Contents

[2. Abstract – 2](#_Toc177675410)

[3. Aircraft Surveillance Technologies – 3](#_Toc177675411)

[Primary Surveillance Radar - 3](#_Toc177675412)

[Secondary Surveillance Radar - 5](#_Toc177675413)

[4. ADS-B 10](#_Toc177675414)

[Message Structure 10](#_Toc177675415)

[Availability and Transmission Rate - 13](#_Toc177675416)

[ADS-B Versions 15](#_Toc177675417)

[5. ADS-B message receiver - 18](#_Toc177675418)

[Receiver Range: 18](#_Toc177675419)

[Antenna Setup 19](#_Toc177675420)

[Receiver Setup 19](#_Toc177675421)

[6. ADS-B Message Types and Creation 24](#_Toc177675422)

[Aircraft identification and Category 24](#_Toc177675423)

[Aircraft Airborne Position 31](#_Toc177675424)

[References 32](#_Toc177675425)

## Abstract

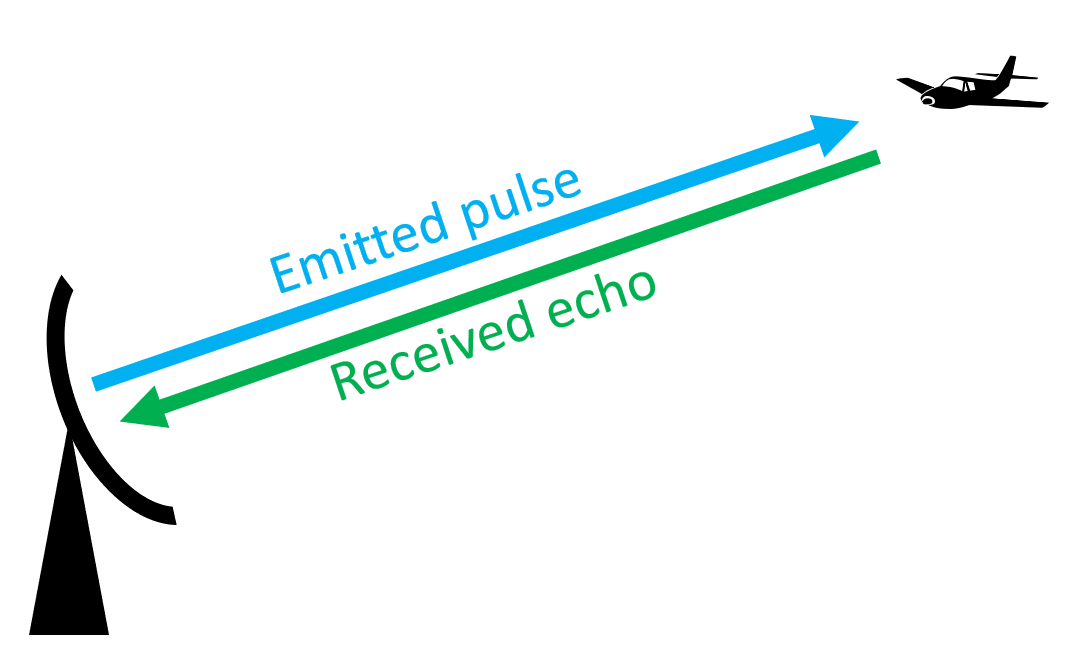
This report presents the development and implementation of a methodology for generating, transmitting and decoding of Automatic Dependent Surveillance–Broadcast (ADS-B) messages, aimed at enhancing the understanding of ADS-B communication protocols and their behavior with the multi-band transceiver currently in development. The report also analyzes security protocols and suggests methods for secure transmission and reception by further encrypting the ADS-B message with 128-bit AES-EBS encryption.

This report presents the development and implementation of a system for generating, transmitting, and decoding Automatic Dependent Surveillance–Broadcast (ADS-B) messages, aimed at enhancing the understanding of ADS-B communication protocols and their security implications. Utilizing MATLAB, several codes were developed to generate ADS-B message signals, including aircraft identification, airborne position, and velocity data. A Software Defined Radio (SDR) was configured to receive and decode these messages, providing practical insights into the communication process. The generated ADS-B messages were successfully transmitted through a transceiver chain and verified for accuracy upon reception. Additionally, a security-focused application was integrated into the system, where the ADS-B messages were encrypted using the AES-EBC algorithm prior to transmission. The encryption and decryption processes were implemented using Java crypto functions within MATLAB, ensuring the confidentiality of the transmitted data. The results demonstrated the effectiveness of the system in both message transmission and security, highlighting the potential for further research and development in secure ADS-B communications. This work contributes to the ongoing efforts to improve aviation safety and data integrity in airborne communication systems.

## Aircraft Surveillance Technologies

### Primary Surveillance Radar

Aircraft Surveillance started with the concept of capturing reflections from a rotating radio transponder with an omni-directional antenna, now known as the Primary Surveillance Radar (PSR). The radar transmits a 1µs pulse for every 1ms and listens to the reflections from the airplanes. Concepts of phase filters and Doppler filters are used to filter out moving targets (aircrafts/drones) and remove static objects (mountains, buildings and other obstacles). The PSR determines the aircraft’s position using 2 key measurements –

