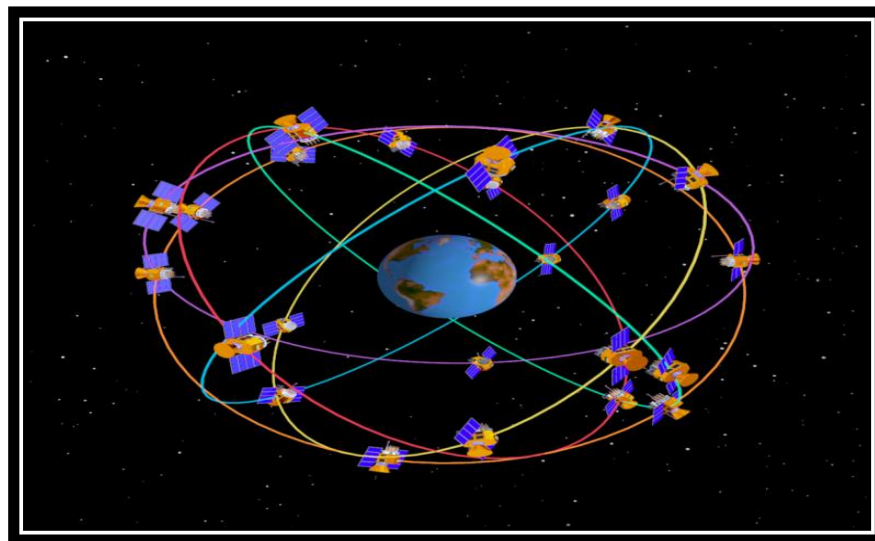


Figure 2.6 Global position system- satellites

GPS satellites complete an orbit in approximately 12 hours, which means that they pass over any point on the earth about twice a day. The satellites rise (and set) about four minutes earlier each day.

- *Satellite Signals*

GPS satellites continuously broadcast satellite position and timing data via radio signals on two frequencies (L1 and L2). The satellite signals require a direct line to GPS receivers and cannot penetrate water, soil, walls or other obstacles. Two kinds of code are broadcast on the L1 frequency (C/A code and P code). C/A (Coarse Acquisition) code is available to civilian GPS users and provides Standard Positioning Service (SPS). P (Precise) code is broadcast on both the L1 and L2 frequencies. P code, used for the Precise Positioning Service (PPS) is available only to the military. Using P code on both frequencies, a military receiver can achieve better accuracy than civilian receivers.

- Control Segment

The U.S. Department of Defense maintains a master control station at Falcon Air Force Base in Colorado Springs. There are four other monitor stations located in Hawaii, Ascension Island, Diego Garcia and Kwajalein. The Department of Defense (DoD) stations measure the satellite orbits precisely. Any discrepancies between predicted orbits and actual orbits are transmitted back to the satellites. The satellites can then broadcast these corrections, along with the other position and timing data, so that a GPS receiver on the earth can precisely establish the location of each satellite it is tracking.

- User Segment: Military and Civilian GPS Users

The U.S. military uses GPS for navigation, reconnaissance, and missile guidance systems. Civilian use of GPS developed at the same time as military uses were being established, and has expanded far beyond original expectations., however, fall into one of four categories: navigation, surveying, mapping and timing.

The User Segment consists of the receivers, processors, and antennas that allow land, sea, or airborne operators to receive the GPS satellite broadcasts and compute their precise position, velocity and time.

- GLONASS

The Russian government has developed a system, similar to GPS, called GLONASS. The full constellation consists of 24 satellites in 3 orbit planes, which have a 64.8-degree inclination to the earth's equator. The GLONASS system now consists of 12 healthy satellites. GLONASS uses the same code for each satellite and many frequencies, whereas GPS which uses two frequencies and a different code for each satellite. Some GPS receiver manufacturers have incorporated the capability to receive both GPS and GLONASS signals. This increases the availability of satellites and the integrity of combined system.

- GALILEO

Galileo is Europe's contribution to the next generation Global Navigation Satellite System (GNSS). Unlike GPS, which is funded by the public sector and operated by the U.S. Air Force, Galileo will be a civil controlled system that draws on both public and private sectors for funding. The service will be free at the point of use, but a range of chargeable services with additional features will also be offered. These additional features would include improved reception, accuracy and availability. Design of the Galileo system is being finalized and the delivery of initial services is targeted for 2008[10]

2.1.5.1 GPS – THE Principle Works

Satellite Navigation is based on a global network of satellites that transmit radio signals in medium earth orbit. Users of Satellite Navigation are most familiar with the 32 Global Positioning System (GPS) satellites. The United States, who developed and operates GPS, and Russia, who developed a similar system known as GLONASS, have offered free use of their respective systems to the international community. The International Civil Aviation Organization (ICAO), as well as other international user groups, have accepted GPS and GLONASS as the core for an international civil satellite navigation capability known as the Global Navigation Satellite System (GNSS).

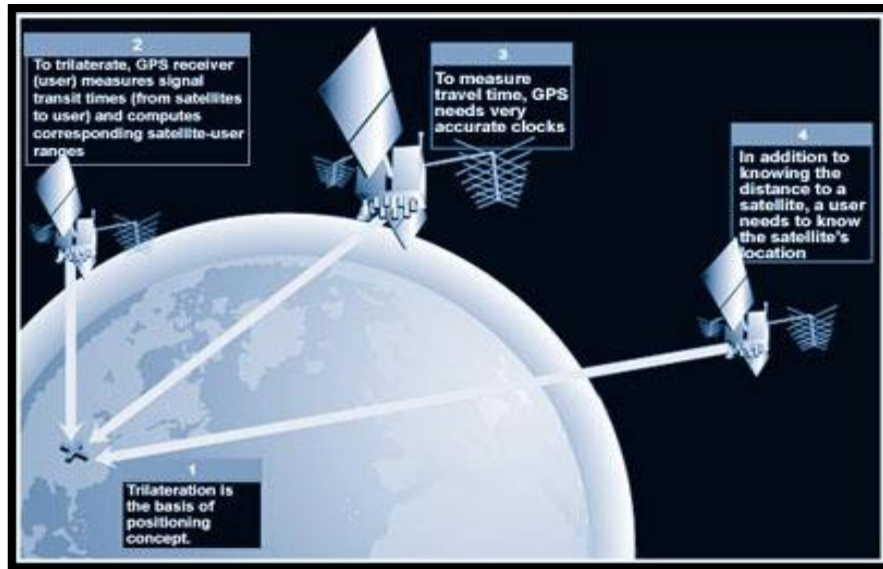


Figure 2.7 Global navigation satellites system

The basic GPS service provides users with approximately 7.8-meter accuracy, 95% of the time, anywhere on or near the surface of the earth. To accomplish this, each of the 32 satellites emits signals to receivers that determine their location by computing the difference between the time that a signal is sent and the time it is received. GPS satellites carry atomic clocks that provide extremely accurate time. The time information is placed in the codes broadcast by the satellite so that a receiver can continuously determine the time the signal was broadcast. The signal contains data that a receiver uses to compute the locations of the satellites and to make other adjustments needed for accurate positioning. The receiver uses the time difference between the time of signal reception and the broadcast time to compute the distance, or range, from the receiver to the satellite. The receiver must account for propagation delays, or decreases in the signal's speed caused by the ionosphere and the troposphere. With information about the ranges to three satellites and the location of the satellite when the signal was sent, the receiver can compute its own three-dimensional position. An atomic clock synchronized to GPS is required in order to compute ranges from these three signals. However, by taking a measurement from a fourth satellite, the receiver avoids the need for an atomic clock. Thus, the receiver uses four satellites to compute latitude, longitude, altitude, and time. [11]

aviators throughout the world use the Global Positioning System (GPS) to increase the safety and efficiency of flight. With its accurate, continuous, and global capabilities, GPS offers seamless satellite navigation services that satisfy many of the requirements for aviation users. Space-based

position and navigation enables three-dimensional position determination for all phases of flight from departure, en route, and arrival, to airport surface navigation.

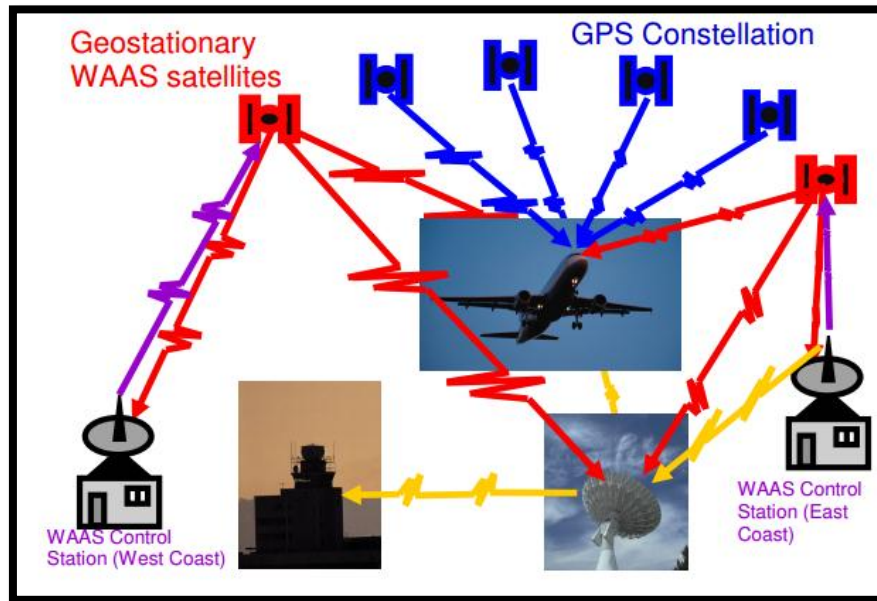


Figure 2.8 GPS constellation

The trend toward an Area Navigation concept means a greater role for GPS. Area Navigation allows aircraft to fly user-preferred routes from waypoint to waypoint, where waypoints do not depend on ground infrastructure. Procedures have been expanded to use GPS and augmented services for all phases of flight. This has been especially true in areas that lack suitable ground based navigation aids or surveillance equipment.

New and more efficient air routes made possible by GPS are continuing to expand. Vast savings in time and money are being realized. In many cases, aircraft flying over data-sparse areas such as oceans have been able to safely reduce their separation between one another, allowing more aircraft to fly more favorable and efficient routes, saving time, fuel, and increasing cargo revenue.

Improved approaches to airports, which significantly increase operational benefits and safety, are now being implemented even at remote locations where traditional ground-based services are unavailable. In some regions of the world, satellite signals are augmented, or improved for special aviation applications, such as landing planes during poor visibility conditions. In those cases, even greater precision operations are possible.

The good news for the aviation community is that GPS is being constantly improved and modernized. A main component of the ongoing civilian modernization effort is the addition of two new signals. These signals complement the existing civilian service. The first of these new signals is for general use in non-safety critical applications. The second new signal will be internationally protected for aviation navigational purposes. This additional safety-of-life civilian signal will make GPS an even more robust navigation service for many aviation applications.

Reliance on GPS as the foundation for today and tomorrow's air traffic management system is a major part of many national plans. Those aviation authorities that are moving forward with GPS have observed and documented reductions in flight time, workload, and operating costs for both the airspace user and service provider. GPS also serves as an essential component for many other aviation systems, such as the Enhanced Ground Proximity Warning System (EGPWS) that has proven successful in reducing the risk of Controlled Flight into Terrain, a major cause of many aircraft accidents.

2.1.5.2 Benefits The Global Positioning System (GPS)

Continuous, reliable, and accurate positioning information for all phases of flight on a global basis, freely available to all. Safe, flexible, and fuel-efficient routes for airspace service providers and airspace users. Potential decommissioning and reduction of expensive ground based navigation facilities, systems, and services.

Increased safety for surface movement operations made possible by situational awareness.

Reduced aircraft delays due to increased capacity made possible through reduced separation minimums and more efficient air traffic management, particularly during inclement weather. Increased safety-of-life capabilities such as EGPWS [12]

2.1.6 Automatic Dependent Surveillance (ADS)

Automatic Dependent Surveillance(ADS)is a surveillance technique in which air craft provide ,via a data link, data derived from on board navigation and position fixed system include air craft identification ,position and other data .There is two main versions of ADS Automatic Dependent Surveillance-Contract(ADS-C) and Automatic Dependent Surveillance – Broadcast (ADS-B)

2.1.6.1 Automatic Dependent Surveillance-Contract(ADS-C)

ADS-C (Contract) is also known as Automatic Dependent Surveillance – Addressed (ADS-A) or simply Automatic Dependent Surveillance (ADS). With ADS-C the aircraft uses on-board navigation systems to determine its position, velocity, and other data, and reports this information to the responsible air traffic control center. The information that may be sent in ADS-C reports includes

Present position (latitude, longitude, altitude, time), Predicted route in terms of next and (next + 1) waypoints, Velocity (ground or air referenced) and Meteorological data (wind speed, wind direction, and temperature). ADS-C reports are sent by point to point satellite or VHF data links. The data links are typically provided by service providers. Typically, fees are charged for the transmission of each message; as most of these costs are borne by the airlines, there is a reluctance to use ADS-C at higher rates than 10-15 minutes between messages]. Sometimes HF data link is used, but with reduced performance.

In ADS-C the airborne and ground systems negotiate the conditions (the Contract) under which the aircraft submits reports (i.e. periodic reports, event reports, demand reports, and emergency reports). Reports received by the ground system are processed to track the aircraft ATC displays in a similar way to surveillance data obtained from SSRADS-C is typically used in oceanic and remote areas where there is no radar, and hence it is mainly fitted to long range air transport aircraft. The aircraft avionics chooses VHF communication when in coverage of the VHF network to lower costs and improve performance. Satellite data-communications is used another times such as when the aircraft is over the ocean. Typically, messages are transmitted infrequently (~ each 15 minutes). The positional data is accompanied by a “figure of merit” value which indicates the accuracy. It is not an integrity value. The strength of ADS-C is Provides surveillance coverage over very remote regions and oceans except in the polar regions, it Supports a subset of the safety net applications and Minimal maintenance costs. The weakness of it is High costs per report (service provider), Low reporting rates, and Expensive avionics fitment. [6]

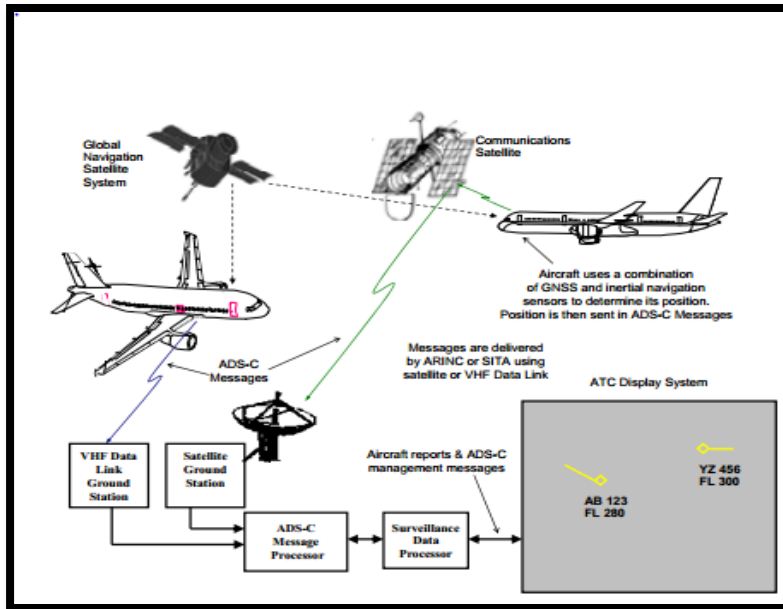


Figure 2.9 Automatic Dependent Surveillance -contract

2.1.6.2 Automatic Dependent Surveillance – Broadcast (ADS-B)

2.1.6.2.1 History and background

The first segment (2006-2010) includes building ground stations in a number of key areas. These areas were chosen as ADS-B test sites due to their high traffic volumes or their proximity to existing ADS-B infrastructure (Kansas, Nebraska, and Louisville).

