DESIGN AND CHARACTERIZATION OF A SECURE AUTOMATIC DEPENDENT SURVEILLANCE-BROADCAST PROTOTYPE

THESIS

Benjamin N. Heruska, Captain, USAF

AFIT-ENG-MS-15-M-041

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-15-M-041

DESIGN AND CHARACTERIZATION OF A SECURE AUTOMATIC DEPENDENT SURVEILLANCE-BROADCAST PROTOTYPE

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Engineering

Benjamin N. Heruska, B.S.

Captain, USAF

March 2015

AFIT-ENG-MS-15-M-041

DESIGN AND CHARACTERIZATION OF A SECURE AUTOMATIC DEPENDENT SURVEILLANCE-BROADCAST PROTOTYPE

THESIS

Benjamin N. Heruska, B.S.
Captain, USAF

Committee Membership:

Robert F. Mills, PhD
Chair

Michael R. Grimaila, PhD, CISM, CISSP
Member

Pranav R. Patel, MS
Member

Abstract

The weblog from theaviationist.com on Aug 13, 2014 noted that an Air Force aircraft taking part in strikes against Taliban leaders in Afghanistan was monitored and tracked via open source Internet methods. During the flight, the Automatic Dependent Surveillance-Broadcast (ADS-B) transmitter was unintentionally turned on. In sensitive mission operations, such as air operations in theater, the broadcasting of ADS-B messages poses a serious security concern, but the small message payload of ADS-B transmissions renders encryption standards such as AES unsuitable. Format-preserving encryption, a technique already in use on small message sizes such as credit card numbers, provides a suitable implementation of strong symmetric key encryption for the military context.

This research proposes a Secure ADS-B (SADS-B) system which utilizes a bump-in-the-wire approach for encryption and decryption of ADS-B messages. SADS-B is a proof-of-concept, real-time encryption/decryption system utilizing the format-preserving encryption algorithm FFX. This research details the design of the SADS-B system and characterizes the prototype's performance using the following metrics: detection and error rates, operational latency, and utilization of resources on the underlying field programmable gate array (FPGA) hardware.

Findings include sub-millisecond operational latency for each encryption and decryption, full data rate throughput capability for ADS-B, and approximately 50% utilization of the available FPGA resources. Overall, findings suggest that a layered security system such as SADS-B is suitable for increasing the security of ADS-B use in the military context.

Acknowledgements

A sincere thank you to my advisor, Dr. Robert Mills, for his patient and timely guidance throughout the thesis process. Your flexibility and clear feedback made you invaluable throughout the last several months.

Many thanks also to Mr. Pranav Patel for his dedication to teaching and reteaching digital signal processing and best practices for HDL coding. Your prior industry experience and sound development methods were essential to successful design and analysis, both in the classroom and during the thesis process.

Thank you to Mr. Dave Prentice of AFRL for providing the Aeroflex IFR 6000 baseband signals, upon which many design decisions were made.

Lastly, thank you to my family. To my parents, your love and support throughout my life has been fundamental in developing academic rigor and is without a doubt why I am where I am today. To my wife, you are without a doubt the most selfless person I have ever known and the best friend anyone could ask for, and your support throughout the last 18 months has been unyielding.

Benjamin N. Heruska

Table of Contents

Page

List of Figures

List of Acronyms

1090ES	extended squitter
ACPI	Advanced Configuration and Power Interface
ADC	analog to digital converter
ADS-B	Automatic Dependent Surveillance-Broadcast
AES	Advanced Encryption Standard
AIS	Automatic Identification System
ARSR	Air Route Surveillance Radar
ARTCC	Air Route Traffic Control Center
ATC	Air Traffic Control
ATCSCC	Air Traffic Control System Command Center
AWGN	additive white Gaussian noise
BER	bit error rate
BITW	bump-in-the-wire
CA	Capability
COMSEC	Communications Security
COTS	Commercial-Off-The-Shelf
CRC	Cyclic Redundancy Check
CSV	comma separated value
CUT	component under test
DDC	digital down conversion
DF	Downlink Format
DUC	digital up conversion
FAA	Federal Aviation Administration
FDMA	Frequency Division Multiple Access

FFX	Format-preserving Feistel-based Encryption
FIR	finite impulse response
FL180	Flight Level 180
FNR	false negative rate
FPE	format-preserving encryption
FPGA	field-programmable gate array
FPR	false positive rate
GPS	Global Positioning System
HDL	hardware description language
I	in-phase
IFR	Instrument Flight Rules
IP	Internet Protocol
IP	intellectual property
IPSec	Internet Protocol Security
LPF	low-pass filter
LUT	Look-Up Table
MAC	message authentication code
MSB	most significant bit
NAS	National Airspace System
NAVAID	navigation aid facility
NCO	numerically controlled oscillator
NextGen	Next Generation Air Transportation System
OOK	on-off keying
OPSEC	Operational Security
PC	personal computer
PI	Parity/Interrogator Identity

PKI	public key infrastructure
PPM	pulse-position modulation
Q	quadrature
RAM	random access memory
SADS-B	Secure ADS-B
SDR	Software-Defined Radio
SNR	signal-to-noise ratio
SSL	Secure Sockets Layer
TDMA	Time Division Multiple Access
TNR	true negative rate
TPR	true positive rate
UAV	Unmanned Aerial Vehicle
UCRC	Ultimate CRC
USAF	United States Air Force
USRP™	Universal Software Radio Peripheral
VFR	Visual Flight Rules
VHDL	Very High Speed Integrated Circuit Hardware Description Language
W-AIS	Warship-Automatic Identification System
WAAS	Wide Area Augmentation System

DESIGN AND CHARACTERIZATION OF A SECURE AUTOMATIC DEPENDENT SURVEILLANCE-BROADCAST PROTOTYPE

I. Introduction

1.1 Background and Motivation

On August 10, 2014, a USAF E-11A aircraft circling the Ghazni region of Afghanistan could be tracked by anyone in the world with Internet access [3]. Figure 1 is a screen capture of what an Internet browser would have displayed. According to the website *The Guardian*, there was a coincident NATO attack in the region [12]. The E-11A aircraft was transmitting an Automatic Dependent Surveillance-Broadcast (ADS-B) signal during its flight, which contains its precise location information, thus allowing any nearby ADS-B receiver to be used to track the location of the aircraft during a combat mission.

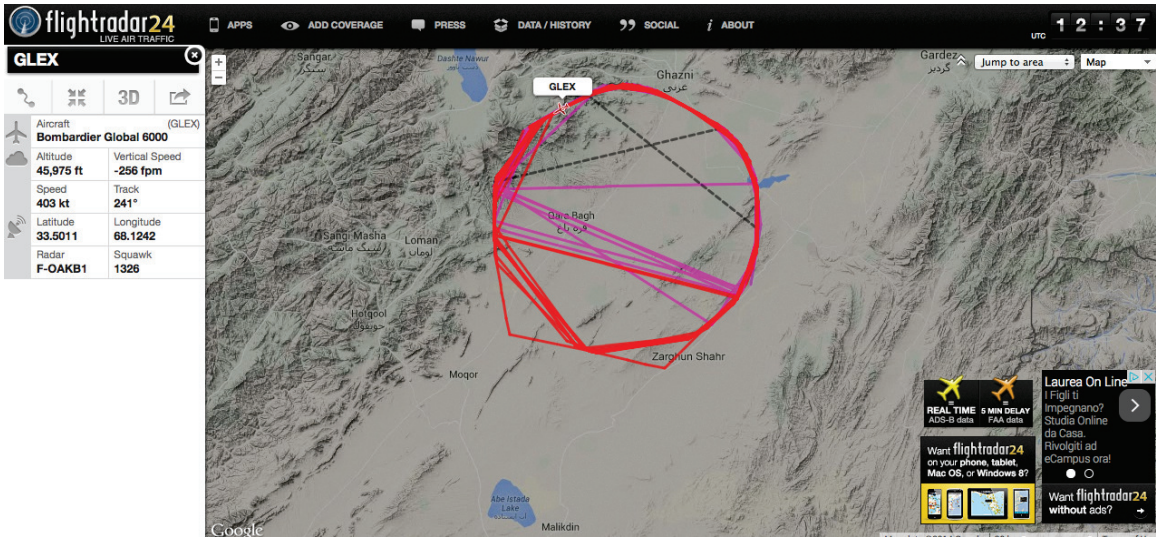


Figure 1. Open source flight track of military aircraft [3].

In sensitive mission operations, such as air operations in theater, the broadcasting of ADS-B messages poses a serious security concern [17, 18, 20]. Previous research has evaluated the possibility of encrypting the data contained in ADS-B messages [1, 11], identifying the expected utility of some various symmetric key encryption algorithms.

ADS-B is one of the major component systems of Next Generation Air Transportation System (NextGen), a phased upgrade program being rolled out by the Federal Aviation Administration (FAA) to improve the robustness of United States National Airspace System (NAS). The benefits of NextGen are notable, as it has the potential to improve the efficiency and safety of commercial flight in the United States. ADS-B specifically increases the precision of positioning information available to everyone equipped, by collecting Global Positioning System (GPS) position information and broadcasting it to the surrounding area.

The United States Air Force (USAF) has identified benefits from implementing and using ADS-B, centered on enhanced safety and mission effectiveness [13]. While commercial aircraft may fly in close proximity out of necessity due to restricted and/or crowded airspace around an airport, military aircraft often intentionally operate at close proximity, sometimes for long periods of time in such mission areas as formation flying and air refueling [13]. Radar is the primary sensor used for automatically retrieving position information about other aircraft, and its precision is weak. The result is the need for verbal adjustments between pilots based on visual contact. ADS-B, if implemented and integrated into planning and tracking and navigation, has the potential to increase mission safety, effectiveness, and efficiency in areas like air refueling.

However, NextGen in such a military context presents security challenges across the fundamental aspects of security: confidentiality, integrity, availability, authenti-

cation, and non-repudiation. The FAA has directly acknowledged security weaknesses in confidentiality and authentication [8]. Some prior work has started to address weaknesses in the underlying ADS-B system, including in a military context. Recently, research has illuminated format-preserving encryption (FPE) as a mechanism to provide confidentiality and also integrity to an extent [10]. Follow-up work was also conducted to evaluate a potential hardware design in simulation [1].

1.2 Problem Statement

This research presents the design, development, and testing of a proof-of-concept Secure ADS-B (SADS-B) prototype, operable in two modes: either (1) as part of the transmit chain, capable of receiving an ADS-B message, encrypting it, and transmitting a SADS-B message in its place, or (2) as part of the receive chain, capable of receiving a SADS-B message, decrypting it, and transmitting the original ADS-B message.

1.3 Research Objectives

The goal of this research is to design, develop and characterize a complete SADS-B prototype, through the following steps:

- Design and develop SADS-B prototype system
- Verify correctness of prototype implementation (system and component-level)
- Evaluate the performance of prototype
  - latency (system and component-level)
  - preamble synchronization component
  - bit-level pulse-position modulation (PPM) demodulator

- Evaluate required field-programmable gate array (FPGA) resources for full data rate implementation of Format-preserving Feistel-based Encryption (FFX)

1.4 Approach

The research objectives are approached systematically using an incremental approach through a series of analysis, modeling and simulation, hardware simulation, and hardware verification. The system builds on previous research conducted by Finke [11], which validated a suitable FFX implementation. To complete the full bump-in-the-wire (BITW) concept, additional digital components are necessary. For these components, the MATLAB® 2014b programming environment was used to develop, analyze and model finite impulse response (FIR) designs for synchronization, demodulation, Cyclic Redundancy Check (CRC) validation, packet generation, and modulation. The hardware design for each component is then developed, including fully unrolled encryption and decryption engines. Using Active-HDL® 9.3 and ModelSim® 10.4, each component design is then simulated and verified, followed by the end-to-end design. Finally, Xilinx ISE 14.7 Design Suite is used to implement the verified design in hardware on a Kintex-7K410T FPGA, onboard a Universal Software Radio Peripheral (USRP™) X310 which normally operates as a Software-Defined Radio (SDR). The top level FPGA design of the X310 is modified to support standalone operation, more representative of the BITW use case. Additionally, the FPGA resources (Look-Up Tables (LUTs), registers and memory) consumed by the SADS-B modules are evaluation using results reported in ISE Design Suite.