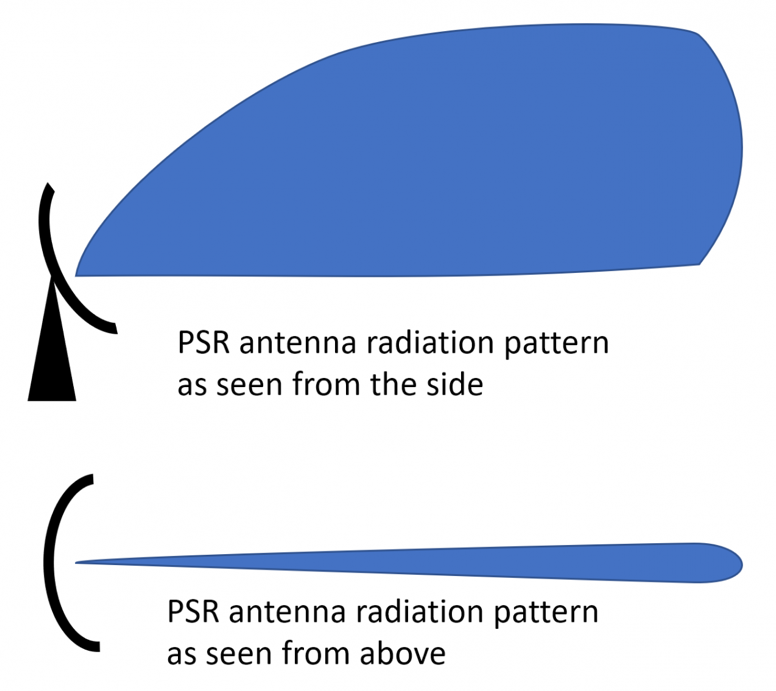
1. Range – Calculated from the time difference between when the pulse was emitted and when the reflection was received.
2. Bearing – Determined by the antenna’s azimuth at the moment the reflection was received.

The PSR outputs the data in a polar coordinate system, providing range and bearing of detected targets relative to the antenna’s position. The range provided is the slant distance from the antenna to the aircraft. This creates a 2D representation of aircraft positions in the PSR’s coverage area.

However, the PSR has several significant disadvantages for aircraft position mapping –

1. Coverage Limitations –

PSR often has inadequate coverage, especially for airspace directly above the antenna due to its radiation pattern. This creates blind spots that require multiple radar installations to fully cover the area.



1. Data Limitations –

It does not provide any information related to the identity of the aircraft. It also fails to capture accurate altitude information due to the nature of its tracking.

1. Detection Issues –

It cannot distinguish between multiple aircraft at the same slant range but different altitudes. It may also detect false targets from ground vehicles, weather, birds, etc. Additionally, performance degrades in the presence of ground clutter and adverse weather conditions.

1. Resource Limitations –

PSRs need optimal sites with unobstructed views and minimal ground clutter. They often require high transmitter power for long range performance and is expensive to install and maintain as compared to other surveillance technologies.

These disadvantages have led to the addition of more technologies on top of the PSR to compensate for the shortcomings. The Secondary Surveillance Radar (SSR), also known as the Air Traffic Control Beacon System (ATCBS), was designed to provide air traffic controllers with more information. The SSR can be installed separately or on top of a primary radar.

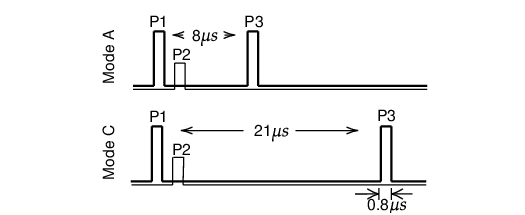
### Secondary Surveillance Radar

#### Mode A/C

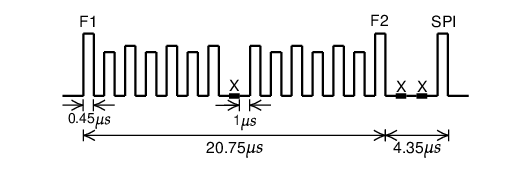
The SSR transmits interrogation messages at 1030MHz, and the aircraft transponder transmits replies at 1090MHz. Initial implementations of SSR worked on Mode A and Mode C (civilian communication protocols) to continuously interrogate the squad code (aircraft ID) and the altitude of the aircraft.

Mode A was designed to provide aircraft identification. An aircraft equipped with a transponder would respond to SSR interrogations by transmitting a 4-digit octal code, which is manually set by the pilot. This code, known as a squawk code, helps air traffic controllers identify the aircraft located by the PSR. Mode A was limited to 4096 unique codes (8484), which posed challenges in areas with high air traffic density.

Mode C added altitude reporting to the capabilities of Mode A. It provided the aircraft's pressure altitude data, allowing air traffic controllers to determine the aircraft's vertical position in addition to its identity. Mode C transponders responded to SSR interrogations by transmitting altitude information, typically with a resolution of 100 feet.

The SSR initiates Mode A and Mode C interrogations with 2 different pulse patterns, which are shown in the figure below –

The pulses are about 0.8μs wide. P1 and P3 are the two main pulses sent by the directional antenna. They are separated by 8μs and 21μs, respectively for Mode A and Mode C. P2 is a pulse emitted through the omnidirectional antenna right after P1. Pulse P2 is introduced for side-lobe suppression. When the power of P2 is higher than P1, the interrogation is likely from the side lobes of the directional antenna and should be ignored by the aircraft. This can happen when the aircraft is close to the radar.

The figure below shows an example of Mode A/C reply –

Each reply consists of two persistent pulses, F1 and F2, separated by 20.3μs. Within this period, either the identity code or the altitude code is encoded using 13 pulses of 0.45μs. The pulses are separated by gaps of 1μs. The pulse at the center serves as a verification pulse and is always absent. The presence or absence of any of the other 12 pulses represents a 1 or 0 bit. When required by air traffic controllers for identification purposes, a special purpose identification (SPI) may follow F2 after two absent pulses.

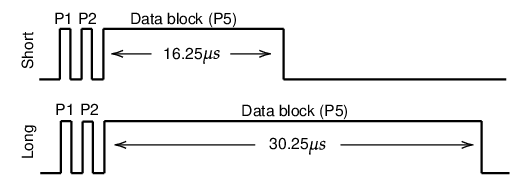
This design of ATCRBS works sufficiently well with low-density air traffic, but it cannot efficiently cope with higher flight densities since all aircraft replies are transmitted on the same frequency. When several aircraft are in the same direction of the radar beam, reply signals can overlap and introduce errors for decoding. This is known as ***synchronous garbling***.