The second segment (2009-2014) of the US implementation involves completing ground station coverage of the US in existing SSR airspace and ramping up aircraft equipage up to 40%. The expansion is likely to be done by completing infrastructure at airports and airspace within an ARTCC in order to maximize benefits in a region. Segment 2 also includes finalizing the “ADS-B Out” definition.

By Segment 3 (2015-2020) 100% of aircraft are to be equipped with at least “ADS- B Out” with the final definition for “ADS-B In” being created. More applications of ADS-B will be certified.

Finally, in Segment 4 (2020-2025), legacy surveillance equipment, especially SSR, is to be decommissioned. Applications that require full equipage will be fully implemented. [9]

2.1.6.2.2 Next Gen Components

Next Gen is comprised of several components working simultaneously to collect information and disseminate it via a broadcast transmission [Figure 2.10]. Its main source of information is from the constellation of GPS satellites. GPS sensors aboard an aircraft determine its location, which is then joined with information from the aircraft's navigational system or Flight Management System (FMS). The FMS provides information such as the flight plan to ensure the aircraft is following its designated path. [5]

The separation services by ATC requires that the controller and system have knowledge of each aircraft's position, altitude, and direction of flight or intentions. It is critical that each participating ADS-B equipped aircraft's information be received by the ground infrastructure and passed on to ATC in a timely fashion. Traffic statistics from received ADS-B targets can provide some information about the capability of the ground infrastructure to receive information. However, it is not sufficient to verify proper reception by the ground infrastructure of an ADS-B message from an aircraft with equipment operating at required minimum operational specifications. The intended use of ADS-B system data in the provision of aircraft separation services by FAA ATC necessitates flight inspection of the system to ensure that the ADS-B signal-in-space (SIS) is present, useable, and safe with aircraft operating at minimum transmission power. Additionally, flight inspection of the SIS can identify areas in the service volume(s) where: interference sources may exist, there is SIS blockage by terrain and buildings, and obstacles (new or temporary) exist in the intended flight operations area, etc. There is no independent monitoring of the ADS-B SIS (i.e., external sampling of the ADS-B SIS broadcast by the ground facilities) as has been the case in previous navigation and landing systems. Without independent monitoring there is no mechanism to evaluate the broadcast signal-in-space and ensure that the ADS-B ground broadcast requirements are fulfilled. Flight inspection of the ADS-B system can verify requirement compliance [8].

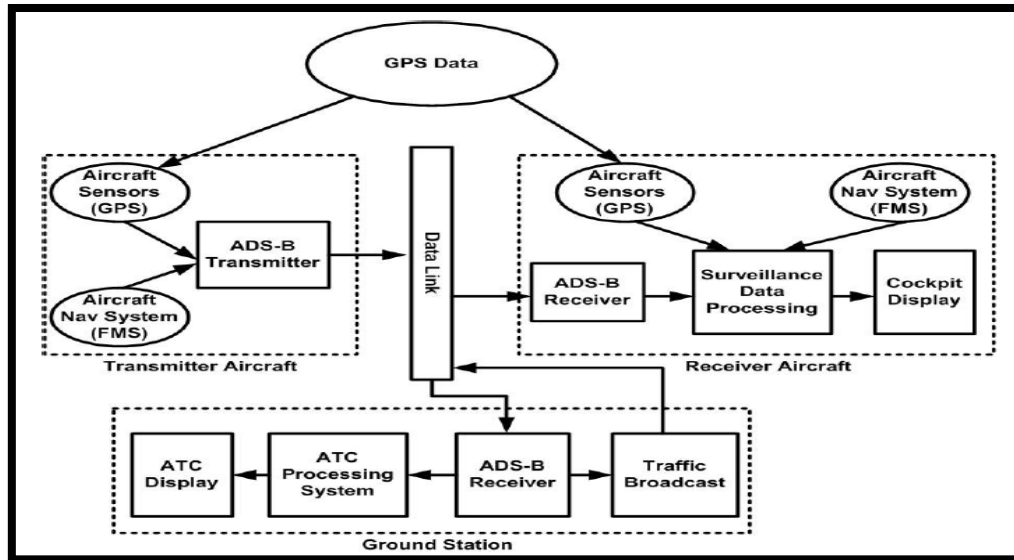


Figure 2.10 Major components of the ADS-B system

Major components of the ADS-B system

Major components of the ADS-B System The FMS and GPS data is fused and broadcasted to other aircraft and ground stations. Both receiving aircraft and ground stations perform similar operations on the message to interpret and display the information on the Control Display Unit (CDU). ADS-B's main purpose is to determine the position of an aircraft and then broadcast that information, along with its altitude, call sign, heading, and aircraft type automatically (i.e., without an SSR interrogation signal) to other aircraft and to air traffic control ground facilities. ADS-B was created with compatibility and ease of transition in mind. It was built using similar aspects of the current aircraft surveillance transmission mode called Mode S or mode select. Mode S operates by interrogating aircraft by a specific aircraft identification number. Only the aircraft possessing the correct identification number will reply to an interrogation with its flight information, eliminating issues with synchronous garbling. Transmission types prior to Mode S include Modes A and C. Mode A provided aircraft identification and Mode C provided altitude. Mode S provides greater capabilities, primarily in the form of aircraft information to include identity, intent, capability and location.

ADS-B is similar to Mode S in that it uses the same transmission frequency of 1090ES MHz's ADS-B uses DF11 and DF17 type of mode S frame to broadcast information such a broadcast is called squitter A squitter message is simply a transmitted message not invoked by any interrogation.

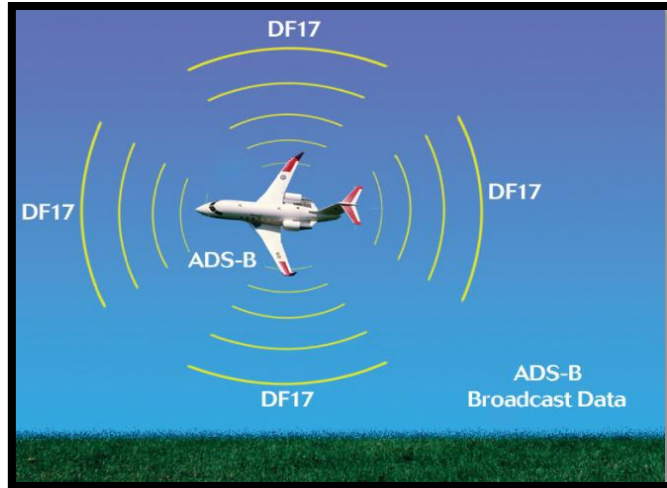


Figure 2.11 Broadcasting of ADS-B Squitter message

DF11 frame is mode S short squitter (56 bits) it do not contain any information other than 24 bit air craft address but the DF17 frame is mode S extended squitter (112 bits) contain an extra 56 bit ADS-B data block as shown below [16].



Figure 2.12 DF11 Frame

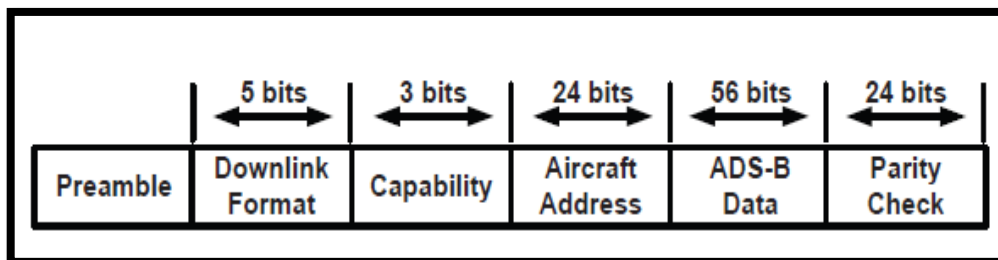


Figure 2.13 DF 12 Frame (ASD-B filed message)

As shown in Figure (2.12), 56 of the 112 bits are for ADS-B specific data to include altitude and airborne position (latitude and longitude). The remaining bits are used for message format, aircraft address, parity check, and finally a few bits for the transponder communication capability Pulse Position Modulation (PPM) is used for encoding and transmitting the Message [5].

There are two types of equipment for ADS-B messages, the first one being *ADS-B In* and the second being *ADS-B Out*

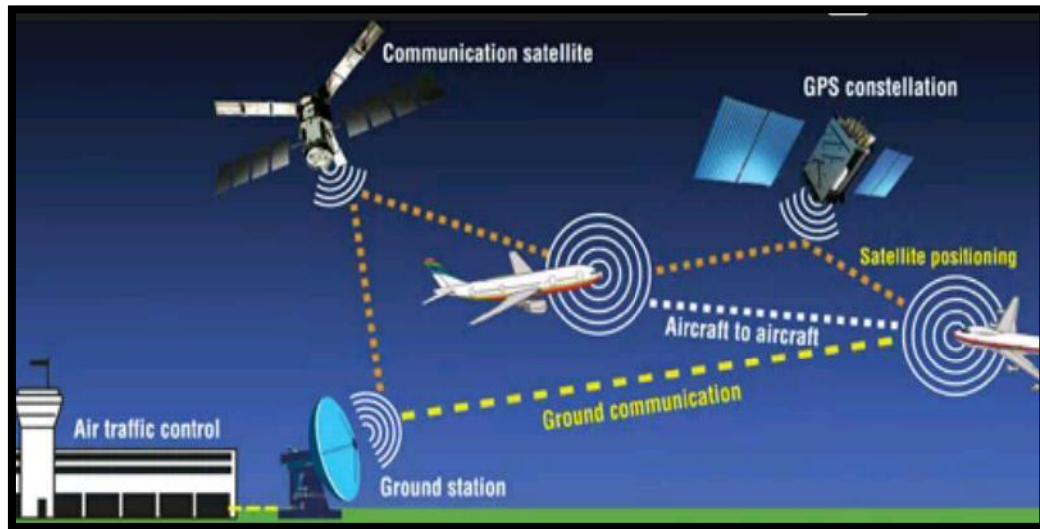


Figure 2.10 ADS-B OUT and IN

Automatic Dependent Surveillance – Broadcast Out: provides a means of automated aircraft parameter transmission between the aircraft and the ATC. It transmits information about identification, altitude, air speed and current position derived through GPS from an equipped aircraft to ground station. ADS-B out work by use ATC transponder to transmit aircraft information to the ground, using mode S 1090MHz extended squatter (random pulses and other non-solicited message) with a refresh rate of 0.5 Sec. The ADS-B signal is broadcast from the aircraft approximately twice per second, and the aircraft is within the coverage radius an ADS-B ground station, data is transmitted to air traffic management.

It can be used in Non-Radar Area , Radar Area and with Air Port Surface:

ADS-B Non-Radar Area[NRA]No need for pin programming to active ADS-B data transmission

ADS-B Radar Area [RAD] Would be the primary mean of surveillance with radar. ADS-B AirPort

Surface [APT] Need tool for surface movement surveillance

ADS-B out have a benefits in air to ground and ground to ground. Air to ground: Enhanced surveillance, Surveillance in non-radar area, Reduced separation standards, ground to ground Improve navigation on surface and enhanced controller awareness of surface traffic

Automatic Dependent Surveillance – Broadcast IN: provides automated aircraft parameter transmission between aircraft themselves, it reception by air craft of Flight information service –

broad cast (FIS-B) and Traffic information service –broad cast (TIS-B) data and other ADS-B data such as direct communication from nearby aircraft [17].

2.1.6.2.3 ADS-B Services

- Traffic information service –broad cast (TIS-B)

TIS-B is an aviation information service that allow pilot to see near real time position and ground track, its present to the pilot combined representation of air craft position derived from GPS satellite and ground based radar data source similar to what an air traffic controller sees on the ground

- Flight information service –broad cast (FIS-B)

FIS-B provide pilot and flight crew of properly equipped with a cockpit display of certain aviation weather and aerodynamic information [1].

2.2 ADS-B Message generation techniques

In The [5] conducted within this thesis explores some of the weaknesses of the system to include the relative ease with which false aircraft targets can be injected The ability to generate, transmit, and insert spoofed ADS-B messages on the display of a commercial ADS-B receiver, identified and exploited a weakness of the ADS-B system. Four demonstrations, conducted within an experimental environment, displayed the potential uses of the system created through this research and its associated impacts. Although it successfully generated ADS-B messages by using a system comprised of GNU Radio, a Universal Software Radio Peripheral (USRP) but it was more expensive complex and the software were not available.

CHAPTER THREE

Proposed ADS-B Message Generation System

3.1 Introduction

The proposed ADS-B generation system consists of:

1. GPS module.
2. Arduino Uno.
3. Liquid crystal display.

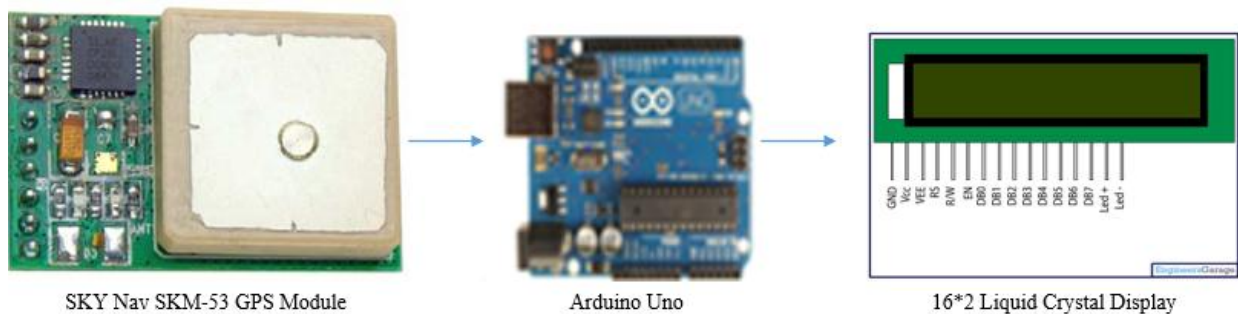


Figure 3.1 ADS-B Message Generation component

In the following, these components were briefly introduced, connected and tested.