To assess performance, external analysis of the hardware systems using an Keysight N5182B MXG Vector Signal Generator and a Teledyne LeCroy 760Zi-A oscilloscope are used to retrieve a true operational latency of the system. Finally, a

second USRP™ is used alongside the N5182B to assess the operational throughput. The results are assessed in the context of ADS-B specifications [21] and in comparison to the findings of previous research utilizing an iterative looping design [1].

1.5 Organization

Chapter II provides an in-depth literature review on the use of ADS-B within NextGen, discusses previous research which addressed security issues with NextGen, and the hardware platform on which the SADS-B prototype will be developed. Chapter III details the SADS-B prototype system, as well as underlying algorithms developed during the course of this research. Chapter IV first describes the methodology for verification and assessing the performance of the SADS-B prototype. Then it presents the results, discussion, and analysis for verification and performance. Lastly, Chapter V contains a summary of contributions as a result of this research, conclusions drawn from it, and recommendations for future SADS-B research.

II. Background

This chapter presents necessary background information to provide a context for assessment and characterization of a SADS-B prototype. It summarizes the emergence of ADS-B at part of systemic upgrade to Air Traffic Control (ATC) in the United States, followed by details about previous work in augmenting the security of ADS-B including encryption techniques. Last, this chapter provides an overview of the hardware platform on which the prototype is developed and implemented.

2.1 National Airspace System

The primary purposes of ATC are to prevent collisions between aircraft and to provide a safe, orderly and expeditious flow of air traffic [9]. For flights under the 18,000 ft level, called Flight Level 180 (FL180), the pilot has the responsibility to maintain safe flying distance from other aircraft. This lower altitude traffic is also referred to as Visual Flight Rules (VFR) traffic. For flights above FL180, called Instrument Flight Rules (IFR) traffic, the federal government is responsible for ATC and has been since the Federal Aviation Act of 1958. The FAA is the organization delegated the responsibility of handling ATC, and they use the NAS system to do so. The NAS is a collection of facilities, system, equipment, procedures, and airports operated by thousands of ground-based personnel [7]. There are 690 ATC facilities with associated systems and equipment to provide radar and communication service, divided into 21 regions, each with their own Air Route Traffic Control Center (ARTCC). The Air Traffic Control System Command Center (ATCSCC) is the headquarters over all the ARTCCs. There are two main influences that drive decisions: traffic congestion and weather. When actions need to be taken to mod-

ify traffic flow, it happens through airline personnel, traffic management specialists, and air traffic controllers.

For the NAS to be effective, it needs location information (position, heading, altitude) for all IFR traffic, at all times. This has been primarily accomplished through the use of radar (providing range and azimuth data) and Mode C (providing altitude information). However, there are significant limitations to this approach. First, voice communication with a pilot are the means of control for an air traffic controller. Second, long-range Air Route Surveillance Radars (ARSRs) can only typically cover a 200 NM radius and are expensive to maintain [7]. There are only about 100 ARSRs in operation in the United States. Additionally, navigation aid facilities (NAVAIDs) provide beacons that increase the available airspace, providing a means to connect between radar sites. Nonetheless, flights at IFR altitudes are restricted to routes that overfly NAVAIDs or ARSRs, leading to many funnel points in the network. This is especially true near airports, where there are often only one or two NAVAIDs for departing aircraft to egress past.

Radar only refreshes approximately once every 12 seconds, and has a margin of error of 300m. A more accurate and accessible source of position information is GPS, which has been around since 1973. When GPS is augmented with ionospheric correction data from Wide Area Augmentation System (WAAS), it can accurately estimate position within 2 meters [7]. Additionally, all IFR aircraft are already required by the FAA to have GPS receivers onboard.

2.2 NextGen

The future vision of the Next Generation Air Transportation System (NextGen) program relaxes many of the restrictions on IFR flight due to mechanisms that reduce reliance on radar. Several technologies are needed to achieve this end vision.

NextGen includes technologies such as performance based navigation, low visibility aids, and electronic flight bag [23]. It also establishes data communications that reduce the amount of required verbal communication between air traffic controllers and pilots. Finally, NextGen adds a system that automatically delivers high precision GPS data to both pilots and air traffic controllers. This system is called Automatic Dependent Surveillance-Broadcast (ADS-B), and it has the potential to eliminate, or at least drastically reduce, reliance on radar.

2.3 ADS-B

As the name suggests, ADS-B *automatically* transmits position information, *dependent* on GPS, to *surveillance stations* (i.e. air traffic controllers) via data *broadcast*. ADS-B consists of two major subsystems: ADS-B Out and ADS-B In. ADS-B Out is the broadcast message (by an aircraft, vehicle, or ground radio station) containing an aircraft identifier and the GPS-based latitude, longitude, and altitude positioning information. ADS-B In is the receiving end, implemented by air traffic controllers and aircraft alike, enabling reception and display of nearby vehicles without reliance on radar.

Because ADS-B Out was designed to be universal, it is not encrypted in any way. The message format for an ADS-B Out broadcast is shown in Figure 2. The cleartext transmission of this data presents a security risk. As described in Section 1.1, this risk is especially true in military operations that are sensitive, such as combat operations.

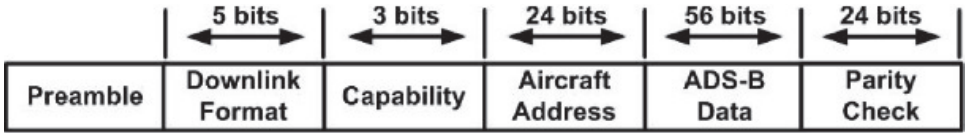


Figure 2. ADS-B DF-17 message format [21].

2.4 Related Work

Prior research efforts have made progress towards more secure implementations of ADS-B, specifically utilizing the DF-17 message which is the primary ADS-B Out message in use by the USAF.

The Communications Security (COMSEC) vulnerabilities associated with ADS-B were identified by Purton et al [20], and later expounded upon by McCallie [18], elaborating on the Operational Security (OPSEC) issues with a military-specific focus. Magazu illustrated the simplicity of performing an injection attack using SDR, an inexpensive Commercial-Off-The-Shelf (COTS) piece of equipment [17].

The path to encryption began with Jochum [15], who proposed using symmetric key cryptography to encrypt the message payload during sensitive operations and tailored the recommendation around the use of the DF-19 header code, which is already reserved for military use in the ADS-B specifications [21]. Finke affirmed the plausibility of using this type of cryptography, and analyzed the theoretical implementation of FPE to handle the unusual bit width of messages [11]. Finally, Agbeyibor confirmed the strength of three different FPE algorithms, simulating an FPGA-based design and assessing expected FPGA resources required.

2.5 Format Preserving Encryption

The need to encrypt cleartext communications containing position information in a military context is not new. Automatic Identification System (AIS) is a commercial system used to receive and transmit position information for sea vessels, similar to the function of ADS-B for aircraft. Like ADS-B, messages are transmitted in cleartext, but with a message length of 128-bits. To keep its transmissions confidential, NATO created Warship-Automatic Identification System (W-AIS), which encapsulates and encrypts one or more AIS messages as its payload (in incre-

ments of 128-bits, up to a maximum of 896 bits), prior to broadcast [14]. Other W-AIS receivers are able decrypt the messages (using a symmetric key) and relay the identification and position information to the display console. W-AIS may directly use Blowfish or Advanced Encryption Standard (AES) algorithms, as both algorithms have been approved and are well suited for the 128-bit AIS message length.

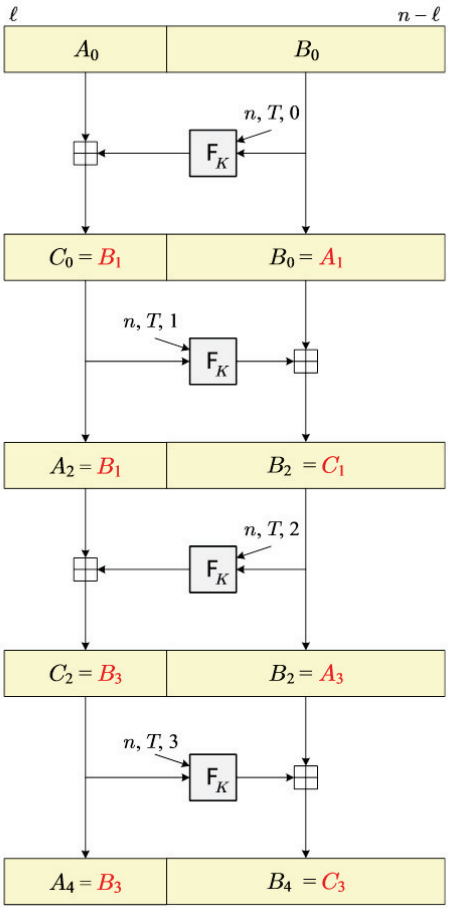


Figure 3. FFX encryption structure [2].

However, ADS-B has a much shorter 112-bit fixed-length message, requiring the use of format-preserving encryption (FPE). In general, FPE algorithms are not dependent on a specific message size or format to be effective. More specifically, they will generate a ciphertext in the same radix and number of digits as the plaintext input, regardless of the key size utilized. The FPE algorithm evaluated

by Finke for use with ADS-B is FFX [11], shown in its generalized form graphically in Figure 3. F_K (as shown in the Algorithm 1) is a one-way function that gives the encryption method its strength.

Algorithm 1 Function F using CBC-MAC of AES [11]

function F(n, T, i, B)
  $vers \leftarrow 1$; $t \leftarrow |T|$ in bytes
  $P \leftarrow [vers]^2 \parallel [method]^1 \parallel [addition]^1 \parallel [radix]^1 \parallel [n]^1 \parallel [split(n)]^1 \parallel [rnds(n)]^1 \parallel [t]^8$
  $Q \leftarrow T \parallel [0]^{-t-9 mod 16} \parallel [i]^1 \parallel 0^{64-|B|} \parallel B$
  $Y \leftarrow$CBC-MAC$(P \parallel Q)$
  return $Y[129 - split(n)..128]$
end function

function CBC-MAC$(m_1 \parallel m_2)$
  $X \leftarrow \text{AES}_K(m_1 \text{ XOR } 0)$
  $Y \leftarrow \text{AES}_K(m_2 \text{ XOR } X)$
  return Y
end function

▷ Note: $[s]^i$ denotes the i-byte string that encodes the number s

The function F_K used for FFX by Finke is an implementation of AES. The 128-bit implementation used performs a truncation step, making it a one-way function. The implementation is maximally balanced, with 52 bits in each half of the Feistel structure, as shown in Figure 4.

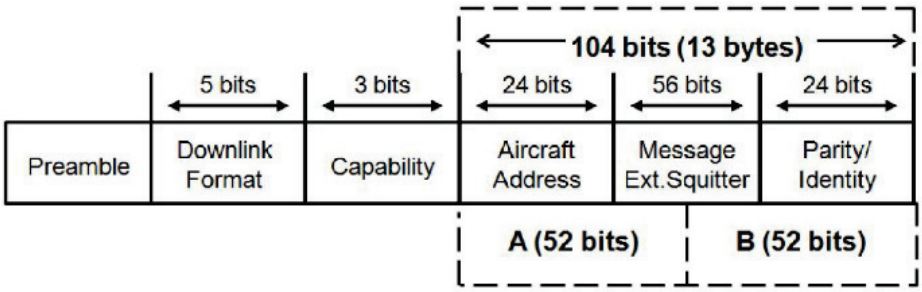


Figure 4. FFX bits superimposed on ADS-B DF-17 message [11].

2.6 USRP™ X310

Ettus Research's USRP™ X310 is a highly customizable FPGA-based radio receiver transmitter. The USRP™ devices typically function as a SDR, as depicted in Figure 5. That is, they function as a slave to a master personal computer (PC) running GNU Radio, which controls the sample rate, tuning, digital down conversion (DDC), and digital up conversion (DUC) in real time. All other processing (such as a generating a spectrogram from the data) is accomplished on the master PC. For example, in receive mode, after the DDC chain, in-phase (I) and quadrature (Q) samples are transmitted over the Ethernet connection (see Figure 5).