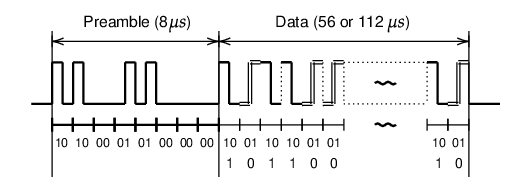
When there are multiple secondary radars in the vicinity, replies originated by other radars may be considered as valid responses of one radar, which in turn, causes errors and confusion. This syndrome is called ***FRUIT*** ***(False Replies Unsynchronized In Time)***.

#### Mode S

Mode S (Mode Select Beacon System) introduces the capability of selective interrogations. This allows the SSR to interrogate different information from different aircraft separately. By using selective interrogation, it largely mitigated the problem of garbling in Mode A/C and thus greatly improved the capacity of the communication channel.

Unlike the limited number (4096) of unique identification codes in Mode A communication, the Mode S transponder is identified by a 24-bit transponder code, which can support up to 16,777,216 (224) unique addresses. In addition to these two major advantages, the Mode S protocol introduced many different types of information that could be interrogated and downlinked.

 The Mode S uplink signal contains parameters that indicate which information is desired by the air traffic controller. There are two types of Mode S interrogations, shown in the figure below. The short interrogation has 56 bits of information contained in the data block, while the long interrogation contains 112 bits of information. The P2 pulse acts as the side-lobe suppression for Mode A/C transponders so that they will ignore the rest of the interrogation pulses. Information in the Mode S interrogation data block uses the Differential Phase-Shift Keying (DPSK) modulation, which is a type of Phase Modulation.

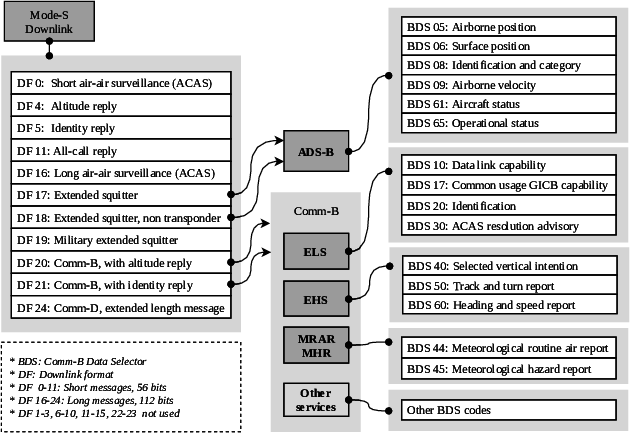
There are two types of Mode S downlink signals, the short reply and the long reply, which correspond to the short and long interrogations from the SSR. For each microsecond, one bit is transmitted. All Mode S replies start with an 8 μs fixed preamble and continue with 56μs or 112μs data block. The structure of the downlink message is shown in the figure below.

The information contained in the data block is modulated using the Pulse Position Modulation (PPM), which is a type of amplitude modulation. In PPM, the 1 bit is represented by a 0.5μs of pulse followed by a 0.5μs flat signal. The 0 bit is reversed compared to the 1 bit, which is represented by a 0.5μs flat signal and followed by a 0.5μs pulse.

The Mode S communication protocol is designed to handle different types of uplink and downlink message formats. The first five bits of the message define the uplink format (UF) or downlink format (DF) number of the message. Based on the UF/DF number, different structures of the data block are presented. The table below shows all available Mode-S formats. Numbers not in this table are currently reserved for future use.

|  |  |  |  |
| --- | --- | --- | --- |
| UF/DF | Bits | Uplink type | Downlink type |
| 0 | 56 | Short air-air surveillance (ACAS) | Short air-air surveillance (ACAS) |
| 4 | 56 | Surveillance, altitude request | Surveillance, altitude reply |
| 5 | 56 | Surveillance, identity request | Surveillance, identity reply |
| 11 | 56 | Mode S All-Call | All-Call reply |
| 16 | 112 | Long air-air surveillance (ACAS) | Long air-air surveillance (ACAS) |
| 17 | 112 | - | Extended squitter |
| 18 | 112 | - | Extended squitter/non transponder |
| 19 | 112 | - | Military extended squitter |
| 20 | 112 | Comm-A, altitude request | Comm-B, altitude reply |
| 21 | 112 | Comm-A, identity request | Comm-B, identity reply |
| 24 | 112 | Comm-C (ELM) | Comm-D (ELM) |

Among all uplink formats, UF 17, 18, and 19 are not used. This is because the corresponding downlink messages (extended squitter messages) are designed to be broadcast automatically without the need for SSR interrogations. One of the most common applications for the extended squitter is the Automatic Dependent Surveillance-Broadcast service, which is commonly known as ADS-B. Figure below shows Mode S services and their relationships with different Mode S downlink formats.



## ADS-B

Automatic Dependent Surveillance-Broadcast (ADS-B) is a surveillance technology designed to allow aircraft to broadcast their flight state periodically without the need for interrogation. The word *automatic* refers to the fact that no inputs from controllers or pilots are required. The word *dependent* indicates this technology depends on information from other onboard systems, such as air data systems and navigation systems.