3.2 GPS Module (SKM -53)

The Sky Navigation SKM53 Series with embedded GPS antenna enables high performance navigation in the most stringent applications and solid fix even in harsh GPS visibility environments. It is based on the high performance features of the MediaTek 3327 single-chip architecture, Its 165dBm tracking sensitivity extends positioning coverage into place like urban canyons and dense foliage environment where the GPS was not possible before. The 6-pin and USB connector design is the easiest and convenient solution to be embedded in a portable device and receiver like PND, GPS mouse, car holder, personal locator, speed camera detector and vehicle locator. There are many applications for GPS modules such as LBS (Location Based Service), Vehicle navigation system, PND (Portable Navigation Device), GPS mouse and Bluetooth GPS receiver and Timing application.

Table 3.1 GPS Module (SKM -53) PIN Descriptions

Pin No.	Pin name	I/O	Description	Remark
UART Port				
1	5V	P	Module Power Supply	VCC:5V±5%
2	GND	G	Module Power Ground	Reference Ground
3	PPS	O	Time pulse Signal (Default 200ms pulse/sec)	Leave Open in not used
4	RST	I	Module Reset (Active Low Status)	
5	TXD	I	TTL:VOH≥0.75 *VDD VOL≤0.25VDD	Pull up if not used
6	RXD	O	TTL:VIH ≥0.7 *VDD VIL ≤0.3	VDD Leave Open in not used
USB Port				
7	5V	P	USB Power Supply	
8	D-	I/O	Data-	
9	D+	I/O	Data +	
10	GND	G	USB Power Supply	

The GPS Module (SKM -53) has a software protocol called NMEA
National Marine Electronics Association [NMEA]

It's a combined electrical and data specification for communication between marine electronics such as GPS receiver and many other types of instrument

the idea of NEMA is to send a line of data called a sentence that is totally self-contained and independent from other sentences. There are stander sentences for each device category[18] .

The NMEA protocol is an ASCII-based protocol, Records start with a \$ and with carriage return/line feed. GPS specific messages all start with \$ GP xxx where xxx is a three-letter identifier of the message data that follows. NMEA messages have a checksum, which, RMC allows detection of corrupted data transfers show Table 3.2

- SKM53 series supports the following NMEA-0183 messages
GGA, GLL, GSA, GSV VTG, ZDA [19]

Table 3.2 NMEA-0183 Output Messages

NMEA Record	DESCRIPTION
GGA	Global positioning system fixed data
GLL	Geographic position—latitude/longitude
GSA	GNSS DOP and active satellites
GSV	GNSS satellites in view
RMC	Recommended minimum specific GNSS data
VTG	Course over ground and ground speed
ZDA	Time and Date

3.3 Arduino UNO

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically, The ATmega328 has 32 KB of flash memory for storing code.

some pins have specialized functions

1. Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
2. External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attach Interrupt () function for details.
3. PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analog Write () function.

4. SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.

5. LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.[13]

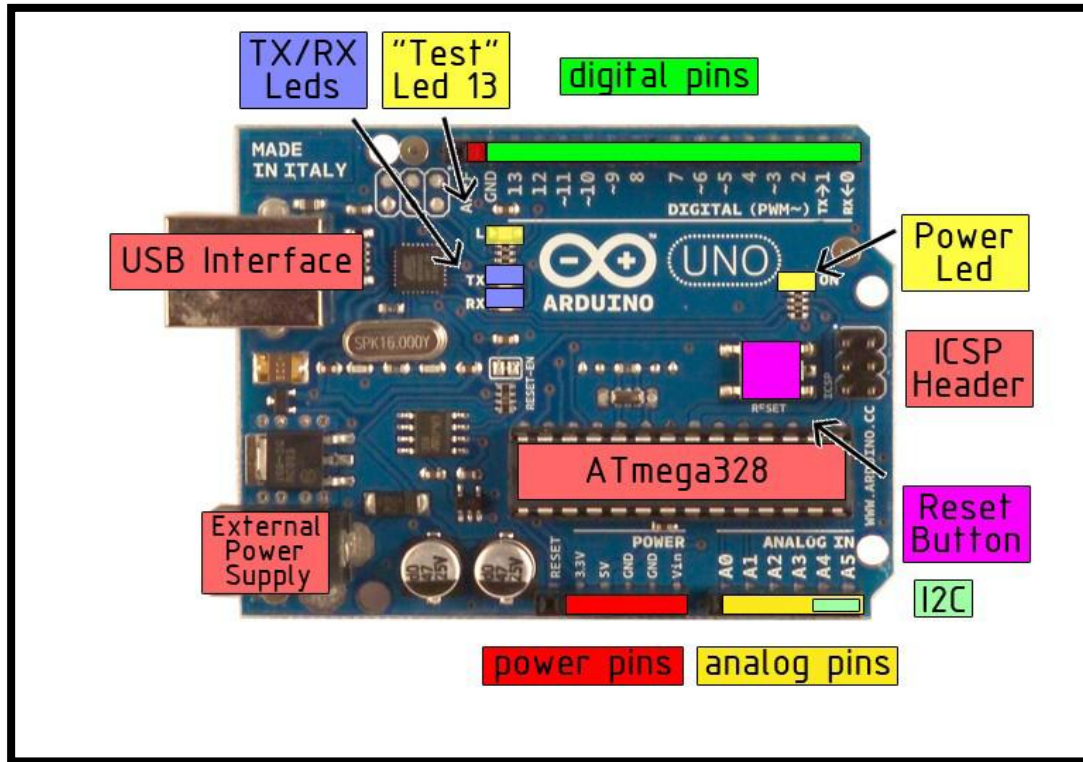


Figure 3.2 Arduino Uno structures

6. The Arduino integrated development environment or Arduino software (IDE) contains a text editor for writing a code, message area, a text console, a tool bar with buttons for common function and a series of menus. It connected to Arduino and Genuine hardware to upload programs and communication with them.

The Arduino support C and C++ Programming languages using special rules of code organization, It consists of two functions that are compiled and linked with a program stub main () in to an executable cyclic executive program

- Setup (): a function that runs once at the start of a program and that can initialize setting
- Loop (): a function called repeatedly until the board [14].

3.4 Liquid Crystal Display 16*2

It's an electronic display module and find a wide range of application, its display is very basic module and is very commonly used in various device and circuits.

This module is preferred over seven segments and other multi segment LEDs, the reason being LCD are economical, easy programmable, have no limitation of displaying special and even custom characters.

It has two register

1. Command register: it stores the command instructions given in LCD.
2. Data register: stores the data to display on the LCD.

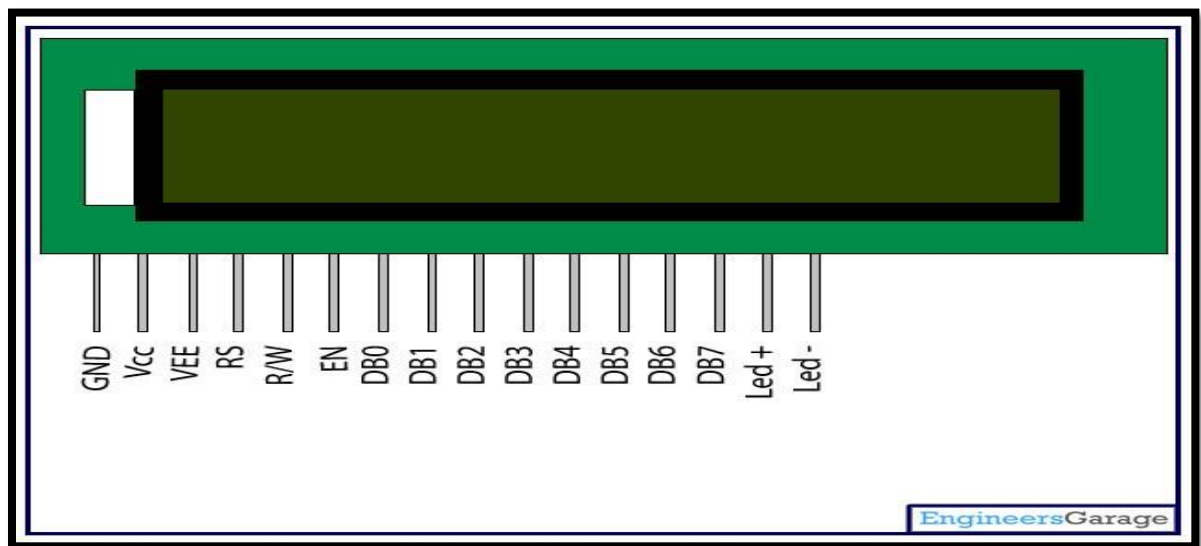


Figure 3.3 Liquid Crystal Display 16*2

3.5 Proteus Design Suite8.0

It's an electronic design automation tool including schematic capture, simulation and PCB layout module.

- Schematic capture: it used for both the simulation of design and as the design phase of a PCB layout project.
- Microcontroller simulation: it works by applying either a hex file or debug file to the microcontroller part on the on the schematic.

3.5.1 Proteus simulation for GPS with Arduino

A GPS library for proteus added as GPS module to proteus simulation, the GPS module hex file which defines how the GPS module should work and also contains dummy NMEA strings, also the Arduino Uno library had been added to proteus library. After extracting all files and libraries to their location open up proteus and add Arduino Uno and GPS module to the work space and connect them as follow: TXD pin of GPS module goes to PIN0 (RXD pin of Arduino) since the GPS module will be transmitting data to Arduino. TXD pin of Arduino is connected to the Virtual terminal so that we can display and view GPS data in the virtual terminal.

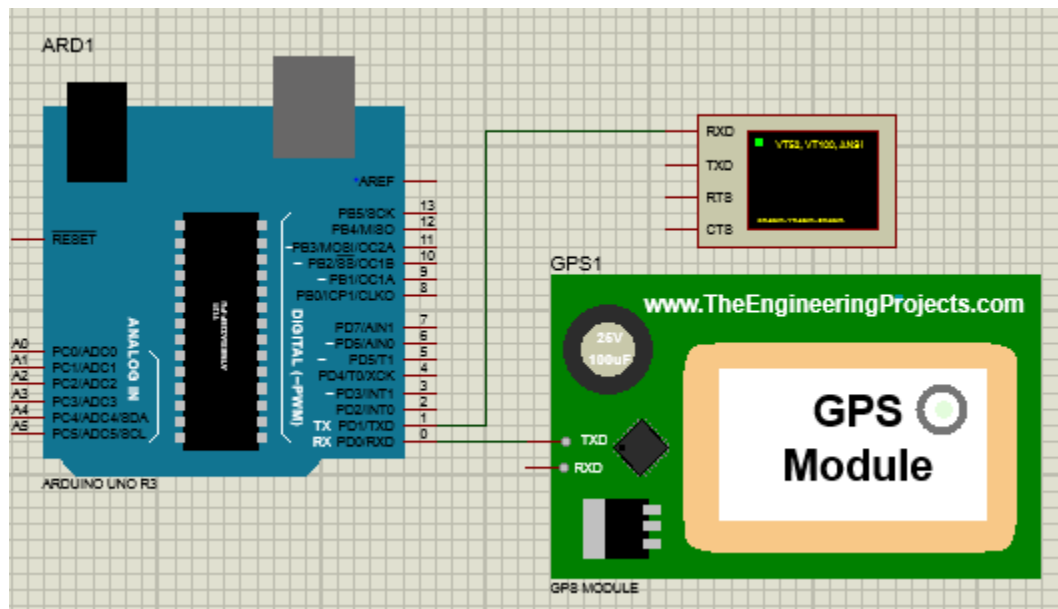


Figure 3.4 Proteus simulation for GPS with Arduino

3.5.2 Proteus simulation for LCD with Arduino

- PIN1 or VSS to ground
- PIN2 (VDD or VCC) to +5 V power
- PIN 3 (VEE) to the end of 10 K to +5 v and ground
- PIN4 or RS (register selection) to PIN 12 of ARDUINO UNO
- PIN5 (R/W) read and write modes to ground
- PIN6 (E) to PIN 11 of Arduino UNO this pin is meant for enabling the LCD module
- PIN 11(D4) to PIN 5 of Arduino UNO

- PIN 12(D5) to PIN 4 of Arduino UNO
- PIN13 (D6) to PIN 3 of Arduino UNO
- PIN14 (D7) to PIN 2 of Arduino UNO
- PIN15 (LDE+) Anode of the back light LED andPIN16 (LDE -) Cathode of the back light LED. [15]

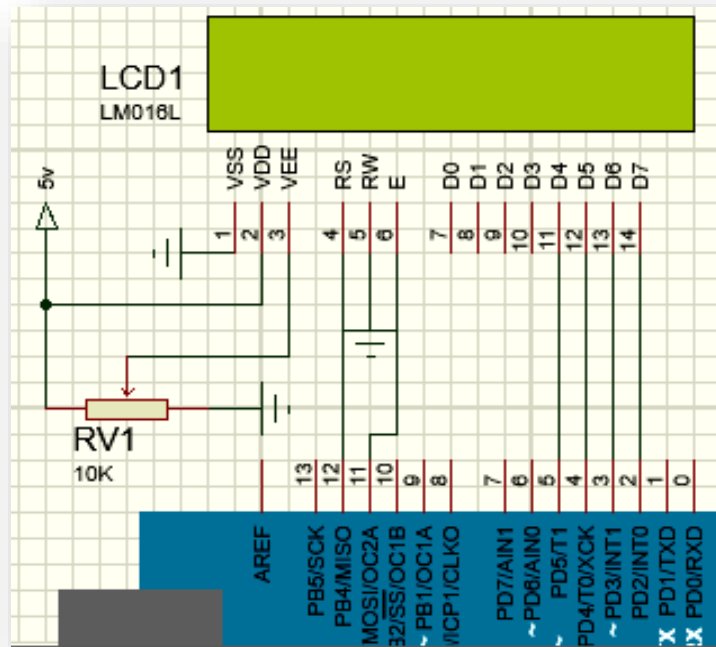


Figure 3.5 Proteus simulation for LCD with Arduino

3.5.3 Proteus simulation for LCD with Arduino and GPS

This simulation is similar to the simulation for LCD with Arduino but here PIN 0 of Arduino UNO is connected to the GPS TX.