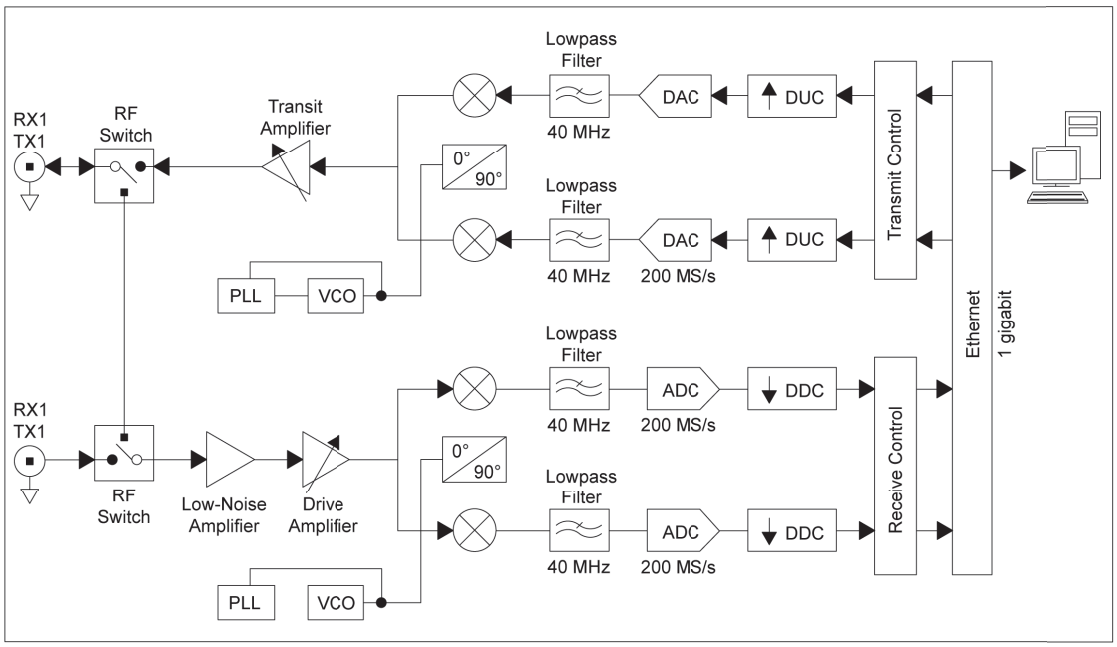


Figure 5. USRP™ X310 receive and transmit processing chains [6].

A similar model (the USRP™ N210) was previously used by Magazu [17] in creating a fake flight track of ADS-B messages.

The X Series, shown in Figure 6, are higher sample rate devices with a more capable FPGA, the Xilinx Kintex-7. Additionally, Ettus Research has made the

Figure 6. USRP™ X310 [6].

hardware description language (HDL) source code available for modification. Written in Verilog, the availability of the source code is essential, as a PC-peripheral device would suit the intended purpose as an avionics prototype. To overcome this limitation, the SADS-B module will be developed in Very High Speed Integrated Circuit Hardware Description Language (VHDL) and implemented alongside the provided source code.

III. System Design and Implementation

This chapter begins with a high-level concept and minimum performance requirements for the SADS-B prototype system, based on ADS-B specifications [21]. It then delves into the design specifics and implementation details of the system.

3.1 BITW Concept

Along the lines of W-AIS, SADS-B will modify a normal ADS-B message (as in Figure 2 on page 8) by encrypting the bits shown in Figure 4 on page 11. In a final implmentation, this would most likely be implemented by modifying an existing ADS-B transponder, but for proof of concept purposes and to provide an interim approach, a BITW approach is suitable.

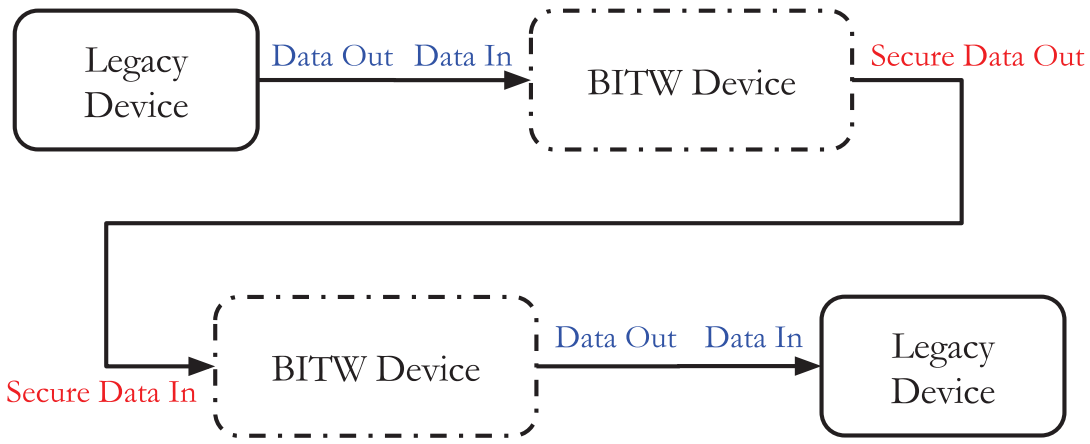


Figure 7. BITW concept diagram.

As defined in IETF RFC4949, bump-in-the-wire (BITW) is "An implementation approach that places a network security mechanism outside of the system that is to be protected" [22]. Figure 7 illustrates a system with a set of paired BITW

devices that perform a security function between two legacy devices. The legacy devices need no special configuration or context in order to communicate through the BITW device. A widely used implementation of a BITW security approach is Internet Protocol Security (IPSec) devices between Internet Protocol (IP) routers that need to traverse an untrustworthy connection. The packets are transmitted to the IPSec devices, which encapsulate and encrypt the received packet before transmitting it across the Internet.

In the context of ADS-B transmission, the ADS-B Out is intercepted by the BITW encryptor, translating the signal to a secure signal, SADS-B Out. The reverse occurs on the receiving end. The proof-of-concept prototype resulting from this research serves either as the SADS-B Encryptor, SADS-B Decryptor, or both.

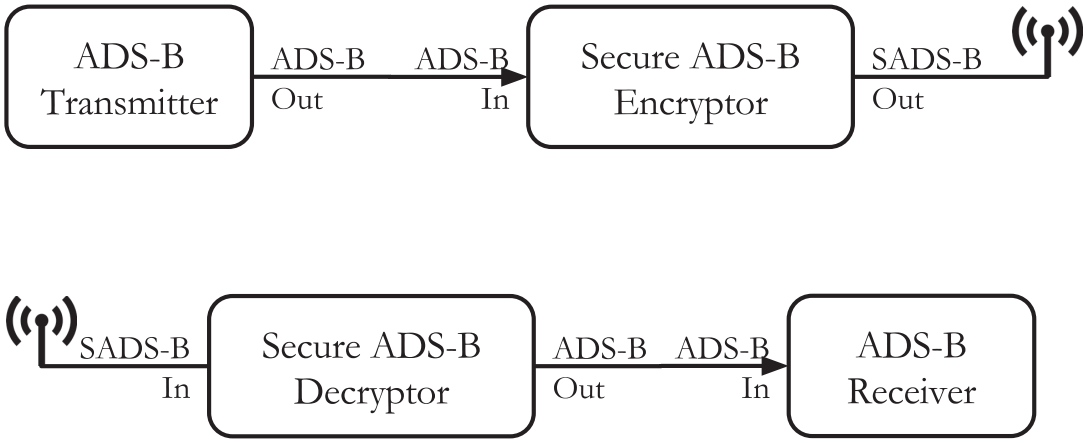


Figure 8. BITW approach to securing ADS-B.

Because the BITW encryption and decryption only apply to specific messages (DF-17) messages, an important option is the pass-through of the received signal. This would allow the pass-through of all other messages (ADS-B or other) on the 1090 MHz frequency.

3.2 Latency Requirement

The ADS-B system is not packet-based, as it functions through broadcast messages. The messages are also time-sensitive, as the FAA DO-260B specification [21] specifies a 100 ms window for ADS-B position messages. Further, the receiver itself is specified as having ± 50 ms tolerance, which leaves only 50 ms maximum allowable operational latency for the SADS-B BITW device.

The in-line BITW architecture, low latency, and pass-through capability requirements for an effective SADS-B prototype support the use of an FPGA-based hardware environment.

3.3 Sampling Rate

For any digitally sampled signal, the minimum required sampling rate, $F_{s,min}$, is twice the analog bit rate, according to the sampling theorem. In the case of PPM, two samples are simply needed to determine the location of the pulse: in the first or second half of a bit frame. In ADS-B, the PPM bit rate is 1 Mb/s, so the required $F_{s,min}$ is 2 MS/s.

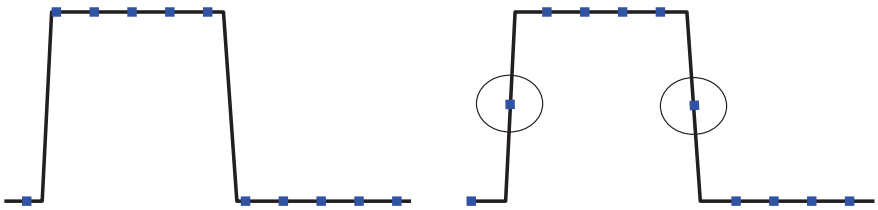
Oversampling is an effective means to reduce receiver noise in digital systems. Oversampling effectively increases the resolution bandwidth $\beta = \frac{F_s}{2}$, causing the noise N_0 to reduce, per Equation 1, where N is the noise power.

$$N = N_0\beta \tag{1}$$

It can also be construed in terms of noise power per sample N_{sample}, according to Equation 2, where M is the oversampling factor.

$$N_{sample} = N_0M \tag{2}$$

Section I.4.1.1 of DO-260B recommends an oversampling factor of at least 4, but recommends 5 or more based on tests conducted by RTCA, Inc [21]. The SADS-B prototype design in this research follows this recommendation with an oversampling factor of 5, such that $F_s = 10$ MS/s. Each pulse width used for sampling is 5 samples. During both preamble detection and demodulation, the edge samples (samples adjacent to transition zones) are ignored. Figure 9 depicts the



(a) convenient sampling across a pulse. (b) inconvenient sampling across a pulse.

Figure 9. Possible conditions during sampling for PPM.

driving factor for this, being the potential for anomalies (i.e. increased noise) in samples in transition zones. As a mechanism to reduce this noise, the samples nearest the transition zones are ignored. This reduces the effective oversampling rate to 3, again following the recommendations from the enhanced detection guidelines in Section I.4.1.1 of DO-260B [21].

3.4 Top-Level Design

Figure 10 shows the top level design of the USRP™ X310, modified for use as the SADS-B prototype. The SADS-B Module as depicted in Figure 10 is the top-level VHDL component, which is the overall focus of this chapter.

The built-in analog components in the USRP™ SBX-40 daughtercard perform coarse tuning for the Mode S carrier frequency of 1090 MHz. The frequency response for the circuit is shown in Figure 49 in Appendix A. This is performed by setting a specific controller on the board, which drives a numerically controlled os-

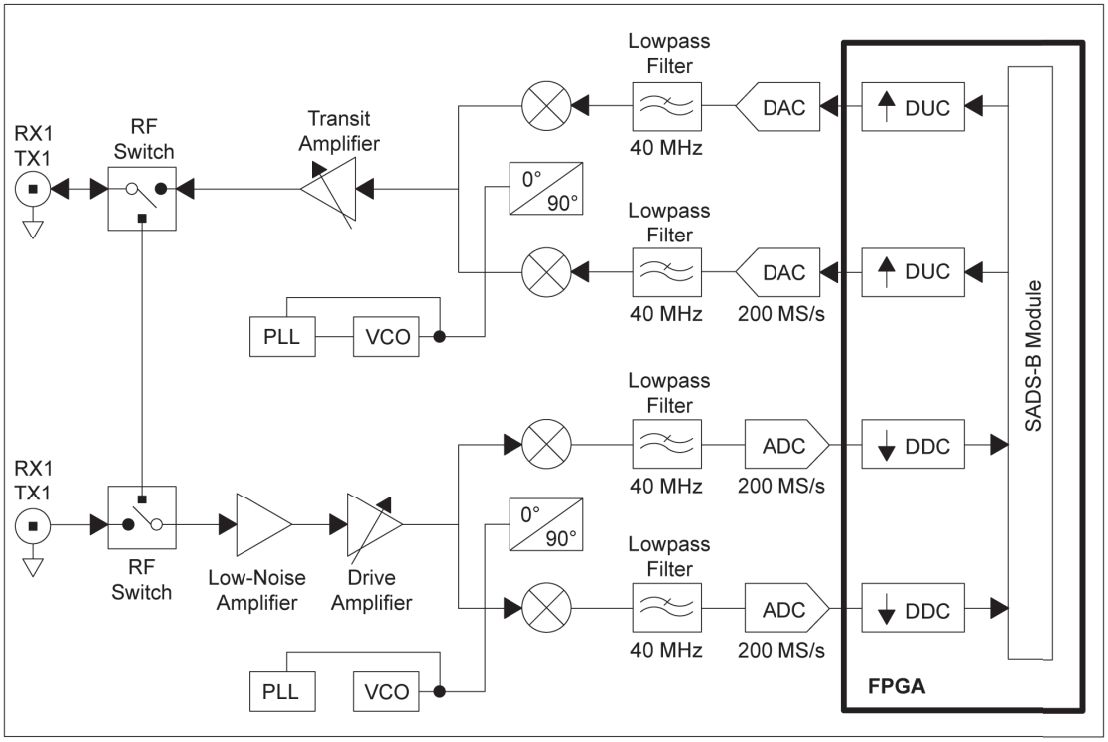


Figure 10. SADS-B top level design. Modified from [6].

cillator (NCO) to mix the 1090 MHz signal down to baseband. It is then passed through a 40 MHz anti-aliasing low-pass filter (LPF) before conversion to digital in the analog to digital converter (ADC) component, from which the I and Q data are transmit to the FPGA at a sampling rate of 200 MS/s. This level of sampling is unnecessary for the 1 Mb/s PPM signal of ADS-B, so the next step is to downsample the the signal.