### Message Structure

A line with a circle and a circle with a circle and a circle with a circle and a circle with a circle and a circle with a circle and a circle with a circle and a circle with

Description automatically generatedAn ADS-B frame is 112 bits long and consists of the following main parts -

|  |  |  |  |
| --- | --- | --- | --- |
| Bit | No. bits | Abbreviation | Information |
| 1–5 | 5 | DF | Downlink Format |
| 6–8 | 3 | CA | Transponder capability |
| 9–32 | 24 | ICAO | ICAO aircraft address |
| 33–88 | 56 | ME | Message, extended squitter |
| (33–37) | (5) | (TC) | (Type code) |
| 89–112 | 24 | PI | Parity/Interrogator ID |

1. **DF (Downlink Format):** The first 5 bits indicate the format of the message. For ADS-B messages, this is set to 10001 (binary), which corresponds to Downlink Format 17.
2. **CA (Capability):** The next 3 bits provide information about the transponder's capabilities, which can vary based on the type of ADS-B message being sent. The capability value can be a decimal value between 0 and 7.

|  |  |
| --- | --- |
| CA | Definition |
| 0 | Level 1 transponder |
| 1–3 | Reserved |
| 4 | Level 2+ transponder, with ability to set CA to 7 when on-ground |
|  |  |
|  |  |
| 5 | Level 2+ transponder, with ability to set CA to 7 when airborne |
| 6 | Level 2+ transponder, with ability to set CA to 7 when either on-ground or airborne |
|  |  |
|  |  |
| 7 | Indicates that the Downlink Request value is 0, or that the Flight Status is 2, 3, 4, or 5, either airborne or on the ground |
|  |  |

A Level 1 transponder is the most basic type of transponder. It primarily provides fundamental identification capabilities. This level typically transmits a unique identification code (squawk code) when interrogated by air traffic control (ATC) but does not provide altitude information or advanced data capabilities. Level 1 transponders may be used in less congested airspace or for operations that do not require detailed surveillance data. A Level 2+ transponder represents a more advanced category with enhanced capabilities beyond basic identification. Level 2+ transponders can provide altitude information and operational status. Level 2+ transponders are required in controlled airspace and are essential for commercial aviation, where precise tracking and altitude reporting are critical for safety.

1. **ICAO Address:** The ICAO address is located from 9 to 32 bits in binary (or 3 to 8 in hexadecimal positions). A unique ICAO address is assigned to each Mode S transponder of an aircraft and serves as the unique identifier for each aircraft, according to ICAO regulations [1]. In principle, this code does not change over the lifetime of the aircraft. However, it is possible to reprogram a transponder so that the messages contain a different address. This has been observed for some military aircraft, as well as some private airplanes opting-in for the FAA Privacy ICAO Address System [2].
2. **ME (Message Elements):** The 56 bits in this section contain various types of data, including position, velocity, and other operational parameters. The specific content depends on the Type-Code defined in this field.

|  |  |
| --- | --- |
| Type Code | Data frame content |
| 1–4 | Aircraft identification |
| 5–8 | Surface position |
| 9–18 | Airborne position (w/Baro Altitude) |
| 19 | Airborne velocities |
| 20–22 | Airborne position (w/GNSS Height) |
| 23–27 | Reserved |
| 28 | Aircraft status |
| 29 | Target state and status information |
| 31 | Aircraft operation status |

1. **PI (Parity/Interrogator ID):** The last 24 bits are used for error checking and known as the CRC (Cyclic Redundancy Check) remainder. The primary purpose of the CRC remainder is to detect errors that may occur during the transmission of ADS-B messages.

### Availability and Transmission Rate

ADS-B messages are available to any receiver equipped with ADS-B IN capabilities, allowing for widespread situational awareness among pilots and air traffic controllers. Different ADS-B messages have different transmission rates. The update frequency also differs depending on whether the aircraft is on-ground or airborne, as well as whether the aircraft is still or moving when on the ground.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Messages | TC | Ground (still) | Ground (moving) | Airborne |
| Aircraft identification | 1–4 | 0.1 Hz | 0.2 Hz | 0.2 Hz |
| Surface position | 5–8 | 0.2 Hz | 2 Hz | - |
| Airborne position | 9–18, 20–22 | - | - | 2 Hz |
| Airborne velocity | 19 | - | - | 2 Hz |
| Aircraft status | 28 | 0.2 Hz (*no TCAS RA and Squawk Code change*) | | |
| 1.25 Hz (*change in TCAS RA or Squawk Code*) | | |
| Target states and status | 29 | - | - | 0.8 Hz |
| Operational status | 31 | 0.2 Hz | 0.4 Hz (no NIC/NAC/SIL change) | |
| 1.25 Hz (change in NIC/NAC/SIL) | |
|  | |

For Operational Status messages (Type Code 31), when there is a change in NIC, NAC, or SIL, the transmission rate **increases to** **1.25 Hz**. This means that if any of these integrity indicators change, an operational status message will be sent more frequently to ensure that air traffic control and other aircraft are aware of the updated reliability of navigation data. If there are no changes in these categories, the transmission rate **remains at 0.4 Hz**.

* **NIC (Navigation Integrity Category)**: NIC indicates the integrity of the navigation data being transmitted. It assesses how accurately the aircraft's position can be trusted based on the navigation system's performance. A higher NIC value means greater confidence in the accuracy of the position information.
* **NAC (Navigation Accuracy Category)**: NAC represents the accuracy of the position information provided by the aircraft's navigation system. It quantifies how close the reported position is to the actual position. This helps air traffic controllers and other aircraft gauge how reliable the positional data is.
* **SIL (Source Integrity Level)**: SIL indicates the reliability of the source providing the position information. It assesses whether the data is trustworthy and meets operational requirements.

For Aircraft Status Messages (Type Code 28), if there is a change in TCAS RA or squawk code (the unique identifier assigned to each aircraft), the transmission rate **increases to 1.25 Hz**. This ensures that any critical changes affecting collision avoidance are communicated promptly. If there are no changes in TCAS RA or squawk code, these messages are sent at a rate of **0.2 Hz**.

* **TCAS RA (Traffic Collision Avoidance System Resolution Advisory)**: TCAS RA refers to alerts generated by a Traffic Collision Avoidance System when an aircraft is at risk of collision with another aircraft. The system advises pilots on necessary maneuvers to avoid potential collisions.

### ADS-B Versions

ADS-B has had 3 versions of implementations throughout its lifespan. This was done to include more information in ADS-B messages. These have been put together from the official ICAO 9871 document [3].