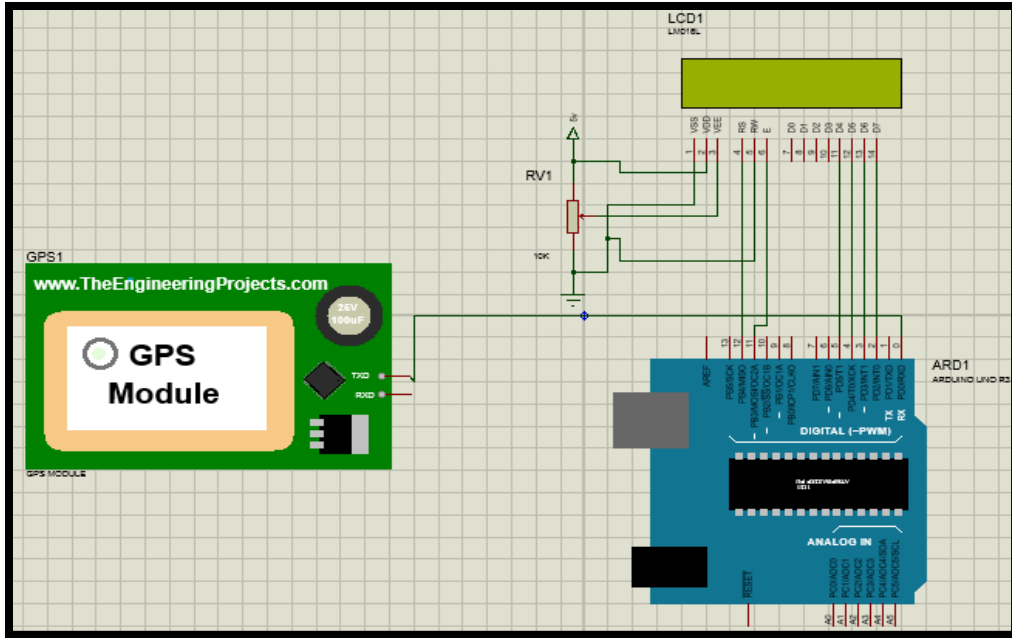


Figure 3.6 Proteus hardware simulation for LCD with Arduino and GPS

CHAPTER FOUR

ADS-B Message Generator

4.1 Introduction

this section, contain the necessary calculations to generate the ADS-B message which is containing the encoded aircraft ID, altitude, positional data, and CRC bits are discussed. The associated Arduino UNO programme that accepts data input from the GPS module and generates a properly encoded ADS-B message are presented completely at the appendix (B). Encoding an ADS-B message requires four inputs: aircraft ID, altitude, latitude and longitude. The aircraft ID will be entered as hexadecimal characters (0-9 & A-F). The altitude will be an integer divisible by 25, as all altitude will be encoded in 25ft increments for greater precision (compared to 100ft increments). Finally, the latitude and longitude inputs represented by degrees, minutes, and seconds (DDMMSS) will be entered in decimal format. The DDMMSS information is converted to decimal degrees for latitude and longitude using the following formula:

$$Decimal Degree = degrees + \frac{mintues}{60} + \frac{secondes}{3600} \dots\dots\dots 4.1$$

After converting latitude and longitude into decimal numbers, it must be determined if they will be negative or positive. Using Figure 4.1 locations north of the equator will yield positive latitudes while locations south of the equator will be negative. Finally, locations east of the prime meridian will result in negative longitudes and vice versa for locations west of the prime meridian.

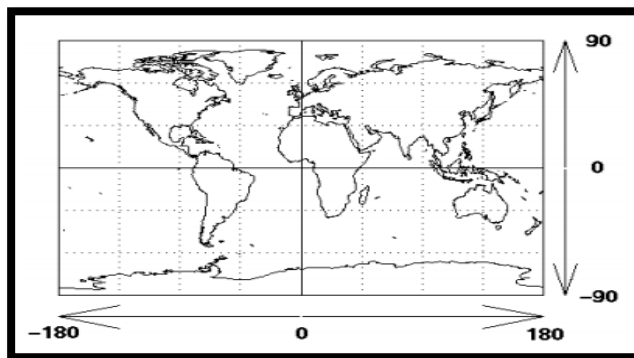


Figure 4.1 Latitude and Longitude Measurements

4.1.1 Down-Link Format (DF) Type

To construct the message as shown in Figure m, the first step is to append the down-link format (DF) format type. This value, 0x8D or 10001101, is added to the message to the Arduino UNO programme. The first five bits (10001) equates to 17 decimal or the desired DF format type. The last three bits are a capability field referring to the specific data being transmitted.

4.1.2 Aircraft Identification Number (ID) Generation

The next data to be entered is the aircraft ID. This is represented by six hex characters or three bytes of data. Since this information is entered as hexadecimal characters it does not need to be manipulated. It can be appended directed to the DF type and stored within the message array.in the figure (4.2) the procedure that used to write the Arduino Uno code to generate aircraft ID

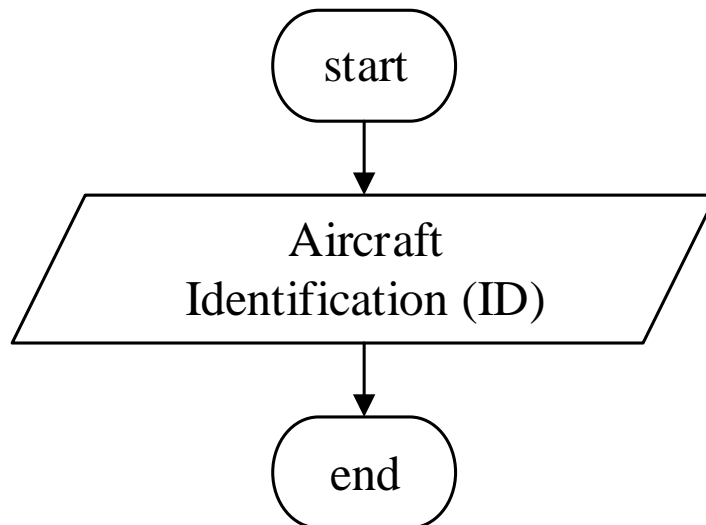
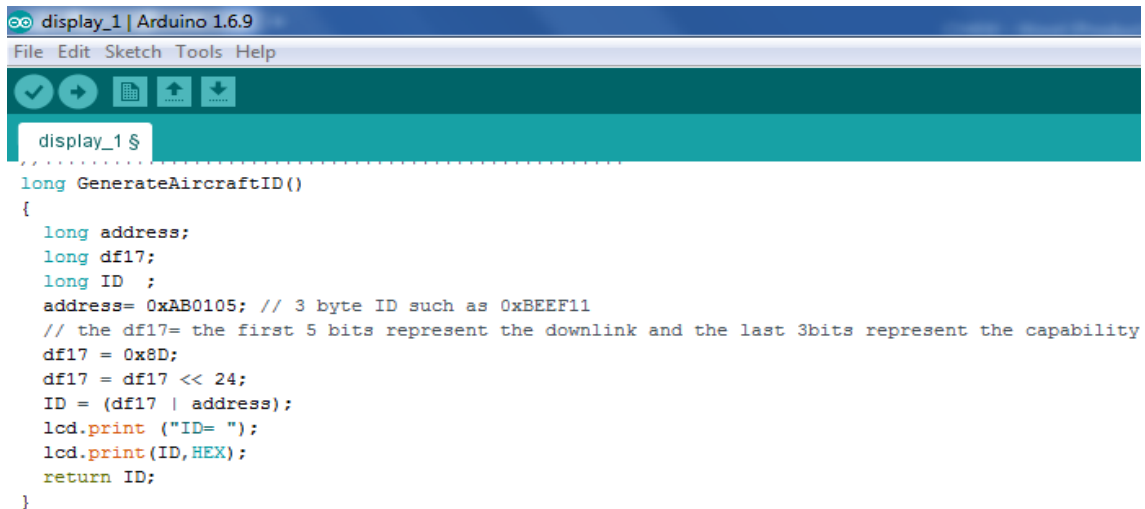


Figure 4.2 Aircraft Identification Number (ID) Generation

The image shows a screenshot of the Arduino IDE interface. The title bar reads "display_1 | Arduino 1.6.9". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for check, run, upload, and download. The main text area shows the following C++ code:

```
display_1 $
long GenerateAircraftID()
{
  long address;
  long df17;
  long ID ;
  address= 0xAB0105; // 3 byte ID such as 0xBEEF11
  // the df17= the first 5 bits represent the downlink and the last 3bits represent the capability
  df17 = 0x8D;
  df17 = df17 << 24;
  ID = (df17 | address);
  lcd.print ("ID= ");
  lcd.print (ID,HEX);
  return ID;
}
```

Figure 4.3 Arduino code for Aircraft Identification Number (ID) Generation

4.1.3 Altitude Encoding

Following the aircraft ID generation is the altitude calculations. This data is entered as a decimal number. To encode it requires subtracting 1,000 and then dividing the number by 25. room must be made for the Q bit. Which is the bit number eight of the 12 bits, counting from left to right, is removed. This bit determines whether the altitude is being reported in 25 foot increments (set to '1') or 100 foot increments (set to '0') . After determining the increment value, the first seven bits are shifted to the right, essentially eliminating the Q bit. This will leave a binary number that can now be converted to decimal. The values in the last four bits are held in memory. Following that, the first eight bits are shifted to the left by one. An OR is then performed to concatenate the shifted bits and the last four bits. To complete the encoding, the Q bit set to 1 (25 ft. increments), is added in the eighth bit position. Similar to the aircraft ID, hexadecimal 0x58 is then appended to the front of the altitude to denote a type code of 11 (01011000), which equates to an airborne position report. The three zeros in the remaining in this field provide surveillance status within the first two bits and the antennas used in the last bit. These three bits are irrelevant to this research and will be set to "000" for all tests. The value for altitude is then stored within the message array. In the figure (4.4) the procedure that used to write the Arduino Uno code to generate the altitude

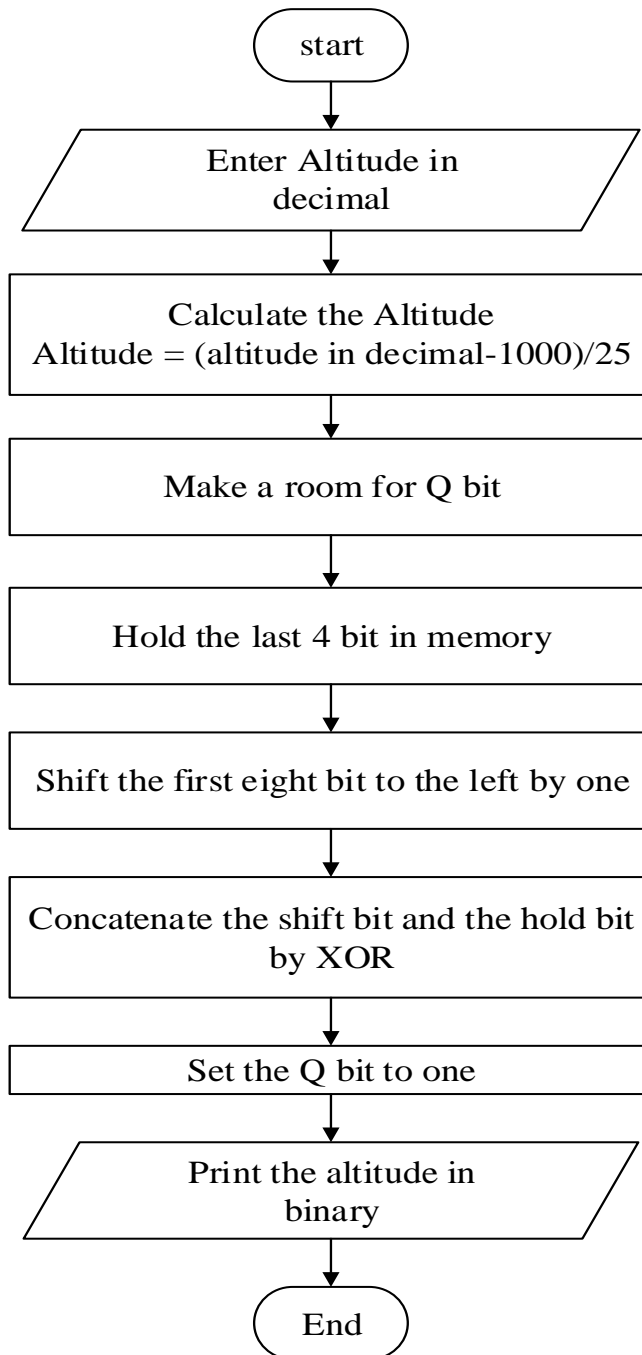


Figure 4.4 Altitude Encoding


```

long CalculateAltitude(int altitude)
{
  //long int altitude;
  int hold;
  long altitudeairborne;
  //altitude =25000;
  altitude = altitude - 1000;
  altitude= altitude/25; // for realaircraft
  //altitude = (altitude + 1)/25;// for simulation purposes
  //Serial.print(altitude,BIN);
  hold = (altitude & 0x000F);// save the last 4 bits in memory
  //lcd.print(hold,BIN);
  //Serial.print("\t");
  altitude=altitude>>4; //shift to the right to remove the last four bits
  altitude<<1; // shift to the left for the Q bit
  altitude = (altitude | 0x0001); // set the Q bit by 1 for 25 ft
  altitude = altitude <<4; // shift to the left 4 bits for the hold bits
  altitude = (altitude | hold); //concatenate to the entire message altitude = (a
  altitudeairborne = altitude;
  lcd.print ("alt= ");
  lcd.print (altitude,BIN);
  lcd.setCursor(0, 0);
  altitudeairborne = (0x00058000| altitudeairborne); //Takes on the TC (0x58) fiel
  //lcd.print(altitudeairborne);
  //Serial.print("\t");
}

```

Figure 4.5 Arduino code for Generate the Altitude value

4.1.4 Latitude and Longitude Encoding

The next two data inputs are the latitude and longitude. These require many more calculations but follow similar procedures as the decoding section. The first step is to determine $Dlat_i$, the latitude zone size [Equation 4.2]. These numbers will remain constant throughout the life of the program. NZ is the number of zones per quadrant which will be 15 at all times. The variable refers to the format type (1 odd or 0 even) Formula

$$Dlat_i = \left(\frac{360}{4*NZ-i} \right) \dots\dots\dots 4.2$$

After determining the $Dlat$ values, the next calculation is the YZ value or Y coordinate within the Zone. This is calculated using the $Dlat$ value and the latitude value input by the user [Equation 4.3]. The result of this equation will give an integer within the 17-bit limit. $Floor(x)$ is defined as the greatest integer k such that $k \leq x$. For example, $floor(5.6)$ will be equal to 5, while $the\ floor(-5.6)$ equals -6. This integer is then assigned to a variable for later use.

$$YZ_I = floor \left(2^{17} * \left(\frac{modulus(lat,Dlat_i)}{Dlat_i} \right) + 0.5 \right) \dots\dots\dots 4.3$$

Realized latitude is then calculated. This value will be approximately the same value as that entered by the user. The latitude, *YZ* and *Dlat* values are all used for this calculation as shown in [Equation 4.4]. The realized latitude or *Rlat* value, calculated during this step is then used to determine the specific longitude zone or *NL*. *NL* is set via a look up table as described earlier in Appendix A.