Internal to the FPGA, Ettus Research has already designed configurable DDC, required for normal operation of the USRP™ when it operates in SDR mode, as previously shown in Figure 5 on page 12. These Verilog-based HDL components are suitable for use in the SADS-B prototype design. In the Ettus Research DDC design, downsample occurs through three successive rounds of filtering through a LPF followed by downsampling. The LPF in each step is set to a stop band frequency

of the $\frac{1}{2}$ the inverse of the downsample rate for that step, as in Equation 3, where F_s is the input sampling rate for that downsample step and M is the downsample factor.

$$B = \frac{F_s}{2M} \tag{3}$$

In the Ettus DDC design, the first two steps are either turned on or off with a hard set downsample rate of 2. The third downsample step is always on, with a configurable integer downsample rate. For example, in the case of a target sample rate of $\frac{200}{20} = 10$MS/s, as discussed in the next section, the steps would be [ON,ON,5], equating to a downsampling factor of $2 \times 2 \times 5 = 20$.

In the output chain, the order of operations is essentially reversed. Three similarly configurable interpolation steps bring the sample rate back up to 200 MS/s from 10 MS/s. First interpolation step is configurable by integer factor, which is again set to 5. The next two steps are a factor of 2, and both are set to "on." After each interpolation step, there is an interpolation filter, which removes high frequency noise. The stop band for this filter is set according to Equation 3, where F_s is the output sampling rate after upsampling.

The FPGA-based SADS-B subsystem, which is the focus of this thesis, consists of two modules: the SADS-B Encryption Module and SADS-B Decryption Module. While similar, there are differences primarily in packetization and CRC validation. The block diagrams are shown in Figures 11 and 12, respectively.

The clock rate for all the components in the design is the native FPGA 200 MHz clock. Every block shown in the functional block diagrams in Figures 11 and 12 have clock inputs (200 MHz) and clock enable inputs (10 MHz) that drive the operation of the components (not shown in the diagram).

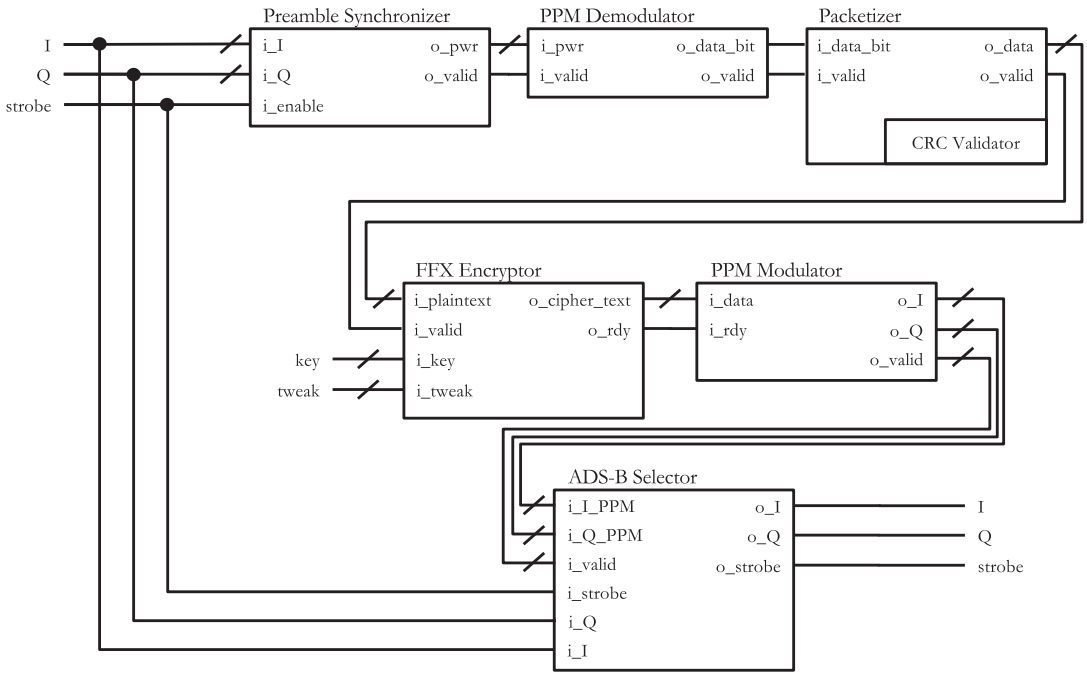


Figure 11. Functional block diagram for the SADS-B encryption module.

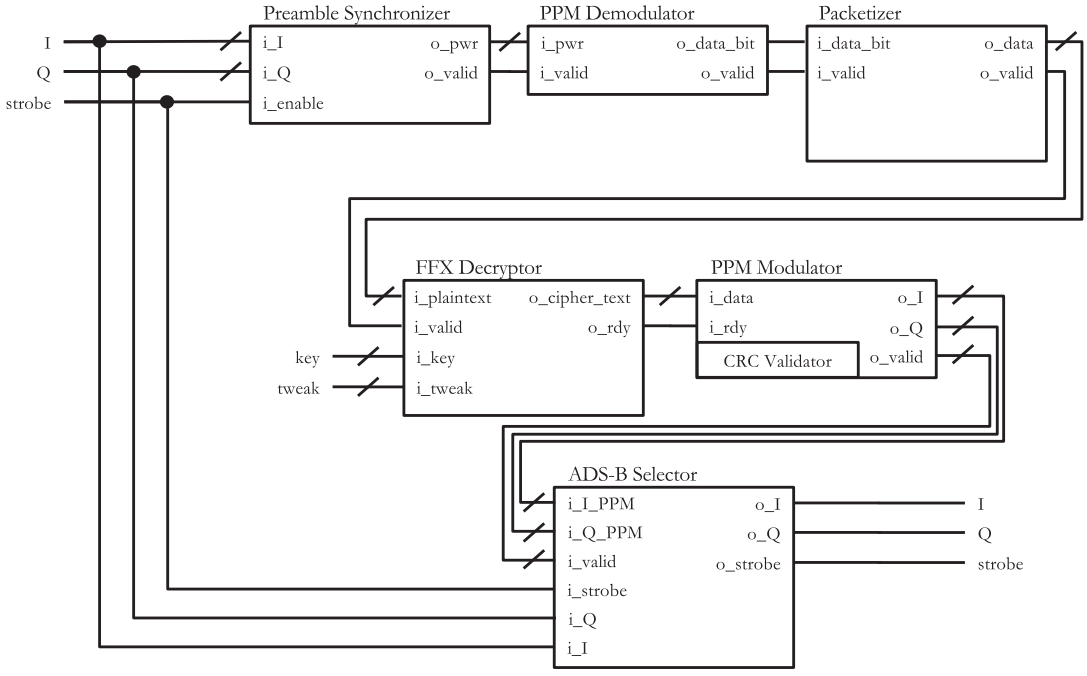


Figure 12. Functional block diagram for the SADS-B decryption module.

3.5 Preamble Detection and Synchronization

The preamble for ADS-B, shown in Figure 13, is 8 μs in duration. It has 4 pulses, with each pulse width the same .5 μs as that of a pulse from the ADS-B data block.

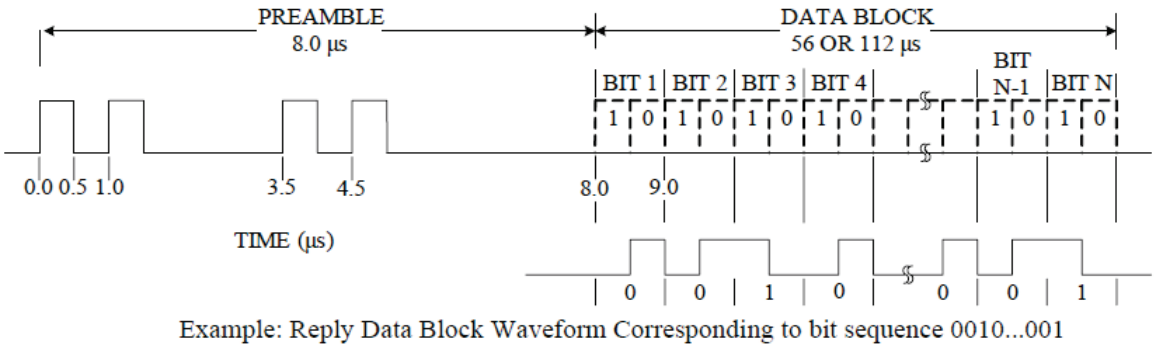


Figure 13. PPM structure of ADS-B preamble and data block [21].

Normally, a matched filter could be used to correlate the incoming signal with the preamble template. The discrete form of a matched filter is shown in Equation 4, given a preamble of length K samples, where $x[k]$ is the value of the k_{th} previous sample, $h[n-k]$ is the k_{th} coefficient in the reverse impulse response of the discretized preamble template, and $y[n]$ is the correlation value at sample n. In this format, $K = T * F_s$ where T is the length of the preamble in seconds and F_s is the sampling rate.

$$y[n] = \sum_{k=0}^{K} h[n-k]x[k] \tag{4}$$

The downsampled sampling rate used for the SADS-B system is 10 MS/s, as discussed in Section 3.3, so $h[n-k]$ would consist of 80 coefficients, the graph of which is horizontally mirrored against the preamble. The preamble detector would trigger the start of the signal, n_{start} where $y[n]$ is at its local maximum above a threshold

correlation value C, according to Equation 5.

$$n_{\text{start}} = n_{y[n],\max} \text{ where } y[n] > C \tag{5}$$

In the case of this preamble, as shown in Figure 14, each of the 0.5 μs pulses would have coefficients with a value of 1. As this is a PPM signal, all other positions are "off" and would have coefficients with a value of 0.

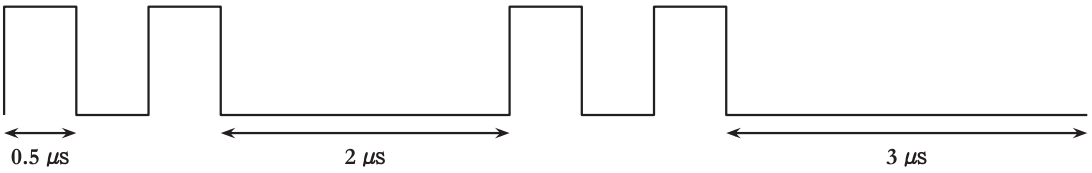


Figure 14. Timing of ADS-B preamble pulses.

However, the use of a matched filter breaks down when using any zero-valued coefficients. In this case, 3/4 of the coefficients are zero, and only 1/4 of the available samples would provide input to the correlation value. This results in false detections, as any received energy between 1070-1110 MHz for 5 μs would appear as a match. To correct for this, all coefficients where $h[i] = 0$ coefficients are converted to $h[i] = -1$. This more correctly utilizes the negative spaces between pulses to assist in the proper rejection of false preambles.