1. **Version 0: DO-260/ED-102**

This was the initial standard for ADS-B, providing foundational guidelines for the technology. Introduced the concept of broadcasting aircraft position, velocity, and identification based on GPS data. The **Navigation Uncertainty Category for Position (NUCp)** was the only means to indicate the accuracy or integrity of horizontal position data. The reliance solely on NUCp for integrity reporting limited the system's ability to provide detailed information about navigation accuracy and integrity.

1. **Version 1: DO-260A**

Introduced distinct parameters for reporting accuracy and integrity, including:

* + **Navigation Accuracy Category for Position (NACp)**: Indicates the accuracy of position data.
  + **Navigation Integrity Category (NIC)**: Reflects the integrity of navigation data.
  + **Surveillance Integrity Level (SIL)**: Assesses the reliability of surveillance data.
  + Added new status parameters and message types, including those for **Traffic Information Service – Broadcast (TIS-B)** and **Automatic Dependent Surveillance – Rebroadcast (ADS-R).**

**TIS-B** is a service that provides information about non-ADS-B equipped aircraft to those that are equipped with ADS-B. It broadcasts traffic information derived from ground-based surveillance systems, such as radar, to enhance situational awareness in the cockpit. **ADS-R** is a method used to rebroadcast ADS-B information received on one frequency to another frequency, allowing different types of equipped aircraft to see each other. For example, it enables communication between aircraft using the 1090 MHz Extended Squitter (for commercial aviation) and those using the Universal Access Transceiver (UAT) at 978 MHz (primarily for general aviation).

1. **Version 2: DO-260B/ED-102A**

This version was developed based on operational experience gained from earlier versions.

* + **Enhanced Integrity Reporting:** Further separated position source and system integrity reporting, allowing for more detailed assessments of navigation data.
  + **Additional NIC Levels:** Introduced more levels of NIC to better support both airborne and surface applications.
  + **Mode A Code Broadcasting:** Incorporated the ability to broadcast Mode A code in emergency/priority messages, increasing transmission rates after a Mode A code change.
  + Expanded parameters in Target State and Status Messages to provide more comprehensive traffic information.

1. **Version 3: DO-260C**

This version focuses on optimizing data transmission and enhancing functionality for modern air traffic management needs.

* + **Interval Management Support:** Enhanced capabilities for managing flight intervals, allowing for better spacing between aircraft during approach and landing.
  + **Weather Data Broadcasting:** Introduced optional messages that can report additional weather parameters such as wind speed, temperature, and icing status.
  + **Improved Receiver Performance:** Enhanced capabilities for track initiation and maintenance, reducing response times for both airborne and surface traffic tracking.
  + **Autonomous Distress Tracking (ADT):** Added functionality to automatically transmit an aircraft's position at least once per minute when in distress, improving search and rescue operations.
  + Support for new applications such as wake turbulence avoidance and hazardous weather detection.

|  |  |
| --- | --- |
| Version | Key Features & Changes |
| Version 0 | Initial standard; relied solely on NUCp for integrity. |
| Version 1 | Introduced NACp, NIC, SIL; added TIS-B/ADS-R messages. |
| Version 2 | Enhanced integrity reporting; more NIC levels; Mode A broadcasting. |
| Version 3 | Interval management support; weather data broadcasting; autonomous distress tracking. |

**Steps to Verify the ADS-B Version**

1. **Step 1**: Determine if the aircraft is transmitting ADS-B messages with Type Code (TC) 31. If no such messages are detected, it can be concluded that the aircraft is operating on Version 0.
2. **Step 2**: If messages with TC=31 are present, examine the version numbers found in bits 41–43 of the Message Element (ME) or bits 73–75 of the data packet.

Once the ADS-B version for an aircraft is identified (which typically remains constant), you can proceed to decode the associated TC=28 and TC=31 messages accordingly.

## ADS-B message receiver

This section outlines the hardware and software setup for an ADS-B receiver using an RTL-SDR (Software Defined Radio) connected to a Raspberry Pi. The setup is designed to receive and decode raw ADS-B messages transmitted at 1090MHz. The receiver setup has been made with low-cost off-the-shelf components along with open-source software tools to support the extraction and decoding of data from Mode S signals.

### Receiver Range:

A diagram of a curved object

Description automatically generated with medium confidenceMode S uses L band signals that follow the line-of-sight propagation, any obstructions between the transmitter and receiver can cause a significant number of signals to be blocked. If no obstacle exists between the aircraft and the receiver, and the transmitter has enough power, the maximum range of the receiver is determined by the curvature of the earth.

The maximum distance () of a Mode S receiver can be obtained by knowing the altitude of the receiver antenna:

where is the radius of the earth, while and are the height of the aircraft and receiver above the sea level. Using this equation, we can calculate the maximum receiving range for aircraft flying at different altitudes. However, in real-life applications, Mode S signal follows the Friis transmission model, which states that the maximum distance also depends on the power of the transmitter, as well as the directivities of the transmission and receiving antennas. When considering these factors, the actual radio range for the receiver is typically lower than the theoretical values calculated from the equations mentioned.

### Antenna Setup

In principle, any antenna designed for the radio frequency around 1 GHz can be used for receiving Mode S signals. The carrier frequency of Mode S is 1090 MHz, which corresponds to the wavelength of 27.5 centimetres. To have an antenna that is tuned to this specific frequency, a simple metal wire (conductor) and a coaxial feeder cable was used to design a dipole antenna. A monopole antenna can also be used for the same purpose. The monopole antenna and the dipole antenna both are half-wavelength (λ/2) antennas with a total conductor length of 13.75 cm.