$$Rlat_i = Dlat_i * \left(\left(\frac{YZ}{2^{17}} \right) + floor \left(\frac{latitude}{Dlat_i} \right) \right) \dots\dots\dots 4.4.$$

in the figure (4.6) the procedure that used to write the Arduino Uno code to encode the latitude

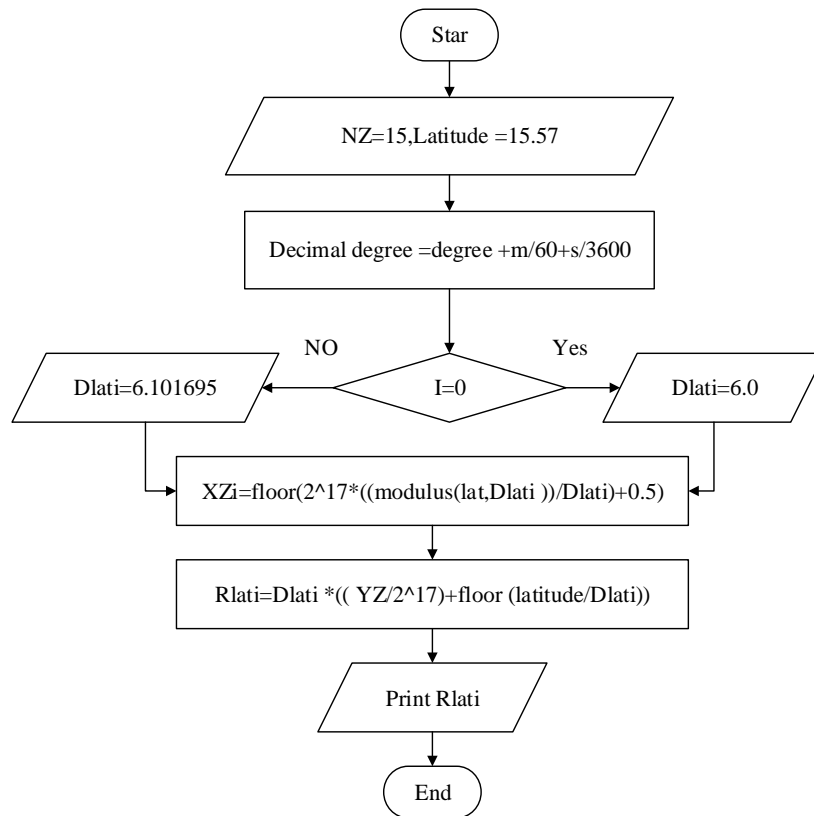


Figure 4.6 Latitude Encoding

```

display_1 | Arduino 1.6.9
File Edit Sketch Tools Help
display_1 $
{
//Calculate YZ which will be what is put into our message
long YZ1;
double modHold;
modHold = Modulus(lat, Dlat1);
modHold = (modHold/Dlat1);
modHold = modHold * pow(2,17);
modHold = modHold + 0.5;
YZ1 = floor(modHold);
lcd.print ("YZ1= ");
lcd.print (YZ1 );
}

```

Figure 4.7 Arduino code for Generate the Latitude value (Odd message)

```

Display_2 | Arduino 1.6.9
File Edit Sketch Tools Help
Display_2 $
long CalculateLatBitsEven(double lat, double longitude)
{
long YZ0;
double modHold;
// Lat= 15.5646;
//double lat;
modHold = Modulus(lat, Dlat0);
modHold = (modHold/Dlat0);
modHold = modHold * pow(2,17);
modHold = modHold + .5;
YZ0 = floor(modHold);
lcd .print ("YZ0=");
lcd.print (YZ0);
}

```

Figure 4.8 Arduino code for Generate the Latitude value (even message)

Using the NL value returned by the lookup table [Appendix A], the longitude zone size ($Dlon$) can then be determined with the simple equation [Equation 4.5]

$$Dlon_i = \begin{cases} \frac{360}{NL(Rlat_i)-i} & \text{when } NL(Rlat_i) - i > 0 \\ 360 & \text{when } NL(Rlat_i) - i = 0 \end{cases} \dots\dots\dots 4.5$$

Finally, the encoded longitude value is calculated. This is known as the XZ value or X coordinate within the Zone:

$$XZ_i = \text{floor} \left(2^{17} * \left(\left(\frac{\text{modulus}(\text{longitude}, D_{lon})}{D_{lon}} \right) + 0.5 \right) \right) \dots\dots\dots 4.6$$

in the figure (4.9) the procedure that used to write the Arduino Uno code to encode the longitude

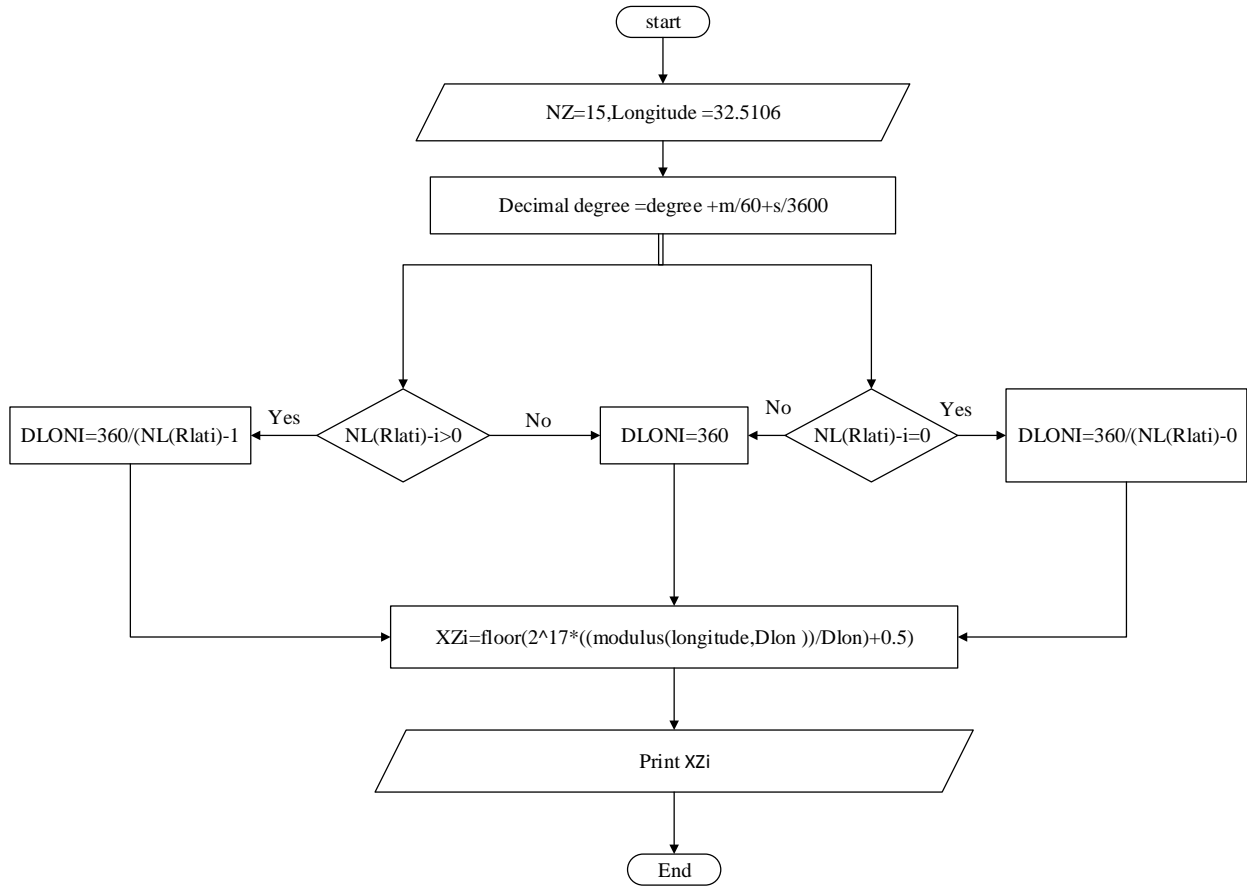


Figure 4.9 longitude encoding

```

display_1 | Arduino 1.6.9
File Edit Sketch Tools Help
display_1 $
else Dlon1 = 360;
//-----
//Calculate XZ the decimal representation of our longitude
long XZ1;
modHold = 0;
modHold = Modulus(longitude, Dlon1);
modHold = (modHold/Dlon1);
modHold = modHold + .5;
modHold = modHold * pow(2,17);
XZ1 = floor(modHold);
lcd.print ("XZ1 =");
|lcd.print (XZ1);

```

Figure 4.10 Arduino code for Generate the Longitude value (Odd message)

```

Display_2 | Arduino 1.6.9
File Edit Sketch Tools Help
Display_2 $
//-----
//Calculate XZ the decimal representation of our longitude
long XZ0;
modHold = 0;
modHold = Modulus(longitude, Dlon0);
modHold = (modHold/Dlon0);
modHold = modHold + .5;
modHold = modHold * pow(2,17);
XZ0 = floor(modHold);
lcd.print ("XZ0=");
|lcd.print (XZ0);

```

Figure 4.11 Arduino code for Generate the Longitude value (even message)

After both numbers have been generated (YZ and XZ), they are concatenated onto each other. It should be noted that these numbers are not added together. The YZ value is shifted to the left 17 positions and then OR'd with the XZ value. This 34-bit value is now ready for transmission as an even message.

4.1.5 Cyclic Redundancy Check (CRC) Generation

The final step of encoding is to apply the CRC polynomial to compute the parity bits. The code for generating CRC bits has some preliminary bit manipulation before beginning the XORing. The first seven lines of code within the CalculateCRC112BitsOdd function ensure 4 bytes of data are in each of the variables to be XOR'd. Once this takes place, the individual portions of the message begin a loop that iterates until all 88 bits have gone through. The result of XORing the 88 bits with polynomial will result in a remainder of 24 bits.

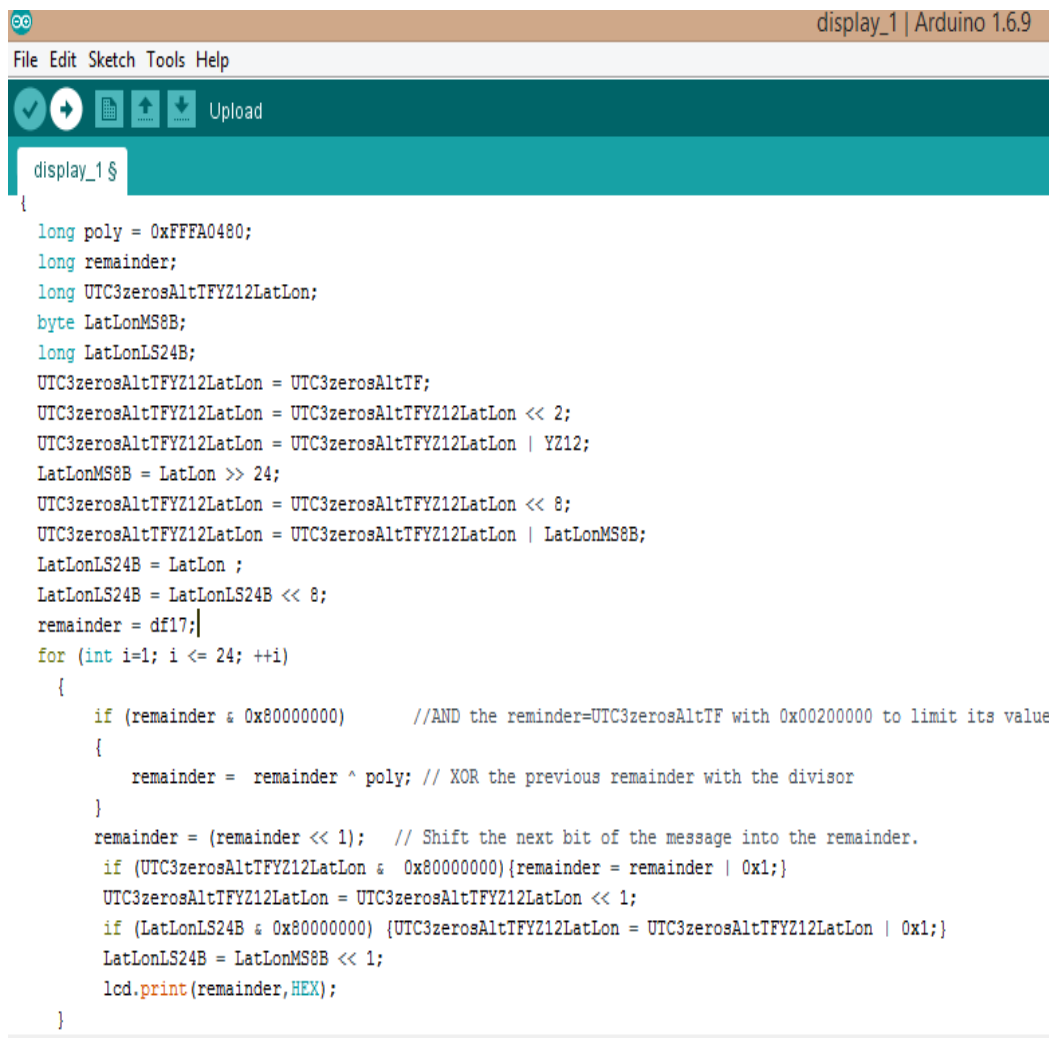
The image shows a screenshot of the Arduino IDE interface. At the top, the title bar reads "display_1 | Arduino 1.6.9". Below the title bar is a menu bar with "File", "Edit", "Sketch", "Tools", and "Help". A toolbar contains icons for a checkmark, a plus sign, a grid, an upload arrow, and a download arrow, with the word "Upload" to the right. The main editor area shows the code for the CalculateCRC112BitsOdd function. The code starts with a function signature "display_1\$" and a brace opening. It defines several variables: "long poly = 0xFFFFA0480;", "long remainder;", "long UTC3zerosAltTFYZ12LatLon;", "byte LatLonMS8B;", and "long LatLonLS24B;". It then performs several bit manipulation operations: "UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTF;", "UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon << 2;", "UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon | YZ12;", "LatLonMS8B = LatLon >> 24;", "UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon << 8;", "UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon | LatLonMS8B;", "LatLonLS24B = LatLon ;", "LatLonLS24B = LatLonLS24B << 8;". It then initializes "remainder = df17;" and enters a "for (int i=1; i <= 24; ++i)" loop. Inside the loop, there is an "if (remainder & 0x80000000) //AND the remainder=UTC3zerosAltTF with 0x00200000 to limit its value" block. Inside this if block, "remainder = remainder ^ poly; // XOR the previous remainder with the divisor" is executed. After the if block, "remainder = (remainder << 1); // Shift the next bit of the message into the remainder." is executed. Then, "if (UTC3zerosAltTFYZ12LatLon & 0x80000000){remainder = remainder | 0x1;}" is executed, followed by "UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon << 1;". Then, "if (LatLonLS24B & 0x80000000) {UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon | 0x1;}" is executed, followed by "LatLonLS24B = LatLonMS8B << 1;". Finally, "lcd.print(remainder,HEX);" is executed. The function ends with a closing brace "}".