A second consideration for the modified matched filter design is scaling. While a clean signal with high signal-to-noise ratio (SNR) (defined as E_b/N_0) works well with the modified matched filter, there is also a high correlation value at the tail of the ADS-B signal itself, in some cases. The worst possible scenario is shown in Figure 15, where the filter aligns with the end of an ADS-B signal ending in [...,1,1,X,0,0], with X representing either a "1" or a "0". Both cases are shown.

These bits are part of the Parity/Interrogator Identity (PI) field in a DF-17

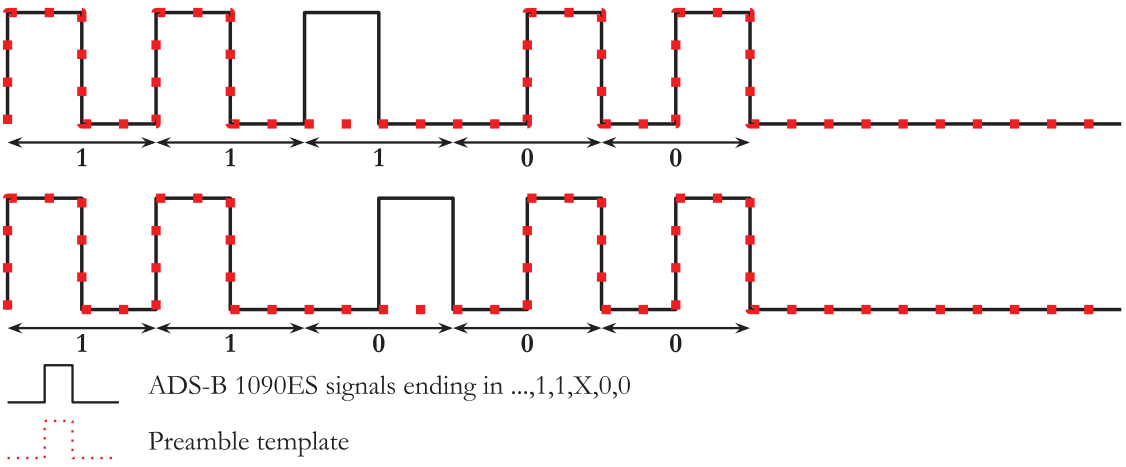


Figure 15. Worst case alignment of preamble with signal for a matched filter.

message, which is a 24-bit CRC. As such, each of the last 5 bits are then uniformly random between 0 and 1. The probability that [1,1,X,0,0] are the last 5 bits of any given ADS-B extended squitter (1090ES) message is $p = \frac{1}{2^4} = 6.25\%$ To be problematic, this situation must have occurred when the proper preamble was not detected. ADS-B has no channel access control, such as Time Division Multiple Access (TDMA) or Frequency Division Multiple Access (FDMA), built in to the protocol, nor is there any collision detection. Therefore, the likelihood of interference during a preamble but not during the end of a signal is non-negligible.

A heuristic to account for this expected occurrence is to adjust the threshold value C. A high threshold value C would result in more of these occurrences being rejected, but also falsely rejects more low-SNR preambles. The inverse would be true in low threshold value C.

An alternative approach is to increase the *penalty* for received signal power in the null (zero power or "off") regions of the preamble template shown in Figure 14. In other words, only scaling the negative coefficients by a penalty factor α and leaving the positive coefficients as +1, resulting in the modified correlation Equation

6.

$$y[n] = \sum_{k=0}^{K} (\alpha(h[n-k]-1)+1)x[k] \tag{6}$$

A consideration for setting the value of α in the preamble detector for SADS-B design is the worst case scenario for a matched filter, shown in Figure 15, where 4 pulses correctly align with the positive coefficients in $h[n-k]$ while 1 pulse is in the null region. Even when using -1 coefficients for the null regions would result in a strong correlation for such a scenario. However, if using a penalty factor set $\alpha > 4$, the single pulse in the null region would negate the otherwise strong correlation value allowing a correct rejection of the false preamble.

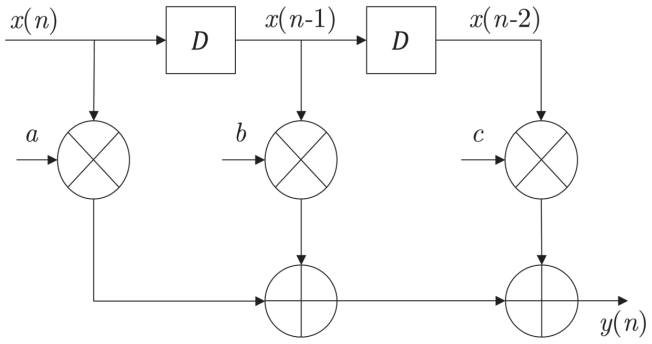


Figure 16. Pipelined FIR filter design [19].

The design utilizes a pipelined FIR filter, with a generic design as shown in Figure 16. In larger pipelined FIR filter designs, the total latency from multiplying and adding may be too great for the clock rate. Given the sampling rate, there is an 80-sample window for preamble correlation, which equates to an 80-order FIR filter. Running on the USRP™ X310's 200 MHz clock, it is not possible to complete a multiply operation and 80 adds in a single clock cycle.

In general, additional registers may be added in a FIR design between some registers and combinational adders, as in Figure 17, or by converting the adder structure from linear to a hierarchical, or chain-out, design. The design for the

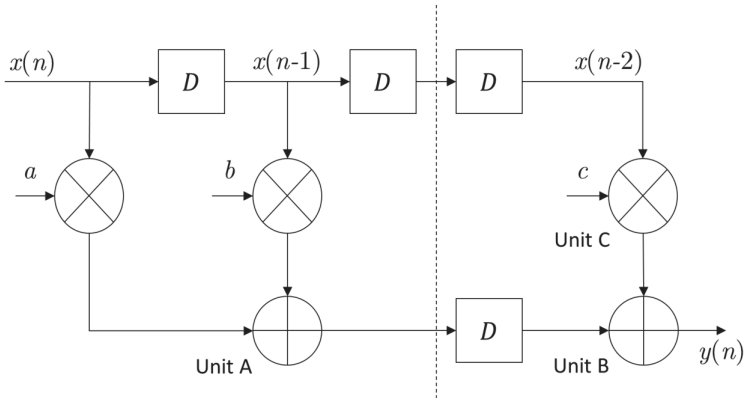


Figure 17. Pipelined FIR filter design with a delay [19].

preamble correlator uses a chain-out design, as shown in Figure 18, with the addition of registers (not shown) between each level of the hierarchy to meet FPGA timing constraints.

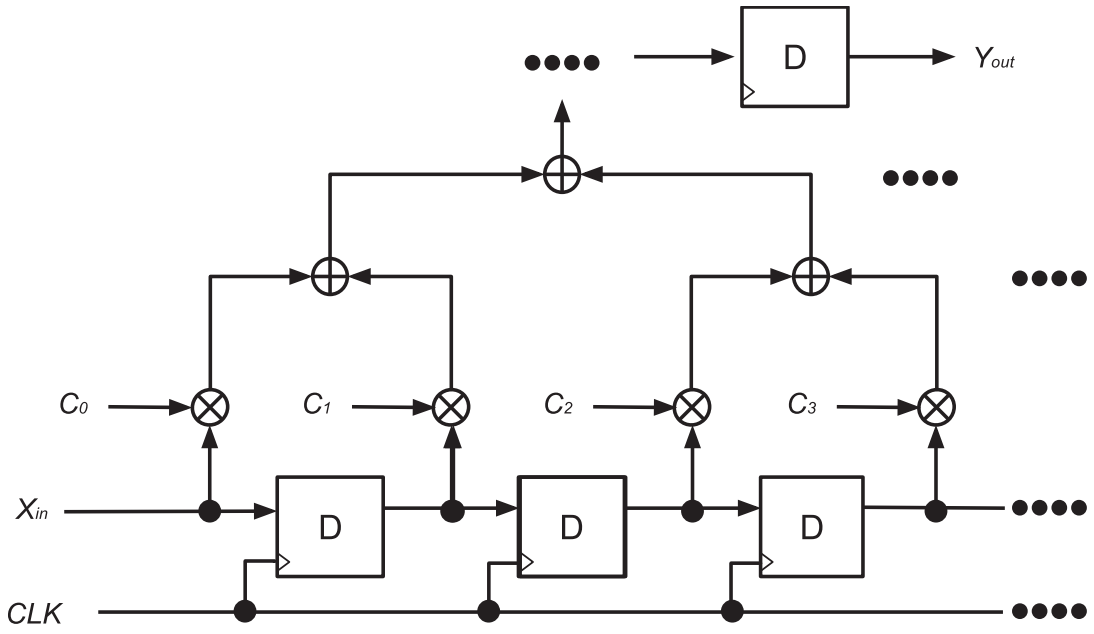


Figure 18. Pipelined FIR chain-out design used for correlator and demodulator.

3.6 PPM Demodulation

In the demodulation step, a simple integrate and dump FIR filter is used, also according to the diagram in Figure 18, with +1 and -1 coefficients according to the '0' PPM bit with the pulse in the first half of the bit frame. The demodulated bit is simply the most significant bit (MSB) of the resulting signed Y_{out} value. As a result of only using the center 3 values of each pulse width (as discussed in Section 3.4), the coefficients for this FIR filter are [0,-1,-1,-1,0,0,1,1,1,0].

3.7 Packetization and CRC Validation

In the SADS-B Encryption Module, the packetizer occurs immediately following the PPM demodulator. It consists of a 112-bit buffer, which shifts in bits as they are output from the demodulator, at a rate of 1 Mb/s or once every 10 samples.

As the inputs are shifted in, they are also input into a CRC validation component, which is an instantiation of the serial version of the Ultimate CRC (UCRC) intellectual property (IP) Core [5]. UCRC is initialized during synthesis with the CRC-24 polynomial in Equation 7, coming from the ADS-B specification [21].

$$P(x) = x^{24} + x^{23} + x^{22} + x^{21} + x^{20} + x^{19} + x^{18} + x^{17} + x^{16} + x^{15} + x^{14} + x^{1} \quad (7)$$

Each time a sample is shifted into UCRC, the computed CRC remainder is updated. When the value of the remainder is 0, the `match_o` signal of UCRC is set high, indicating the CRC matches. An output register in the packetizer is updated whenever `match_o` is set high. Then the output valid signal in the packetizer turns is set high for a clock enable period, enabling the next component, the FFX encryptor.

3.8 FFX Encryption

The encryption design implements the FFX-A2 algorithm according to the parameters radix=2, length=104, method=2, split=52, rounds=12, according to the parameters used in Finke [11]. It does so using a fully-pipelined implementation, in contrast with an iterative looping version in previous research [1]. This has a greater impact on decryption and is discussed in more detail in Section 3.9.

As a valid packet sample is received from the packetizer, as shown in Figure 11 on page 20, it is broken into the header bits (the first 8 bits: the Downlink Format (DF) and Capability (CA) fields), and the encryption plaintext (the remaining 104 bits).

The header is passed around the encryption module unchanged using a delay buffer with 8-bit width and equal in length to the number of clock cycles required to complete an encryption. The output of the header delay buffer is passed directly into the modulator component.

Internal to the FFX encryption component, there are layers of similar delay buffers. Each round of the Feistel structure (recall Figure 3 from page 10) consists of a one way F_k function. The F_k function is detailed in Algorithm 1. There are two AES [4] rounds (yielding X and then Y), according to the F_k algorithm. The AES core utilized was designed by Pranav Patel of the Air Force Institute of Technology and was verified during previous research [16]. Because of the pipelined structure, a delay buffer is required to keep Q available until the first encryption completes, generating P‘. Additionally, this structure also requires 24 AES cores (2 per round) across the given the 12-round implementation. Consequently, with all the other FPGA components involved in prototyping an SADS-B system, this structure requires more resources than the Kintex-7 has available (the initial design overmapped the FPGA by 7%. The following optimization is thus implemented to

reduce the number of required cores to 13 from 24. The first AES operation in each round is identical, effectively reproducing P‘ every round. Therefore, the first round will produce P‘ and provide it as an output to each subsequent round, as shown in Figure 19. Note that the d block is a delay buffer for Q.