### Receiver Setup

The receiver is built using a RTL-SDR. SDRs allow the user to select the center frequency of the receiver and the sampling rate, as well as define the bandwidth of the onboard low-pass filter if it is supported. An RTL-SDR Blog V4 R828D RTL2832U 1PPM TCXO HF Bias Tee SMA Dongle was used with a Raspberry Pi 3 to capture the Mode S signals. It can work with radio frequencies from 24 to 1766 MHz. The maximum sampling rate is 2.8 million samples per second (MSPS). This SDR is only capable of listening in the frequency ranges (can receive radio signals but cannot transmit radio signals). The dipole antenna was mounted at an elevated position to maximize reception range. An active antenna with a Low-Noise Amplifier (LNA) can be used to further reduce the signal-to-noise ratio.

The Raspberry Pi runs a basic copy of the Raspbian Operating System. Several software tools are available to work with some of these SDR devices and decode Mode S and ADS-B signals directly.

#### dump1090

dump1090 [4] is one of the most frequently used open-source Mode S decoder. It also offers embedded HTTP server that displays the currently detected aircrafts on Google Map, along with TCP server streaming and receiving raw data to/from connected clients (using --net). Additionally, it provides an interactive command-line-interface mode where aircrafts currently detected are shown as a list refreshing as more data arrives. It also supports CPR coordinates decoding and track calculation from velocity.

1. The directory was cloned (downloaded) to a local source as shown:

$ git clone https://github.com/antirez/dump1090.git

1. Installing the dependencies:

sudo apt-get install build-essential debhelper librtlsdr-dev pkg-config dh-systemd libncurses5-dev libbladerf-dev

1. Compile the directory

$ cd dump1090

$ make

The successful compilation marks the complete installation of dump1090. Once it is compiled and the RTL-SDR receiver is connected, the following command starts receiving and decoding the signals:

$ ./dump1090

With the default option, Mode S messages and decoded information are displayed in the terminal.

1. \*8d451dbd9905b5018004005979c5;

2. CRC: 000000

3. RSSI: -20.6 dBFS

4. Score: 1800

5. Time: 9214131.08us

6. DF:17 AA:451DBD CA:5 ME:9905B501800400

7. Extended Squitter Airborne velocity over ground, subsonic (19/1)

8. ICAO Address: 451DBD (Mode~S / ADS-B)

10. Air/Ground: airborne

12. Ground track 271.4

14. Groundspeed: 436.1 kt

16. Geom rate: 0 ft/min

18. NACv: 0

To view Raw Mode S messages, the –raw option can be used

$ ./dump1090 –raw

The terminal output will only have the raw ADS-B messages.

\*5d4074358ad00c;

\*8d407435990dbd01900484f66c3c;

\*8d40743558af828cd326fe0c2fe9;

\*a80011b1e0da112fe0140060939f;

\*a00015b8c2680030a80000318667;

\*5d4074358ad030;

\*5d4074358ad030;

dump1090 can also provide a live view of all aircraft seen by the receiver using the –interactive option:

A table with numbers and a few words

Description automatically generated with medium confidence$ ./dump1090 –interactive

#### pyModeS

PyModeS is an open-source Python library developed by Junzi Sun and contributors from TU Delft's Aerospace Engineering Faculty for decoding Mode-S and ADS-B signals [5]. The generated sample ADS-B messages was verified using pyModeS. The stable version of pyModeS can be installed as:

pip install --upgrade pyModeS

A screenshot of a computer error

Description automatically generatedpyModeS provides the possibility to view live traffic through the modeslive command.

For this project, the RTL-SDR was used to fetch traffic data from dump1090. The dump1090 data source was then supplied through a TCP output stream using these commands –

$ dump1090 --net --quiet

$ modeslive --source net --connect localhost 30002 raw

The functionalities of this are further broken down to use the low-level decoding functionalities to decode and verify the generated ADS-B messages.

Core functions of pyModeS can be used to decode Downlink Format, ICAO address, ADS-B Type Code, as well as to perform parity check:

import pyModeS as pms

pms.df(msg) # Downlink Format

pms.icao(msg) # Infer the ICAO address from the message

pms.crc(msg) # Perform parity check

pms.typecode(msg) # Obtain ADS-B message Type Code

ADS-B related functions allow information such as identity, position, and velocities to be decoded.

# position messages

pms.adsb.position(msg\_even, msg\_odd, t\_even, t\_odd)

pms.adsb.altitude(msg)

# velocity messages

pms.adsb.velocity(msg)

There are also several functions designed to infer and decode Mode S downlink messages.

pms.common.altcode(msg) # Mode S altitude code (DF=4/20)

pms.common.idcode(msg) # Mode S squwak code (DF=5/21)

pms.bds.infer(msg) # Infer Modes S BDS code

Once a Mode S message type is identified, type specific functions can be used to decode corresponding parameters.

# BDS 4,0

pms.commb.selalt40mcp(msg) # MCP/FCU selected altitude (ft)

pms.commb.selalt40fms(msg) # FMS selected altitude (ft)

pms.commb.p40baro(msg) # Barometric pressure setting (mb)

# BDS 5,0

pms.commb.roll50(msg) # Roll angle (deg)

pms.commb.trk50(msg) # True track angle (deg)

pms.commb.gs50(msg) # Ground speed (kt)

pms.commb.rtrk50(msg) # Track angle rate (deg/sec)

pms.commb.tas50(msg) # True airspeed (kt)

# BDS 6,0

pms.commb.hdg60(msg) # Magnetic heading (deg)

pms.commb.ias60(msg) # Indicated airspeed (kt)

pms.commb.mach60(msg) # Mach number (-)

pms.commb.vr60baro(msg) # Barometric altitude rate (ft/min)

pms.commb.vr60ins(msg) # Inertial vertical speed (ft/min)

These are some commonly used functions related to Mode S decoding. A complete list of the APIs can be found in the pyModeS library API documentation.

## ADS-B Message Types and Creation

ADS-B messages can be broadly categorized into several types, each serving a specific purpose in transmitting aircraft information.