Figure 4.12 Arduino code for CRC

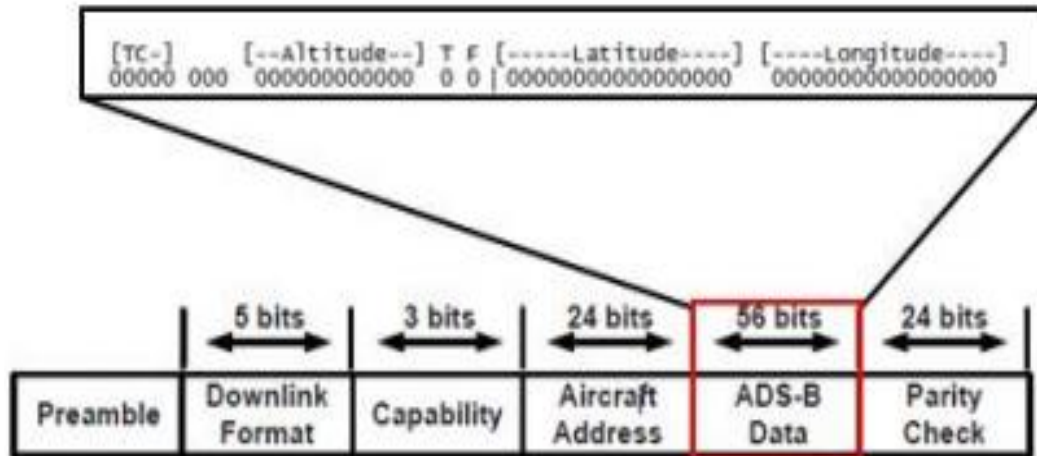


Figure 4.13 ADS-B message format. Modified from

4.2 Results

4.2.1 GPS output from proteus simulation

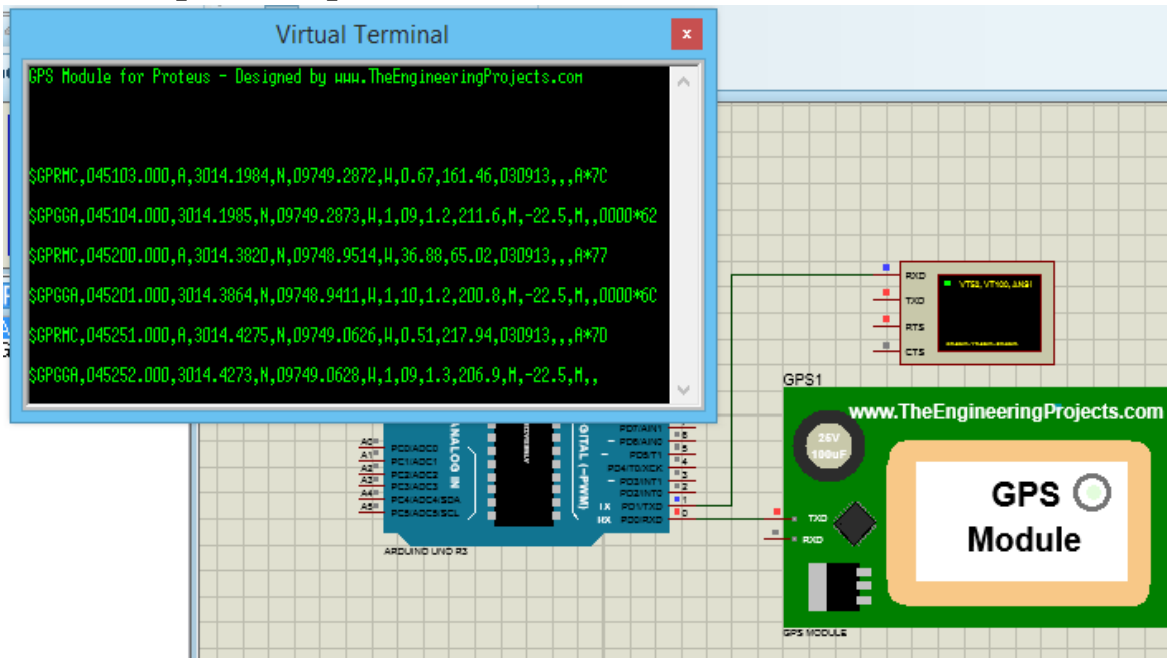


Figure 4.14 GPS output from proteus simulation

- GPS output from hardware circuit

\$GPGSA,A,3,09,17,30,07,28,19,,,,,,,,,1.95,1.70,0.96*0F

\$GPGSV,3,1,09,28,56,011,34,17,47,134,39,05,40,262,,30,32,038,40*7F

\$GPGSV,3,2,09,19,30,172,22,41,30,102,38,09,19,131,35,07,16,064,40*72

\$GPRMC,100046.000,A,1533.8819,N,03232.3675,E,0.00,226.32,190416,,,A*6A

The GPS output decoding to get the position

This result had been decoding to find (altitude, longitude, latitude and UTC time) as shown below

Table 4.1 NMEA-0183 Output Messages Decoding

	UTC Time	Position	Course	Speed, kn/kph	Altitude	HDOP, VDOP, PDOP	Satellites	CRC OK?
GSV						--	9	CRC OK
GSV						--	9	CRC OK
GSV						--	9	CRC OK
RMC	2016-04-19T10:00:46Z	15°33'52.9 1°N, 32°32'22.0 5"E	226.32°			--		CRC OK
GGA	2016-05-02T10:00:47Z	15°33'52.9 1°N, 32°32'22.0 5"E			190.6	1.7 --	6	CRC OK

4.2.2 The Aircraft Identification Encoding

It is entered as hexadecimal characters it does not need to be manipulated so the ID is entering as 0XAB0105 also by add 0x8D as downlink format 17 (the first 5 bits represent the downlink and the last 3bits represent the capability) so the result is expose to be 0x8DAB0105.

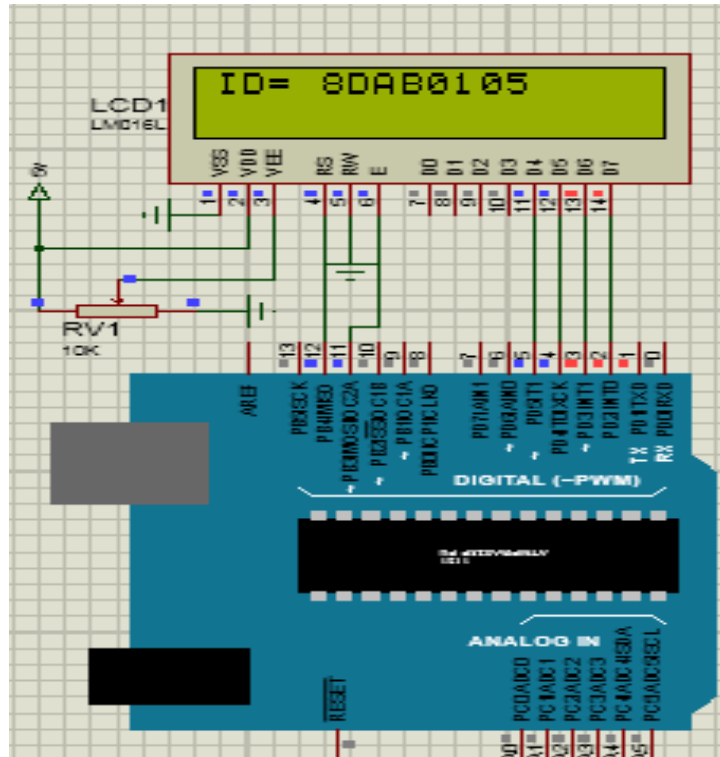


Figure 4.15 The Aircraft Identification Encoding

The result for the aircraft identification from the simulation is typical to result the for the real aircraft identification which is entered first

4.2.3 The Altitude encoding

The altitude enters as 25,000 subtracting 1,000 and then by 25 the room must be made for the Q bit then the first seven bits are shifted to the right also the last four bits are held in memory finally the first eight bits are shifted to the left by one. An OR is then performed to concatenate, so by applying the above steps to the altitude (25,000) the result is= 1936 which is represented by 011110010000 in binary.

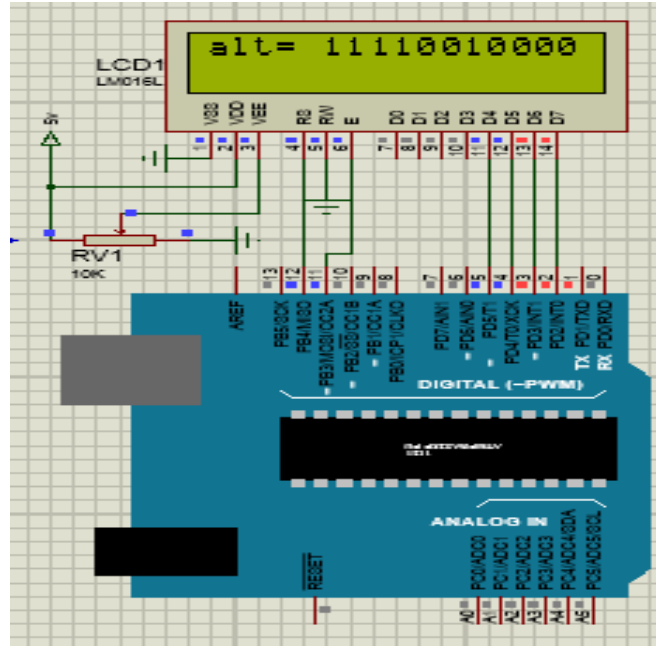


Figure 4.16 The altitude for odd and even message

The result from the simulation is the same as real result

4.2.4 The latitude encoding

4.2.4.1 The latitude for odd message

The latitude can be encoding by calculate YZ_1 so by applying the equation(4.3) as shown below

$$YZ_1 = \text{floor} \left(2^{17} * \left(\frac{\text{modulus}(15.5646, 6.101694915254237288135593220339)}{6.101694915254237288135593220339} \right) + 0.5 \right)$$

The result is $YZ_1 = 72203$

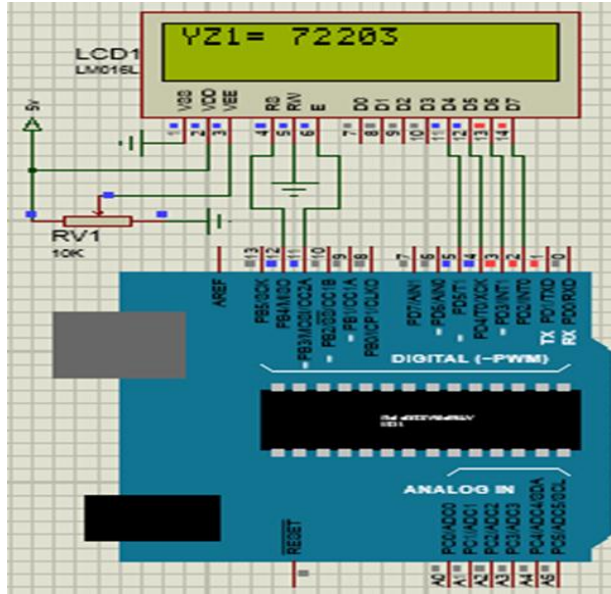


Figure 4.17 The latitude for Odd message

The result for YZ_1 from the simulation is typical to the result of YZ_1 from real calculation

4.2.4.2 The latitude for even message

The latitude can be encoding by calculate YZ_0 so by applying the equation(4.3) as shown below

$$YZ_0 = \text{floor} \left(2^{17} * \left(\frac{\text{modulus}(15.5646, 6.0)}{6.0} \right) + 0.5 \right)$$

The result is $YZ_0 = 77870$

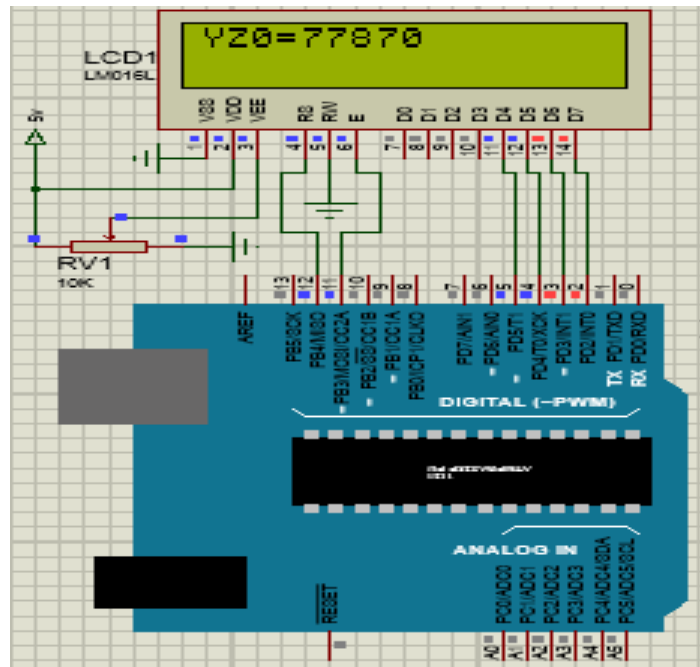


Figure 4.18 The Rlat for odd messege

The result for YZ_0 from the simulation is same to the result of YZ_0 from real calculation

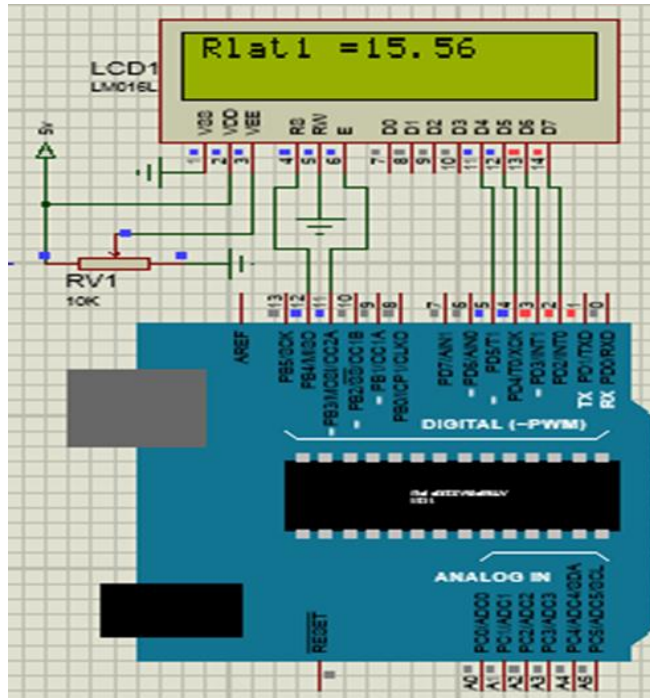


Figure 4. 19 The Rlat for odd messege

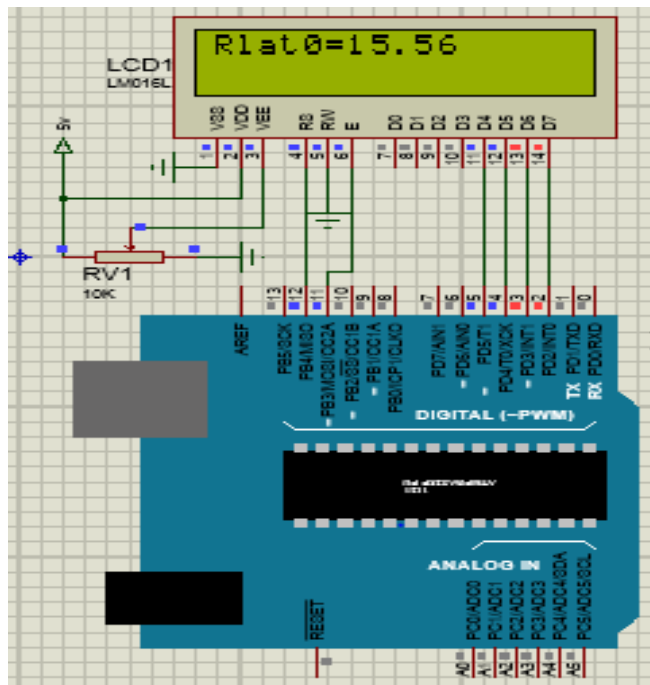


Figure 4.20 The Rlat for even messege

4.2.5 The longitude encoding

4.2.5.1 The longitude for odd message

To encode the longitude for our message firstly the longitude zone size ($Dlon$) must be calculate used the equation (4.5) as shown below

$$Dlon_1 = \left\{ \frac{360}{57(15.56) - 1} \right.$$

The result of is $Dlon_1 = 0.39256$ by take the value of $Dlon_1$ and apply it to equation(4.6) to find XZ_1 to encode the longitude as shown below

$$XZ_1 = floor \left(2^{17} * \left(\left(\frac{modulus(32.5394, 0.39256)}{0.39256} \right) + 0.5 \right) \right)$$

The result of $XZ_1 = 73621$.