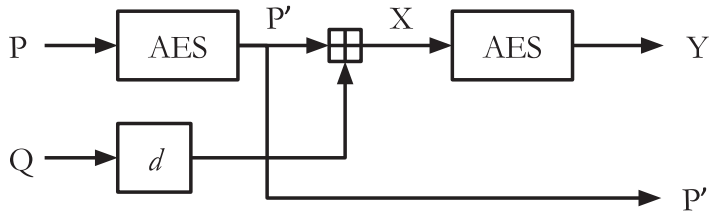


Figure 19. First round of FFX Encryption in the unrolled design.

Each subsequent round is then altered to take in an extra P‘ input, allowing 11 out of the 12 instances of F_k to have only one AES core, as shown in Figure 20. This has the added benefit of eliminating the need for delay buffers inside F_k in other rounds, and also reduces the length of other delay buffers across the entire design.

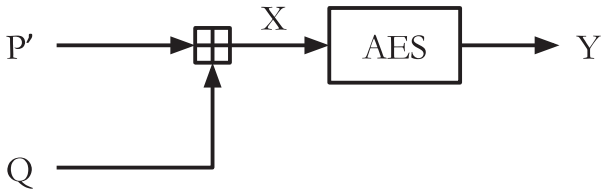


Figure 20. Other rounds of FFX Encryption in the unrolled design.

Given the unrolled and fully pipelined structure of the SADS-B modules, delay buffers are used in each round of FFX according to the block labeled d in Figure 21. The figure highlights the operation of two rounds of encryption in the design, out of twelve.

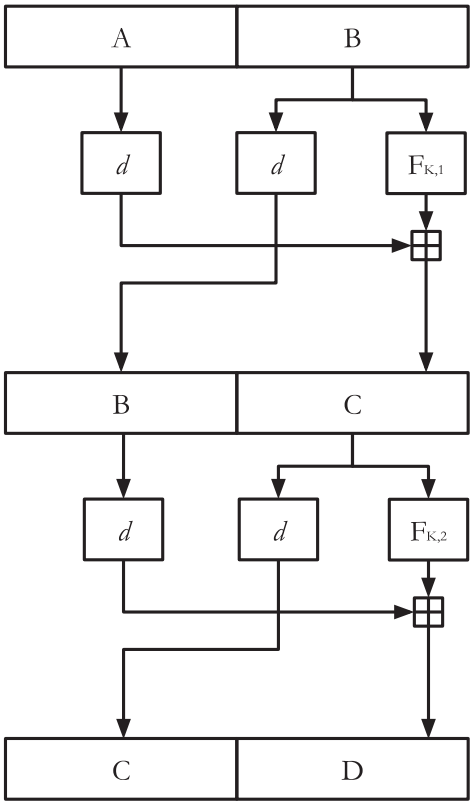


Figure 21. Two example rounds of FFX Encryption.

3.9 FFX Decryption

One benefit of utilizing a Feistel structure in an encryption scheme is the common design for encryption and decryption. In the case of FFX as implemented for ADS-B, there is only one small adjustment that needs to be made. The design is identical to that described in Section 3.8, except that $n = (12 - i)$ instead of $n = i$ for the F_k function in a given encryption round, inputs $C||B$ instead of $A||B$, and outputs $B||A$. All as shown in the decryption diagram in Figure 22.

The fully pipelined aspect of the overall design becomes especially important for decryption. In the SADS-B Decryption Module, the packetizer cannot perform CRC validation, as the bit stream is encrypted, including the parity field containing

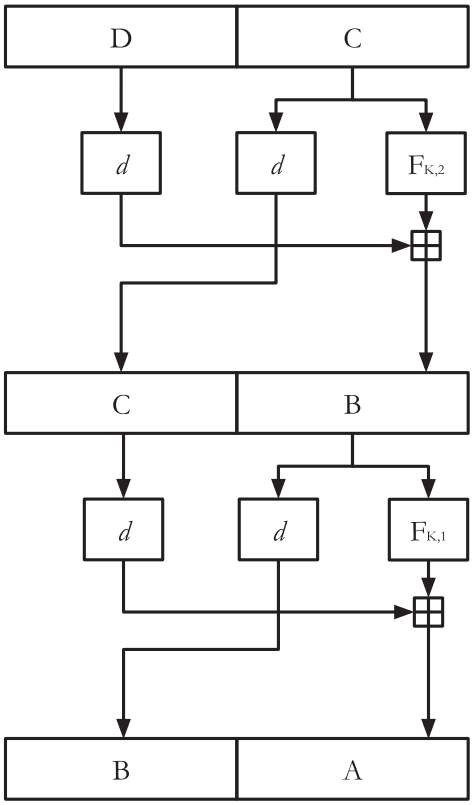


Figure 22. Two example rounds of FFX Decryption.

the CRC. Therefore, every valid sample from the PPM demodulator is shifted into the packetizer, which updates its 112-bit output signal and pulses the output valid signal for one clock cycle. Therefore, every sample (bit) from the PPM Demodulator results in a decryption, which could sometimes be from overlapping signals. Only after the decryption is performed on any given 112-bit sequence can the CRC validation step occur, as depicted in Figure 12 on page 20.

3.10 PPM Modulation

3.10.1 Encryption Version.

PPM modulation in the SADS-B Encryption Module is straight forward. Using Figure 12 as a reference, when the `i_rdy` signal is high, the 112-bit (encrypted) `i_data` packet is stored in a 112-bit vector. Then, on each subsequent clock enable (10 MS/s), the bits are shifted out in order (DF bits first), converted to I and Q data in `o_I` and `o_Q`, respectively, while the `o_valid` signal is set high for one clock enable period. Because the PPM signal is phase agnostic, each pulse I and Q are set to 25% of the dynamic range, $2^{16}/4 = 0x(4000)$ and a negative space is set to $0x(0000)$.

The modulation begins with a proper preamble, and `o_valid` set high. Given the 10 MS/s sample, there are 80 I and Q samples generated before any actual bits are modulated from the 112-bit vector. There are 4 sets of 5-sample pulses that are modulated, according to the offsets shown in Figure 14 on page 22. After the preamble has been modulated, each data bit (starting with the DF-17 header) is then converted to a 10-sample PPM frame using a counter with simple logic. When the last bit has been modulated, `o_valid` returns low.

3.10.2 Decryption Version.

In the SADS-B Decryption Module, the additional step of CRC validation occurs before the modulation into I and Q occurs. Again, using an instance of the serial UCRC component, the CRC value is checked as the bits are shifted out of the 112-bit input vector into the UCRC component [5]. Simultaneously, they are also shifted into a second 112-bit vector, which acts as an output buffer. When the last sample is shifted into the UCRC checker, the `o_match` signal from UCRC should go high, indicating a CRC match. If this occurs (and the packet has been validated),

the output buffer will empty as it is converted into to the 10 MS/s modulation signal described above in Section 3.10.1.

3.11 ADS-B Selector

The ADS-B Selector component in both the SADS-B Encryption Module and Decryption Module is not an instantiated component like the other functional blocks shown in Figures 11 and 12 from page 20. It consists of a simple buffer for the input I, Q, and strobe signals, equal in length (cycle count) to the rest of the design. This allows the input signal to be seamlessly passed through the SADS-B module for all cases where `i_valid` no signal modification is occurring.

IV. Results and Analysis

This chapter details the process used to verify and characterize the SADS-B prototype, according to accuracy, latency, and FPGA resource utilization. It then presents and evaluates the results.

4.1 Verification Methodology

From the research goals outlined in Section 1.3, there are two aspects of verification that were accomplished: component-level and system-level.

4.1.1 Component Verification.

Each major component is verified in several steps along the way to verifying the system as a whole.

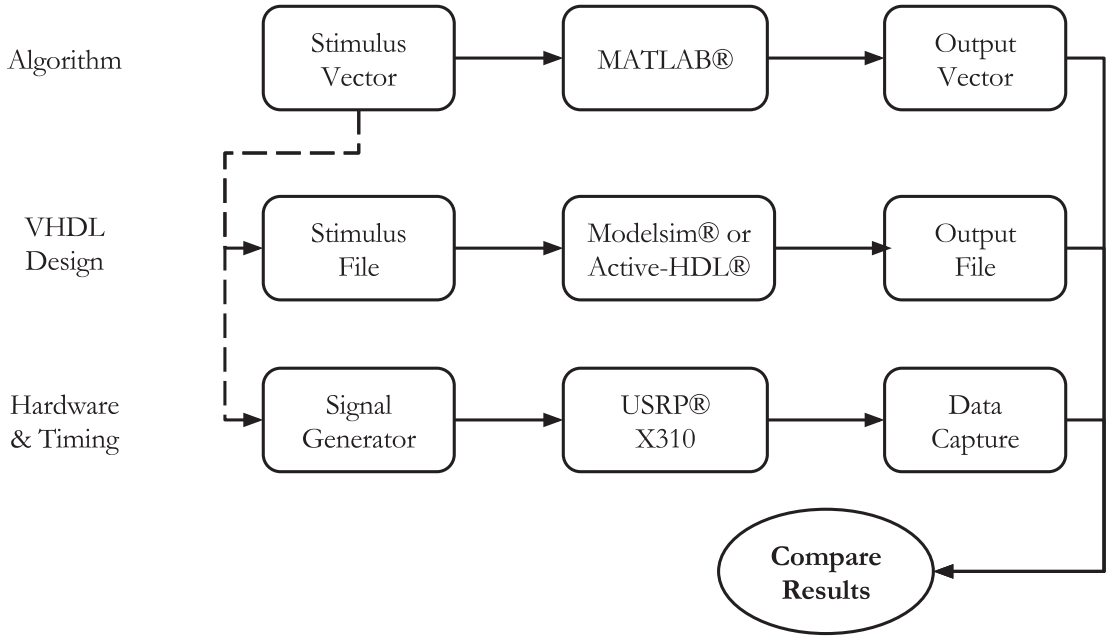


Figure 23. Component verification process.

The diagram in Figure 23 depicts the overall process by which the components were verified. At the algorithm level, an ideal stimulus vector is generated (e.g. an ideal ADS-B message or PPM signal) in MATLAB®. The stimulus vector is used as a test input for the component under test (CUT). The output from the MATLAB® instantiation of that component (e.g. a preamble detector) is another vector (e.g. the ADS-B data stream). The output vector is then be plotted or otherwise used to verify the correctness of the CUT design at the algorithm level.

Second, the stimulus vector is used to generate the stimulus file for the design verification. The stimulus file is a text file that contains a single numerical data sample on each line. Additionally, a VHDL test bench file is also needed to perform three functions:(1) provide simulated input to the hardware, (2) instantiate the CUT, and (3) record the output of the simulation. The input and output occur by means of simulation-only VHDL components. These components, the data source and data sink, read input from a specified file or write output to a specified file, respectively. Active-HDL® or ModelSim® use the test bench design to simulate the operation of the CUT. An example of a stimulus file would be a set of [I,Q] data inputs to the preamble detector or PPM demodulator. The data source reads one line from the stimulus file (typically 16-bit integer format), and supplies the data as input to the CUT, once per clock cycle. The data sink converts the output of the CUT to a numerical value, and writes it to a file, once per simulated clock cycle.

Third, the stimulus vector is used to drive the signal generator for hardware and timing verification. For testing purposes, an additional Xilinx ISE ChipScope® Pro HDL component is added into the design to monitor signals of interest to monitor the operation of the CUT. As described in Section 2.6, ChipScope® Pro provides a mechanism to capture real-time data signals to an attached PC for static

analysis. Alternatively, the signals may be verified using data capture tools available via the USRP™ top level code, using the GNU Radio command line tools. After compilation of the design in Xilinx ISE, the design is programmed onto the FPGA on the X310. Using MATLAB®, the stimulus vector is uploaded to the Keysight N5182B signal generator using the Advanced Configuration and Power Interface (ACPI) interface over the network. Depending on the method of capture, either the Xilinx ISE Analyzer tool is used to control and retrieve data from the ChipScope® Pro component, or the files captured via GNU Radio are loaded in MATLAB® for static analysis.

Lastly, the outputs verification phases are compared to each other and/or ground truth information to verify functional correctness of the CUT.

4.1.2 System Verification.

Encryption.