### Aircraft identification and Category

A line of black letters and numbers

Description automatically generatedADS-B messages include a specific type known as the Aircraft Identification and Category message. This message serves two primary purposes: it transmits the aircraft's identification, commonly referred to as the callsign, and communicates the aircraft's wake vortex category. By broadcasting this information, the message enables air traffic control systems and other aircraft to accurately identify and categorize the transmitting aircraft. In this message, the Type Code can be from 1 to 4. The 56-bit ME filed consists of 10 parts and is structured as follows:

#### Identification (Callsign)

The aircraft identification included in the message is the callsign. A callsign is not a unique identifier of an aircraft, since different aircraft flying the same route at different times would share the same callsign.

The last eight fields (C1 to C8) in the previous structure diagram represent the callsign characters. The characters are mapped based on a lookup table which maps the corresponding decimal number (represented in binary code) to each character. The character mapping is as follows -

A black background with white numbers

Description automatically generatedThe # symbols represent characters that are not used. The character and their decimal representations are as follows –

The ␣ symbol refers to a space character. It is worth noting that it is easy to identify that a callsign character is encoded using the lower six bits of the same character in ASCII (American Standard Code for Information Interchange) code.

#### Wake Vortex Category

The CA value in combination with TC value defines the wake vortex category of the aircraft.

|  |  |  |
| --- | --- | --- |
| TC | CA | Category |
| 1 | ANY | Reserved |
| ANY | 0 | No category information |
| 2 | 1 | Surface emergency vehicle |
| 2 | 3 | Surface service vehicle |
| 2 | 4-7 | Ground obstruction |
| 3 | 1 | Glider, sailplane |
| 3 | 2 | Lighter-than-air |
| 3 | 3 | Parachutist, skydiver |
| 3 | 4 | Ultralight, hang-glider, paraglider |
| 3 | 5 | Reserved |
| 3 | 6 | Unmanned aerial vehicle |
| 3 | 7 | Space or trans-atmospheric vehicle |
| 4 | 1 | Light (less than 7000 kg) |
| 4 | 2 | Medium 1 (between 7000 kg and 34000 kg) |
| 4 | 3 | Medium 2 (between 34000 kg to 136000 kg) |
| 4 | 4 | High vortex aircraft |
| 4 | 5 | Heavy (larger than 136000 kg) |
| 4 | 6 | High performance (>5 g acceleration) and high speed (>400 kt) |
| 4 | 7 | Rotorcraft |

ADS-B has its own definition of wake categories, which is different from the ICAO wake turbulence category definition commonly used in aviation. The relationships of ICAO wake turbulence category (WTC) and ADS-B wake vortex category are:

* ICAO WTC L (Light) is equivalent to ADS-B (TC=4, CA=1).
* ICAO WTC M (Medium) is equivalent to ADS-B (TC=4, CA=2 or CA=3).
* ICAO WTC H (Heavy) or J (Super) is equivalent to ADS-B (TC=4, CA=5).

#### ADS-B Message Creation

A MATLAB function has been defined that generates an ADS-B message for aircraft identification and category. It takes inputs related to the aircraft and constructs a complete ADS-B message, including error checking bits (CRC). The code then converts this message into a Pulse Position Modulation (PPM) encoded signal, which is how ADS-B messages are typically transmitted.

##### Main Function: ADSB\_aircraftID\_category.m

##### Input Parameters:

* DF: Downlink Format (5 bits)
* CA: Capability (3 bits)
* ICAO\_hex: ICAO address in hexadecimal (24 bits)
* type\_code: Message type code (5 bits)
* category: Aircraft category (3 bits)
* aircraft\_id: Aircraft identification/callsign (up to 8 characters)

DF = 17;

CA = 5;

ICAO\_hex = '4840D6';

type\_code = 4;

category = 0;

aircraft\_id = 'KLM1023';

##### Processing Stages:

1. **Binary Conversion:**

* Converts DF and CA to binary strings.
* Converts ICAO address from hex to binary.
* Converts type\_code and category to binary.

DF\_bin = dec2bin(DF, 5);

CA\_bin = dec2bin(CA, 3);

ADS\_B\_message = [DF\_bin CA\_bin];

ICAO\_bin = dec2bin(hex2dec(ICAO\_hex), 24);

type\_code\_bin = dec2bin(type\_code, 5);

category\_bin = dec2bin(category, 3);

1. **Aircraft ID Encoding:**

* Pads aircraft\_id to 8 characters.
* Converts each character to a 6-bit binary representation using charToBinary6bit.

aircraft\_id\_bin = '';

aircraft\_id = pad(aircraft\_id, 8, 'right', ' ');

for i = 1:length(aircraft\_id)

aircraft\_id\_bin = [aircraft\_id\_bin

charToBinary6bit(aircraft\_id(i))];

end

1. **Message Assembly:**

* Concatenates all binary parts to form the complete ADS-B message.

payload\_bin = [type\_code\_bin category\_bin aircraft\_id\_bin];

ADS\_B\_complete = [ADS\_B\_message ICAO\_bin payload\_bin];

1. **CRC Calculation:**

* Calculates CRC (Cyclic Redundancy Check) parity bits using ADSB\_CRC.

ADS\_B\_hex = binaryToHexManual(ADS\_B\_complete);

[parity\_bin, parity\_hex] = ADSB\_CRC(ADS\_B\_hex);

ADS\_B\_with\_parity = [ADS\_B\_complete parity\_bin];

1. **Final Message Formation:**

* Appends CRC bits to the message.

disp('ADS-B Message without Parity (Hexadecimal):');

disp(ADS\_B\_hex);

disp('Parity Bits (Hexadecimal):');

disp(parity\_hex);

ADS\_B\_hex\_final = binaryToHexManual(ADS\_B\_with\_parity);

disp('Final ADS-B Message with Parity (Hexadecimal):');

disp(ADS\_B\_hex\_final);

1. **PPM Signal Generation:**

* Generates a PPM encoded signal using generatePPM.