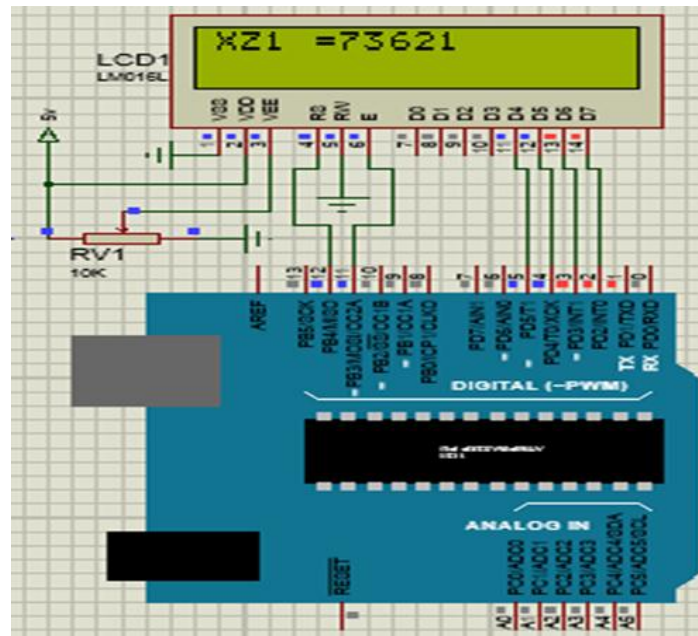


Figure 4.21 The longitude for odd message

4.2.5.2 The longitude for even message

To encode the longitude for our message firstly the longitude zone size ($Dlon$) must be calculate used the equation (4.5) as shown below

$$Dlon_0 = \left\{ \frac{360}{57(15.56) - 0} \right.$$

The result of is $Dlon_0 = 0.39213$ by take the value of $Dlon_0$ and apply it to equation(4.6) to find XZ_0 to encode the longitude as shown below

$$XZ_0 = \text{floor} \left(2^{17} * \left(\left(\frac{\text{modulus}(32.5394, 0.39213)}{0.39213} \right) + 0.5 \right) \right)$$

The result of $XZ_0 = 85468$

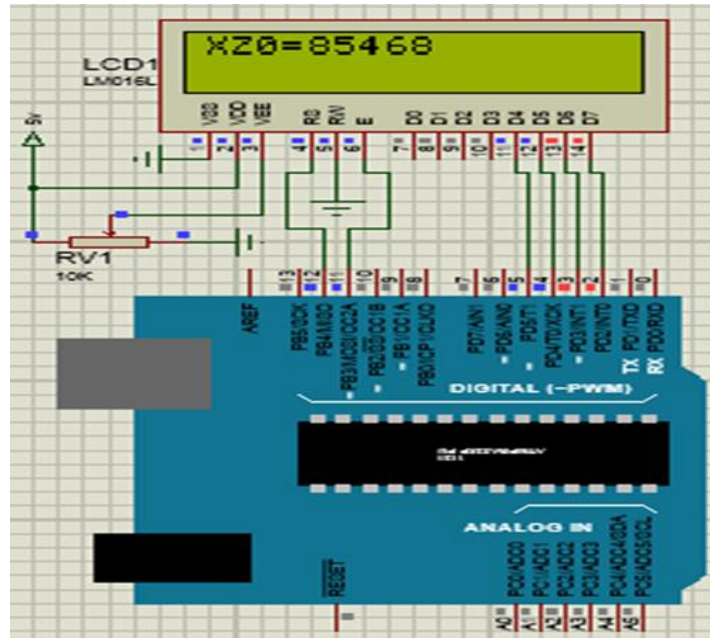


Figure 4.22 The longitude for even message

The result for xz_0 from the simulation is same to the result of xz_0 from real calculation

4.2.6 Cyclic Redundancy Check for odd message

By manually divided the complete message to the remainder (24 bit) the result message was (8000E4)

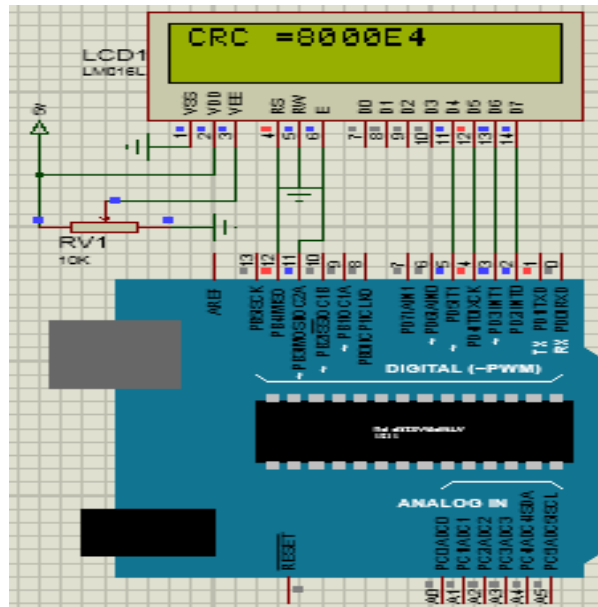


Figure 4.23 Cyclic Redundancy Check for odd message

By manually divided the complete message to the remainder (24 bit) the result for even message was (772D50)

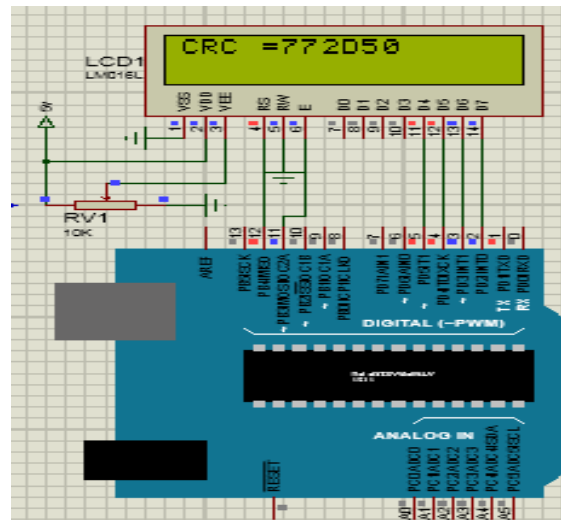


Figure 4.24 Cyclic Redundancy Check for even message

4.2.7 The complete Message

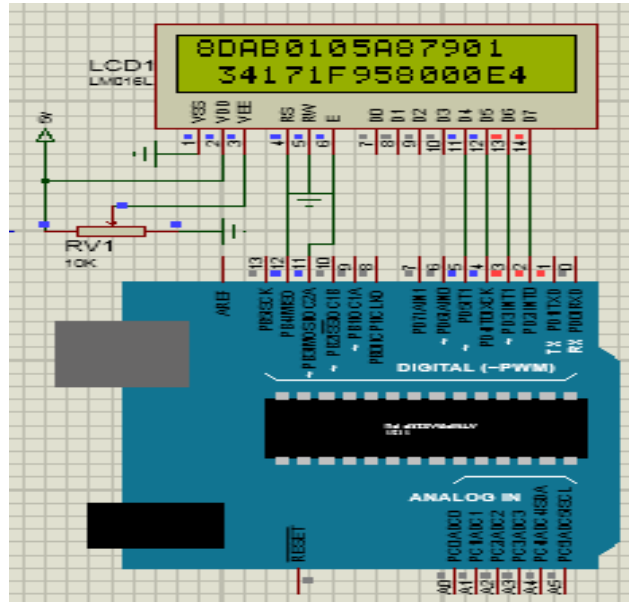


Figure 4.25 The complete odd message

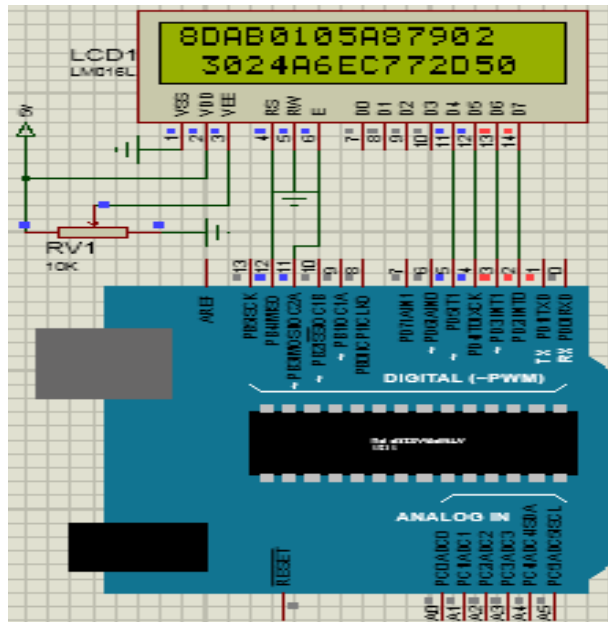


Figure 4.26 The complete even message

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

This chapter contains the conclusion, and recommendations and the future work.

5.2 Conclusion

ADS-B was developing to increase the safety and the efficiency of the navigation system. But the method was used to generate ADS-B message were expensive and not available. The purpose of this project was to generate ADS-B message, using relatively inexpensive equipment.

The system produced through this project was comprised of the following components Sky Nav SKM-53 to extract the position from the GPS, Arduino Uno hardware and software to make interface between the SKM 53 and the computer to receive the position

development of an Arduino Uno code to generate the properly ADS-B messages, the message was generating and the aims of the project is done

5.3 Recommendations

Although the system was developed by using inexpensive equipment (i.e. Arduino Uno) was had the ability to generate ADS-B messages, but it had a small register size (32bits) which is not enough to generate the ADS- B messages in binary So, a different Arduino with register size more than 32 bit such as Arduino MEGA 1280 can be utilized.

5.4 Future work

The future work of this project is to complete ADS-B transmitter and combing it with ADS-B receiver.

Reference

- 1-poster _ESAVS2013_JZ_.PDF
- 2- https://en.m.wikipedia.org/wiki/automatic-dependent-surveillance-_broadcast.
- 3-http://en.m.wikipedia.org/wiki/Air_traffic_control
- 4- http://www.aopa.org/Advocacy/Air_Traffic_Services
- 5 -DOMENIC MAGAZU III, MS CAPTAIN, USAF MARCH 2012 -EXPLOITING THE AUTOMATIC DEPENDENT SURVEILLANCEBROADCAST SYSTEM VIA FALSE TARGET INJECTION THESIS DOMENIC MAGAZU III, CAPTAIN, USAF AFIT/GCO/ENG/12-07
- 6-1.0 – September 2007 Guidance Material on Comparison of Surveillance Technologies (GMST) Edition
- 7- Air Surveillance for smart landing facilities in the small aircraft transportation system.pdf
- 8- Ph.D. Avionic Engineering Center School of Electrical Engineering and Computer Science Ohio University Athens, Ohio 45701-2979REVIEW OF THE AUTOMATIC DEPENDENT SURVEILLANCE-Broadcast (ADS-B) SYSTEM AND DEVELOPMENT OF ADS-B FLIGHT INSPECTION REQUIREMENTS, METHODOLOGIES, AND PROCEDURES by Robert J. Thomas, M.S.E.E. Michael F. DiBenedetto,
- 9- Edward A. Lester and R. John Hansman Report No. ICAT-2007-2 August 2007. Benefits and Incentives for ADS-B Equipage in the National Airspace System
- 10-<http://www.navcen.uscg.gov/?pageName=gpsmain>
- 11-http://www.faa.gov/about/office_org/headquarters_offices/ks
- 12-<http://www.gps.gov/applications/aviation>
- 13- D-c- Arduino uno.pdf digital.csic.es
- 14-<https://en.m.wikipedia.org/wiki/Arduino>
- 15-http://circuitdigest.com/microcontroller_project/Arduino_LCD_inter_facing_tutorial.
- 16-ECE499-SP2.ngDabin .pdf
- 17-Final Rule for ADS-B Out.pdf
- 18- <https://en.m.wikipedia.org/wiki/nmea-0183>
- 19- SKM53-DS-030609.pdf

Appendix A(number of longitude zones (NL))