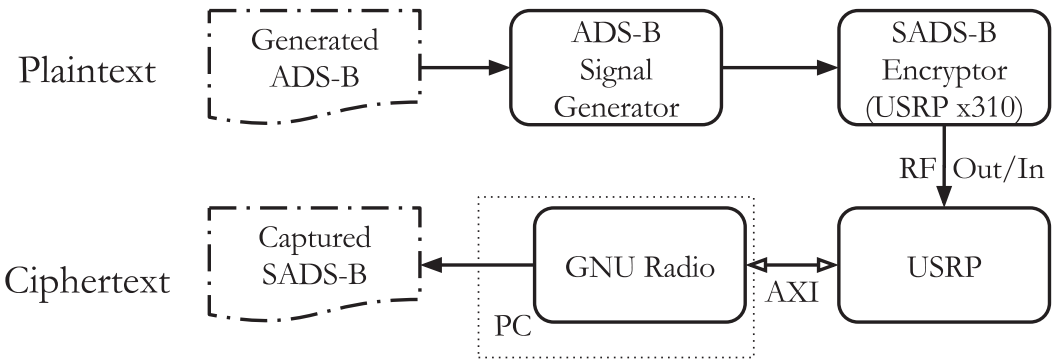


Figure 24. End-to-end verification in encryption mode.

System verification for SADS-B in encryption mode is accomplished using the previously captured ADS-B signals from the Aeroflex IFR 6000. The signals have approximately 80 μs of quiet surrounding the signal. Figure 25 shows a plot of the

signal power for one of the Aeroflex ADS-B signals.

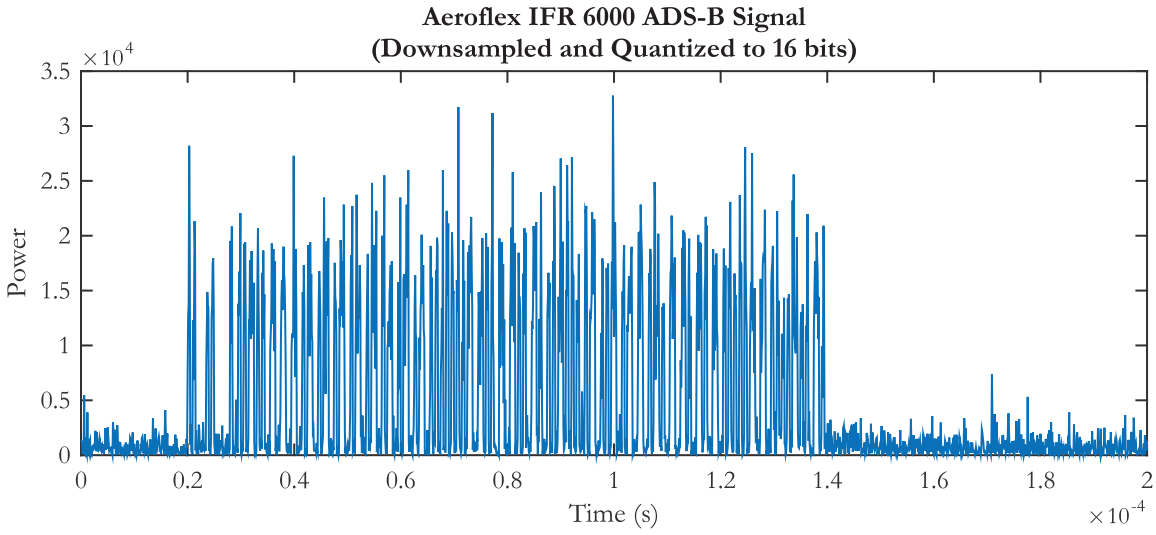


Figure 25. Example Aeroflex IFR 6000 signal.

Using an ACPI interface over a TCP/IP connection, MATLAB® is used to upload one of the signals to the Keysight N5182B signal generator. The signal generator is then configured to continuously repeat the signal, which simplifies data collection. With the repeating signal approximately 200 μs in length, any data capture of at least 400 μs will contain at least one complete ADS-B message.

In correct operation, the SADS-B prototype would receive the ADS-B Out signal from the signal generator and transmit the SADS-B Out signal. The next device in the chain is a second USRP™ acting as a SDR configured (using GNU Radio) according to the diagram shown in Figure 5. It is set as a simple receiver, sampling as 10 MS/s, centered on 1090 MHz. The data is captured on the controlling PC. The function of this USRP™ is to capture the SADS-B Out signal for verification through static analysis in MATLAB®. The received data is in an interleaved [I,Q] format, each of which are 32 bit precision floating point numbers.

The next step in MATLAB® is to setup the data for analysis. This consists of de-interleaving the [I,Q] data, plotting, manual inspecting the plot for the signal,

and cropping the signal down. The signal can then be analyzed by using the PPM Demodulator MATLAB® model to retrieve the bits.

The recovered bits are compared with known ciphertext (determined previously when verifying the FFX Encryptor component in Section 4.2.4.

Decryption.

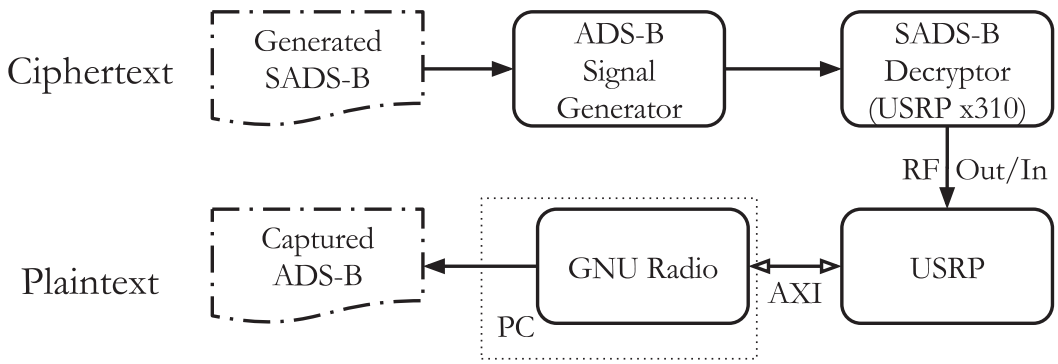


Figure 26. End-to-end verification in decryption mode.

A similar process is followed for verification of the SADS-B prototype in decryption mode, with a few modifications as depicted in Figure 26.

Specifically, the signal just captured during the Encryption test is used to drive the N5182B vector signal generator. In the final step, instead of comparing with the known ciphertext, a comparison is made with known plaintext.

4.2 Component Verification Results

Several captured files from an Aeroflex IFR 6000 are used to verify the preamble detector. The captures are baseband I and Q data samples, saved in comma separated value (CSV) format, at a a sample rate of 50 MS/s. The ADS-B values

used to create the data are known, providing the basis for comparison throughout verification.

4.2.1 Preamble Detection.

Modeling and Simulation.

For the MATLAB® simulation, the captured files are loaded, downsampled to 10 MS/s, and quantized to 16-bit samples using the MATLAB® Communications System Toolbox™ function `fi`. The I and Q samples are combined into received signal power, $P = I^2 + Q^2$.

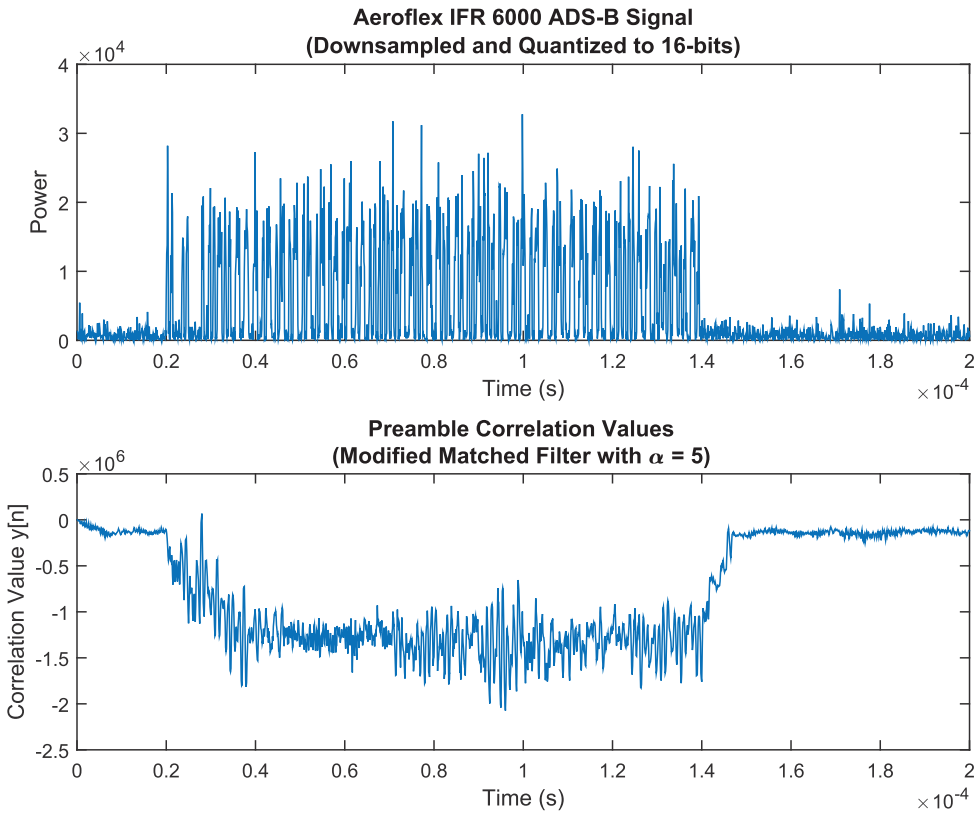


Figure 27. MATLAB® results for preamble detection with $\alpha = 5$.

The first plot in Figure 27 shows the resulting waveform plot of power over

time. The second plot in Figure 27 shows the correlation coefficient $y[n]$ (as discussed in Section 3.5) over time. The first local maximum aligns correctly to the signal start at $t = 3.08 * 10^{-4}$ s, with a correlation value of $y[n] = 5.61 * 10^{-4}$.

Hardware Simulation.

In the case of the preamble detection, there are two stimulus files created by the MATLAB® simulation. The files written contain the downsampled I and Q data (10 MS/s), consisting of one sample per line in integer representation of the signed 16-bit samples.

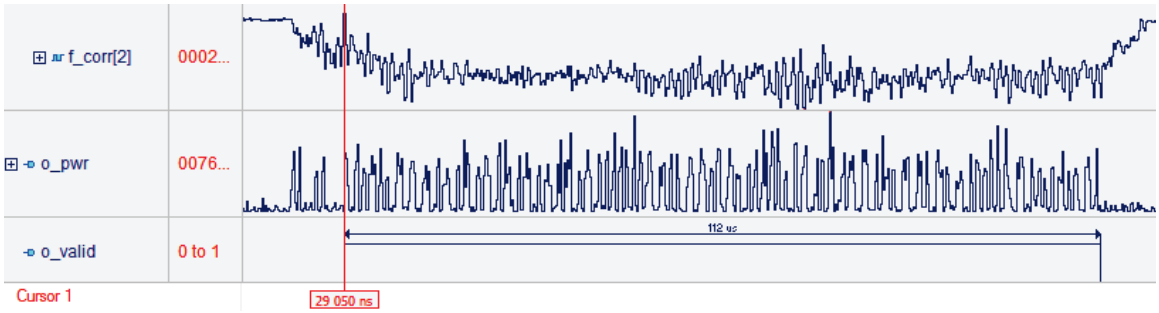


Figure 28. Hardware simulation results for preamble detection with $\alpha = 5$.

Figure 28 shows the results of the simulation, via a screenshot of the simulation interface in the Active HDL program. It shows the output valid signal `o_valid` is correctly set high and low corresponding to the presence of the ADS-B data signal, for a period of exactly 112 μs. Additionally, it shows the timestamp of the rising edge of `o_valid` as 29,050 ns, which gives context to Figure 29.