[ppm\_signal, time\_axis] = generatePPM(ADS\_B\_with\_parity);

figure;

plot(time\_axis \* 1e6, ppm\_signal);

ylim([-0.5, 1.5]);

grid on;

title('PPM Encoded ADS-B Message');

xlabel('Time (μs)');

ylabel('Amplitude');

1. **Saving PPM Signal to File**

outputPath = 'C:\Users\rauna\OneDrive - UW\Study\Project\Summer\_Internship\ADS-B\ADS-B\_WaveGen\ADSB\_Encode\CSV\ppm\_signal.txt';

writematrix([time\_axis', ppm\_signal'], outputPath, 'Delimiter', 'tab');

disp(['PPM signal saved to: ', outputPath]);

##### Supporting Functions

1. **charToBinary6bit**

* Convert a character to its 6-bit binary representation.

function char\_bin = charToBinary6bit(char)

if char >= 'A' && char <= 'Z'

char\_dec = double(char) - double('A') + 1;

elseif char >= '0' && char <= '9'

char\_dec = double(char) - double('0') + 48;

elseif char == ' '

char\_dec = 32;

else

error('Invalid character for aircraft ID');

end

char\_bin = dec2bin(char\_dec, 6);

end

1. **binaryToHexManual**

function hex\_str = binaryToHexManual(bin\_str)

hex\_str = '';

for i = 1:4:length(bin\_str)

nibble = bin\_str(i:min(i+3, length(bin\_str)));

if length(nibble) < 4

nibble = [nibble, repmat('0', 1, 4-length(nibble))];

end

dec\_val = sum(2.^(3:-1:0) .\* (nibble == '1'));

if dec\_val < 10

hex\_str = [hex\_str, char(dec\_val + '0')];

else

hex\_str = [hex\_str, char(dec\_val - 10 + 'A')];

end

end

end

1. **ADSB\_CRC**

* Calculate CRC parity bits for the ADS-B message.

function [remainder\_bin, remainder\_hex] = ADSB\_CRC(data\_hex)

generator\_bin = '1111111111111010000001001';

generator = double(generator\_bin) - '0';

data\_bin = hexToBinaryVector(data\_hex, 88);

data\_bin = [data\_bin, zeros(1, 24)];

for i = 1:(length(data\_bin) - length(generator) + 1)

if data\_bin(i) == 1

data\_bin(i:i+length(generator)-1) = xor(data\_bin(i:i+length(generator)-1), generator);

end

end

remainder = data\_bin(end-(length(generator)-2):end);

remainder\_bin = num2str(remainder);

remainder\_bin = strrep(remainder\_bin, ' ', '');

remainder\_hex = dec2hex(bin2dec(remainder\_bin), 6);

end

1. **generatePPM**

* Generate a Pulse Position Modulation signal from the binary message.

function [ppm\_signal, time\_axis] = generatePPM(binary\_message)

bit\_rate = 1000000;

samples\_per\_second = 20000000;

samples\_per\_bit = samples\_per\_second / bit\_rate;

pulse\_width\_samples = round(0.5 \* samples\_per\_bit);

ppm\_signal = zeros(1, length(binary\_message) \* samples\_per\_bit);

for i = 1:length(binary\_message)

if binary\_message(i) == '0'

start\_index = round((i-1) \* samples\_per\_bit) + 1;

else

start\_index = round((i-1) \* samples\_per\_bit + samples\_per\_bit/2) + 1;

end

end\_index = min(start\_index + pulse\_width\_samples - 1, length(ppm\_signal));

ppm\_signal(start\_index:end\_index) = 1;

end

time\_axis = (0:length(ppm\_signal)-1) / samples\_per\_second;

end

##### Outputs

The output would look something like this:

ADS-B Message without Parity (Hexadecimal):

8D4840D6202CC371C32CE0

Parity Bits (Hexadecimal):

576098

Final ADS-B Message with Parity (Hexadecimal):

8D4840D6202CC371C32CE0576098

PPM signal saved to: C:\Users\rauna\OneDrive - UW\Study\Project\Summer\_Internship\ADS-B\ADS-B\_WaveGen\ADSB\_Encode\CSV\ppm\_signal.txt

* **ADS-B Message without Parity (Hexadecimal):**

This is the hexadecimal representation of the ADS-B message before the parity bits are added. It includes the Downlink Format, Capability, ICAO address, and the encoded aircraft identification.

* **Parity Bits (Hexadecimal):**  
  These are the CRC parity bits calculated for error detection. They are appended to the message to ensure integrity during transmission.
* **Final ADS-B Message with Parity (Hexadecimal):**  
  This is the complete ADS-B message including the parity bits, ready for transmission.
* **PPM signal saved notification:**  
  This confirms that the Pulse Position Modulation (PPM) encoded signal has been saved to a file at the specified location.
* **PPM Signal Plot:**  
  The graph shows the PPM encoded signal over time. Each pulse represents a bit in the ADS-B message, with the position of the pulse indicating whether it's a '0' or '1'.

A barcode with blue lines

Description automatically generatedAdditionally, the code will generate a plot of the PPM encoded signal, which would look something like this:

### Aircraft Airborne Position

# References

|  |  |
| --- | --- |
| [1] | W. A. H. a. K. N. Blythe, "Ads-b implementation and operations guidance document," *International Civil Aviation Organization. ,* 2011. |
| [2] | D. a. B. C. Gray, "Privacy impact assessment (PIA) - federal aviation administration (FAA) privacy ICAO address system.," *U.S. Department of Transportation. ,* 2019. |
| [3] | ICAO, "Technical provisions for mode s services and extended squitter.," *International Civil Aviation Organization. ,* 2008. |
| [4] | antirez, "github/dump1090," [Online]. Available: https://github.com/antirez/dump1090. |
| [5] | J. Sun, "github.com/junzis/pyModeS," [Online]. Available: https://github.com/junzis/pyModeS. |