Condition	Transition Latitude		Number of Longitude Zones, NL	
	Degrees (decimal)	32-bit AWB (hexadecimal)		
If lat <	10.47047130	07 72 17 54	Then NL(lat) =	59
Else if lat <	14.82817437	0A 8B 63 03	Then NL(lat) =	58
Else if lat <	18.18626357	0C EE B5 50	Then NL(lat) =	57
Else if lat <	21.02939493	0E F4 48 D6	Then NL(lat) =	56
Else if lat <	23.54504487	10 BE 3E 9F	Then NL(lat) =	55
Else if lat <	25.82924707	12 5E 12 29	Then NL(lat) =	54
Else if lat <	27.93898710	13 DE 23 2C	Then NL(lat) =	53
Else if lat <	29.91135686	15 45 32 43	Then NL(lat) =	52
Else if lat <	31.77209708	16 97 EF 0B	Then NL(lat) =	51
Else if lat <	33.53993436	17 D9 C2 3B	Then NL(lat) =	50
Else if lat <	35.22899598	19 0D 3E 35	Then NL(lat) =	49
Else if lat <	36.85025108	1A 34 62 2C	Then NL(lat) =	48
Else if lat <	38.41241892	1B 50 C4 78	Then NL(lat) =	47
Else if lat <	39.92256684	1C 63 AE 77	Then NL(lat) =	46
Else if lat <	41.38651832	1D 6E 2F 8C	Then NL(lat) =	45
Else if lat <	42.80914012	1E 71 2A 88	Then NL(lat) =	44
Else if lat <	44.19454951	1F 6D 5F 49	Then NL(lat) =	43
Else if lat <	45.54626723	20 63 71 E6	Then NL(lat) =	42
Else if lat <	46.86733252	21 53 F0 01	Then NL(lat) =	41
Else if lat <	48.16039128	22 3F 54 E9	Then NL(lat) =	40
Else if lat <	49.42776439	23 26 0C C7	Then NL(lat) =	39
Else if lat <	50.67150166	24 08 77 22	Then NL(lat) =	38
Else if lat <	51.89342469	24 E6 E8 E0	Then NL(lat) =	37
Else if lat <	53.09516153	25 C1 AD DF	Then NL(lat) =	36
Else if lat <	54.27817472	26 99 0A 48	Then NL(lat) =	35
Else if lat <	55.44378444	27 6D 3B A2	Then NL(lat) =	34
Else if lat <	56.59318756	28 3E 79 B3	Then NL(lat) =	33
Else if lat <	57.72747354	29 0C F7 42	Then NL(lat) =	31
Else if lat <	58.84763776	29 D8 E2 B2	Then NL(lat) =	30
Else if lat <	59.95459277	2A A2 66 89	Then NL(lat) =	30
Else if lat <	61.04917774	2B 69 A9 E5	Then NL(lat) =	29

Appendix B(Arduino Uno Code To Generate ADS-B message)

```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  Serial.begin (9600);

  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  // Print a message to the LCD.
}

#define pi3.14159265;
long Dlat0 = 6;          //even messages
double Dlat1 = 6.101694915254237288135593220339; //odd messages
long val;
long modval;
long m;
long RE;
long alt;
void loop() {
  // Turn off the display:
  lcd.noDisplay();
  delay(500);
  // Turn on the display:
  lcd.display();
  delay(500);
  // put your main code here, to run repeatedly:
  alt=CalculateAltitude(25000);
  RE=CalculateCRC112BitsOdd(0x8DAB0105, 0x2A1E40, 0x1, 0x68171F95);
```

```
CalculateLatBitsOdd(15.5646,32.5394);
```

```
L=Modulus (15.5646,Dlat0);
```

```
xeven=CalculateLatBitsEven(15.5646,32.5394);
```

```
m= message(0x8DAB0105, 0x2A1E40, 0x1, 0x34171F95,0x8000E4);
```

```
}
```

```
//+++++
```

```
+
```

```
double Modulus(double val, double modval)
```

```
{
```

```
long result;
```

```
double modulusresult ;
```

```
if (val < 0) //Checking for negative angles
```

```
{
```

```
val = val + 360;
```

```
result = floor(val / modval);
```

```
modulusresult = val - (result * modval);
```

```
return modulusresult ;
```

```
}
```

```
else
```

```
{
```

```
result = floor(val / modval);
```

```
modulusresult = val - (result * modval);
```

```
return modulusresult ;
```

```
}
```

```
}
```

```
//=+++++
```

```
long CalculateAltitude(int altitude)
```

```
{
```

```

int hold;

long altitudeairborne;

altitude =25000;

altitude = altitude - 1000;

altitude= altitude/25; // for realaircraft

hold = (altitude & 0x000F);// save the last 4 bits in memory

altitude=altitude>>4; //shift to the right to remove the last four bits

altitude=altitude<<1; // shift to the left for the Q bit

altitude = (altitude | 0x0001); // set the Q bit by 1 for 25 ft

altitude = altitude <<4; // shift to the left 4 bits for the hold bits

altitude = (altitude | hold); //concatenate to the entire message altitude = (altitude | 0x010);

altitudeairborne = altitude;

lcd.print ("alt= ");

lcd.print (altitude,BIN);

lcd.setCursor(0, 0);

altitudeairborne = (0x00058000| altitudeairborne); //Takes on the TC (0x58) field

lcd.print(altitudeairborne);

return altitude;

}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

long GenerateAircraftID()

{

long address;

long df17;

address= 0xAB0105; // 3 byte ID such as 0xBEEF11

// the df17= the first 5 bits represent the downlink and the last 3bits represent the capability

df17 = 0x8D;

df17 = df17 << 24;

```

```

df17 = (df17 | address);
lcd.print(df17,HEX);
return df17;
}

//+++++
void CalculateLatBitsOdd(double lat, double longitude)
{
//Calculate YZ which will be what is put into our message
long YZ1;
double modHold;
modHold = Modulus(lat, Dlat1);
modHold = (modHold/Dlat1);
modHold = modHold * pow(2,17);
modHold = modHold + 0.5;
YZ1 = floor(modHold);
lcd.print ("YZ1= ");
lcd.print (YZ1 );
lcd.setCursor(0, 0);
//-----
//Calculate Rlatiude for airborne
double Rlat1;
double floorHoldtemp;
long floorHold;
Rlat1 = YZ1 / pow(2,17);
floorHoldtemp = (lat/Dlat1);
floorHold = floor(floorHoldtemp);
Rlat1 = Rlat1 + floorHold;

```

```

Rlat1 = Rlat1 * Dlat1;
int NILat;
NILat = GenerateNlatValue(Rlat1);
//-----
//Calculate DLongitud
double Dlon1;
if ((NILat - 1) > 0) { Dlon1 = (360/double((NILat-1))); }
else Dlon1 = 360;
//-----
//Calculate XZ the decimal representation of our longitude
long XZ1;
modHold = 0;
modHold = Modulus(longitude, Dlon1);
modHold = (modHold/Dlon1);
modHold = modHold + .5;
modHold = modHold * pow(2,17);
XZ1 = floor(modHold);
lcd .print ("XZ1 =");
lcd.print (XZ1);
lcd.setCursor(0, 0);
//-----
//Ensure this fits into our 17 bit space
long LatLon;
YZ1 = Modulus(YZ1,pow(2,17));
XZ1 = Modulus(XZ1,pow(2,17));
int YZ12;
if ( YZ1>=0 & YZ1<pow(2,15)) {YZ12=0;}
else if( YZ1 >= pow(2,15) & YZ1 < pow(2,16)) {YZ12=1;}

```



```

else if( YZ1 >= pow(2,16) & YZ1<(pow(2,16)+pow(2,15))) {YZ12=2;}
else if (YZ1< pow(2,17) & YZ1>= (pow(2,16)+pow(2,15))) {YZ12=3;}
else {YZ12=0;}
YZ1 = YZ1 << 17;
LatLon = (YZ1 | XZ1);
if (LatLon < pow(2,31) & LatLon >= pow(2,30))    {YZ12= YZ12 << 1;}
else if (LatLon < pow(2,30)& LatLon >= pow(2,29)) {YZ12= YZ12 << 2;}
else if (LatLon < pow(2,29)& LatLon >= pow(2,28)) {YZ12= YZ12 << 3;}
else if (LatLon < pow(2,28)& LatLon >= pow(2,27)) {YZ12= YZ12 << 4;}
else{YZ12= YZ12 << 5;}
LatLon = (LatLon | 17179869184);
//return (YZ1MSBs,LatLon);
lcd.print('YZ12=');
lcd.print("LatLon,BIN=");
lcd.print(YZ12,BIN);
lcd.print(LatLon,BIN);
lcd.setCursor(0, 0);
}
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
int GenerateNlatValue(double latitude)
{
int NL;
if (latitude >= 87.00000000)           {NL=1;}
else if (latitude <87.00000000 && latitude >= 86.53536998) {NL=2;}
else if (latitude <86.53536998 && latitude >= 85.75541621) {NL=3;}
else if (latitude <85.75541621 && latitude >= 84.89166191) {NL=4;}
//.....
else if (latitude <18.18626357 && latitude >= 14.82817437) {NL= 57;}

```

```

else if (latitude <14.82817437 && latitude >= 10.47047130) {NL= 58;}
else {NL= 59;}
return NL;
}
//+++++
++
long UTC3zerosAltTFdata(int UTC, int altitude)
{
    long UTC3zerosAltTF;           // this variable contains the UTC, 3zeros, altitude and
    TF with total 22 bits

    UTC3zerosAltTF = UTC;           // put the UTC value at UTC3zerosAltTF

    UTC3zerosAltTF = UTC3zerosAltTF << 3;    //shift left dor the 3 zeros locations

    UTC3zerosAltTF = UTC3zerosAltTF & 0xfffff8 ;    // AND the UTC3zerosAltTF with
    0xfffff8 to attache the 3 zeros

    UTC3zerosAltTF = UTC3zerosAltTF << 12;    //shifleft UTC3zerosAltTF by 12 bits for
    altitude location=12 bits

    altitude = altitude & 0xfff;           //aquize only the 12 bits of the altitude

    UTC3zerosAltTF = UTC3zerosAltTF | altitude;    // locate the altitude with the
    UTC3zerosAltTF

    UTC3zerosAltTF = UTC3zerosAltTF << 2 ;    // shift left the UTC3zerosAltTF to attache
    the TF=00

    UTC3zerosAltTF = UTC3zerosAltTF & 0xfffffc;    // AND the UTC3zerosAltTF with
    0x0xfffffc to attache the 2 zeros

    return UTC3zerosAltTF;
}
//+++++

long CalculateCRC112BitsOdd(long df17, long UTC3zerosAltTF, int YZ12, long LatLon)
{
    long poly = 0xFFFA0480;
    long remainder;

```

```

long UTC3zerosAltTFYZ12LatLon;
byte LatLonMS8B;
long LatLonLS24B;
UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTF;
UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon << 2;
UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon | YZ12;
LatLonMS8B = LatLon >> 24;
UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon << 8;
UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon | LatLonMS8B;
lcd.print(UTC3zerosAltTFYZ12LatLon,HEX);
LatLonLS24B = LatLon ;
LatLonLS24B = LatLonLS24B << 8;
remainder = df17;
for (int i=1; i <= 24; ++i)
{
//If the uppermost bit is a 1
    if (remainder & 0x80000000) //AND the remainder=UTC3zerosAltTF with 0x00200000
to limit its value for the UTC3zerosAltTF only =22 bits
    {
remainder = remainder ^ poly; // XOR the previous remainder with the divisor
    }
remainder = (remainder << 1); // Shift the next bit of the message into the remainder.
if (UTC3zerosAltTFYZ12LatLon & 0x80000000){remainder = remainder | 0x1;}
UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon << 1;
if (LatLonLS24B & 0x80000000) {UTC3zerosAltTFYZ12LatLon =
UTC3zerosAltTFYZ12LatLon | 0x1;}
LatLonLS24B = LatLonMS8B << 1;
lcd.print(remainder,HEX);
}

```

```

// Return only the relevant bits of the remainder as CRC = 24 bits
return remainder;
//return (remainder >> 24);
}
long message (long df17, long UTC3zerosAltTF, int YZ12, long LatLon, long CRC)
{
    long message ; // this variable contains the ADS-B message
    long UTC3zerosAltTFYZ12LatLon;
    long message;
    long latlon;
    message = df17;
    lcd.print( message,HEX);
    message=UTC3zerosAltTF;
    message=message| YZ12;
    UTC3zerosAltTFYZ12LatLon = message;
    UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTF;
    UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon << 2;
    UTC3zerosAltTFYZ12LatLon = UTC3zerosAltTFYZ12LatLon | YZ12;
    lcd.print( UTC3zerosAltTFYZ12LatLon,HEX);
    lcd.setCursor(1, 1);
    lcd.print( LatLon ,HEX);
    lcd.print("8000E4");
    lcd.setCursor(0, 0);
    return message;
}
//Encoding of Even message
long CalculateLatBitsEven(double lat, double longitude)
{

```

```

long YZ0;
double modHold;
modHold = Modulus(lat, Dlat0);
modHold = (modHold/Dlat0);
modHold = modHold * pow(2,17);
modHold = modHold + .5;
YZ0 = floor(modHold);
//Calculate Rlatiude for airborne
double Rlat0;
double floorHoldtemp;
long floorHold;
Rlat0 = YZ0 / pow(2,17);
floorHoldtemp = (lat/Dlat0);
floorHold = floor(floorHoldtemp);
Rlat0 = Rlat0 + floorHold;
Rlat0 = Rlat0* Dlat0;
lcd.print("Rlat0=");
lcd.print(Rlat0 );
delay (10);
int NILat;
NILat = GenerateNlatValue(Rlat0);
NILat=57;
//-----
//Calculate DLongitud
double Dlon0;
if ((NILat - 0) > 0) { Dlon0 = (360/double((NILat-0))); }
else Dlon0 = 360;
lcd.print(Dlon0);

```

```

//-----
//Calculate XZ the decimal representation of our longitude
long XZ0;
modHold = 0;
modHold = Modulus(longitude, Dlon0);
modHold = (modHold/Dlon0);
modHold = modHold + .5;
modHold = modHold * pow(2,17);
XZ0 = floor(modHold);
lcd.print("XZ0=");
lcd.print(XZ0);
lcd.setCursor(0,0);
//Ensure this fits into our 17 bit space
long LatLon;
YZ0 = Modulus(YZ0,pow(2,17));
XZ0 = Modulus(XZ0,pow(2,17));
int YZ12;
if ( YZ0>=0 & YZ0<pow(2,15))           {YZ12=0;}
else if( YZ0 >= pow(2,15) & YZ0 < pow(2,16))   {YZ12=1;}
else if( YZ0 >= pow(2,16) & YZ0<(pow(2,16)+pow(2,15))) {YZ12=2;}
else if (YZ0< pow(2,17) & YZ0>= (pow(2,16)+pow(2,15))) {YZ12=3;}
else {YZ12=0;}
YZ0 = YZ0 << 17;
LatLon = (YZ0| XZ0);
if (LatLon < pow(2,31) & LatLon >= pow(2,30))    {YZ12= YZ12 << 1;}
else if (LatLon < pow(2,30)& LatLon >= pow(2,29)) {YZ12= YZ12 << 2;}
else if (LatLon < pow(2,29)& LatLon >= pow(2,28)) {YZ12= YZ12 << 3;}
else if (LatLon < pow(2,28)& LatLon >= pow(2,27)) {YZ12= YZ12 << 4;}

```

```
else{YZ12= YZ12 << 5;}  
LatLon = (LatLon | 17179869184);  
//return (YZ1MSBs,LatLon);  
lcd.print(YZ12,BIN);  
lcd.print(LatLon,BIN);  
}
```