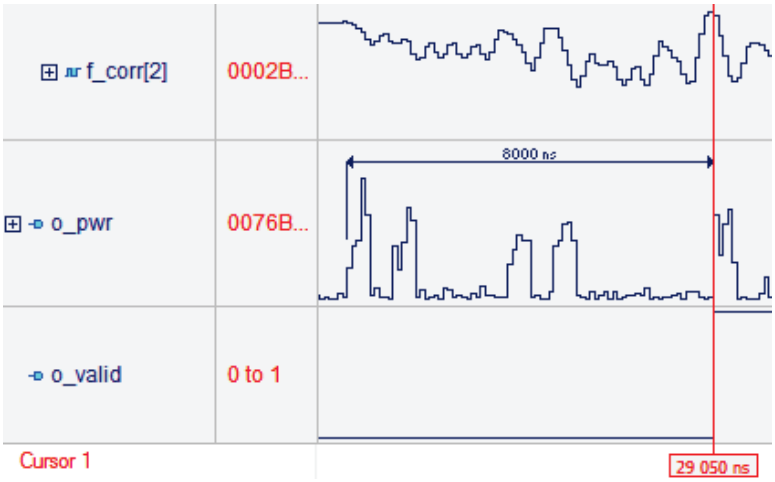


Figure 29. Hardware simulation results zoomed in on preamble.

In Figure 29, the 8 μs (8000 ns) preamble (shown in `o_pwr`) is exactly preceding the sample that is classified as n_{start}, the beginning of the ADS-B data signal. The `o_valid` also is set high in alignment with the first sample of the data signal and the peak correlation coefficient $y[n]$ (shown as `f_corr[2]`). The output of the synchronizer component during hardware simulation in Active-HDL® is a 1120 sample text file, where the first sample in the file is the first sample of the first data bit of the ADS-B signal.

4.2.2 PPM Demodulation.

The demodulator only reads inputs when `o_valid` from the synchronizer is set high. Recalling the top plot from Figure 27, the data input read by the demodulator is essentially a shifted and clipped version, shown in Figure 30.

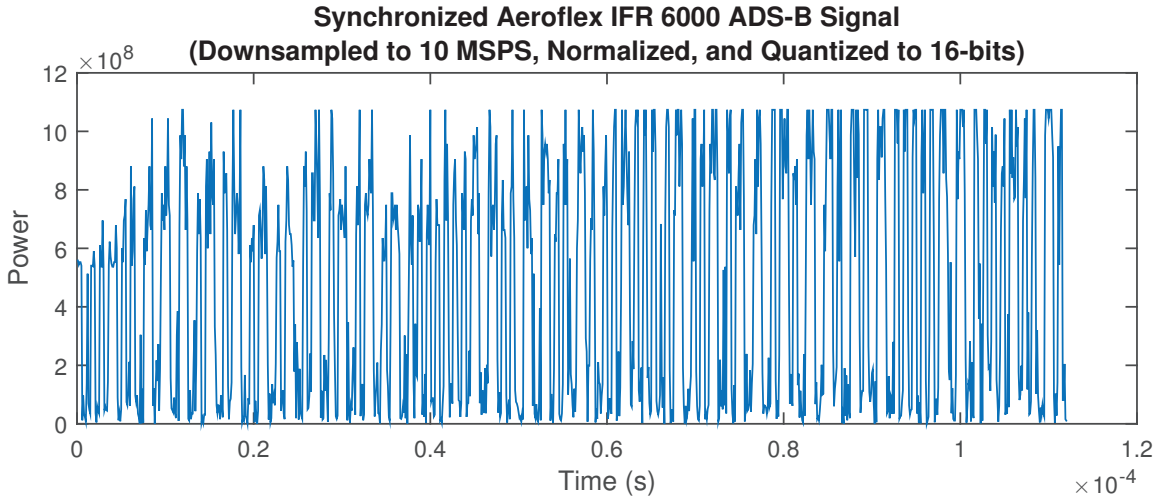


Figure 30. Plot of synchronizer output as received into the demodulator.

Modeling and Simulation.

The demodulation output should be the 112 bits that this PPM signal represents. For verification in MATLAB®, the stimulus vector is the 1120 samples

as plotted in Figure 30 and the output is a 112 bit vector. In this case, the simplest mechanism to verify the bits are correct, aside from manual inspection, is to parse the data values into the respective ADS-B fields and compare with the ground truth data. However, the PI field containing the 24-bit CRC was not explicitly provided as input to the Aeroflex IFR 6000, but the ground truth for it is calculated from the first 88 bits of the message. The Communications System Toolbox™ in MATLAB® provides CRC utilities, which are utilized for this purpose. The `plot_sim_sync_output.m` MATLAB® script is used to perform the parsing and CRC calculation, then compare the results.

```
>> plot_sim_sync_output;
df = 17, ca = 6, aa = A92492, me = 4805D763C894D6
message being tested:            10001110101010010010010010010010010010000000010111010111
                                 01100011110010001001010011010110
message with checksum computed:  10001110101010010010010010010010010010000000010111010111
                                 01100011110010001001010011010110110100000010100000111011
message as received/interpreted: 10001110101010010010010010010010010010000000010111010111
                                 01100011110010001001010011010110110100000010100000111011
bits should be zero if no error: 0
```

Figure 31. Verification output of PPM demodulation in MATLAB®.

The printout from the function is shown in Figure 31. As shown, the computed CRC values match the demodulated bits. Additionally, the message data matches known values.

Hardware Simulation.

The hardware simulation for demodulation verification is performed in ModelSim®, using the 1120-sample stimulus file generated as output during synchronizer verification. The output and enable signals are verified for correct timing. Additionally, the final output is verified through static analysis using a MATLAB® script.

Figure 32 shows the results of the ModelSim® simulation. From the figure, the period of the 1 μs (1000000 ps) is shown between samples. Also, the output

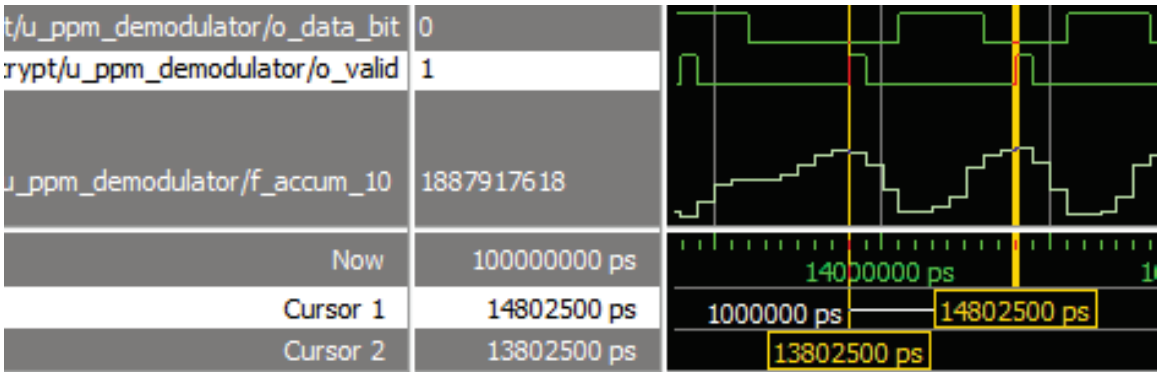


Figure 32. ModelSim® 10.4 screenshot of demodulator verification.

samples are aligned with the peaks and valleys in the signal, indicating a correctly implemented pipeline structure.

The output file `captured_demod_signal.txt` is analyzed using a similar script from modeling and simulation verification. The `plot_sim_demod_output.m` performs essentially the same steps as the `plot_sim_sync_output.m` previously used, only instead of reading the input from a MATLAB® vector, the input is read in from `captured_demod_signal.txt` file as output from the demodulator. The output confirms the number of bits in addition to the correctness of the bits, all shown in Figure 33.

```
>> plot_sim_demod_output
file captured_sim_demod_signal_1.txt is length 112
df = 17, ca = 6, aa = A92492, me = 4805D763C894D6
message being tested:            10001110101010010010010010010010010010000000010111010111
                                 01100011110010001001010011010110
message with checksum computed:  10001110101010010010010010010010010010000000010111010111
                                 01100011110010001001010011010110111010000001010000011101 1
message as received/interpreted: 10001110101010010010010010010010010010000000010111010111
                                 01100011110010001001010011010110111010000001010000011101 1
bits should be zero if no error: 0
```

Figure 33. Verification of demodulator hardware design output using MATLAB®.

4.2.3 Packetization & CRC Validation.

Verification for the packetizer component is focused on hardware implementation, as there is a readily available CRC verification mechanism using the MATLAB® Communications System Toolbox™ and the correct implementation of packetization is essentially a hardware timing problem. Therefore, rather than generate an explicit stimulus vector to verify the packetizer component, the test bench for hardware verification consists of reusing the previous test bench for PPM demodulation, with the outputs wired as the inputs for the packetizer.

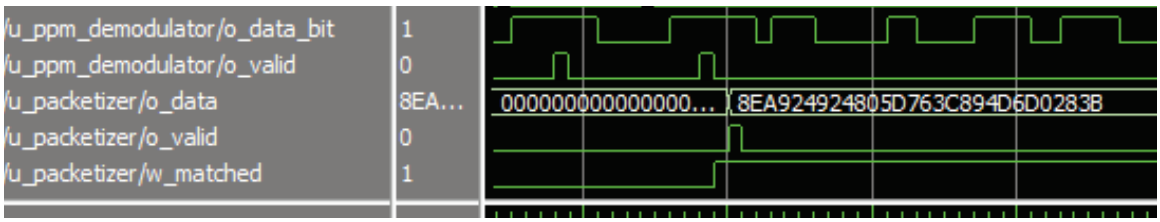


Figure 34. ModelSim® 10.4 screenshot of packetizer verification.

Figure 34 shows the ModelSim® results for the described test bench. As the last sample is output according to the demodulator's `o_valid` signal, the CRC matches (shown by `w_matched_crc`) and the output is correctly loaded into the output register, `o_data`. The message in the test case is known, including the CRC bits, which match the output shown.

4.2.4 FFX Encryption and Decryption.

There are no published test vectors for for verifying FFX encryption as there are for other standards such as AES. However, many plaintext-ciphertext pairs were generated by Finke in the evaluation of FFX for use in securing ADS-B during previous research [11]. These plaintext-ciphertext pairs will serve as the basis for verification for the implementation of FFX. As such, the verification for the encryption and decryption components will be performend in hardware simulation.

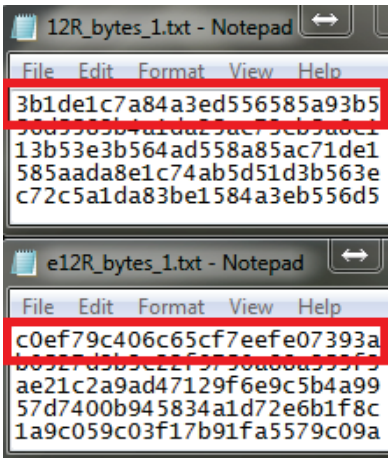


Figure 35. Screenshot of Finke test plaintext and ciphertext files [11].

Figure 35 shows an example plaintext/ciphertext pair that is used to verify the SADS-B encryption and decryption components, where the `12R_bytes_1.txt` file contains the plaintext messages and `e12R_bytes_1.txt` contains the corresponding ciphertext. As the ADS-B DF and CA fields are not encrypted, they are not shown in the files. The encryption key for the test files is known and set statically prior to simulation. The key is shown in Figure 36.

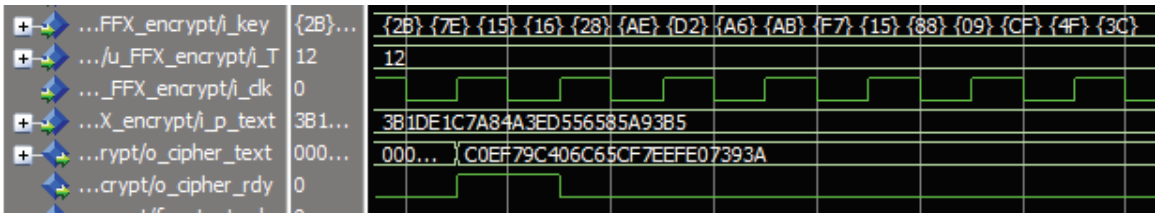


Figure 36. Screenshot of encryption component in ModelSim®.

For encryption verification, Figure 36 shows the plaintext message input and the corresponding ciphertext output, aligned to the clock cycle when the output valid `o_cipher_text_rdy` is set high. The input and output match the known plaintext/ciphertext pair.

Similarly, for decryption, Figure 37 shows the ciphertext message input and the corresponding plaintext output, aligned to the clock cycle when the output valid

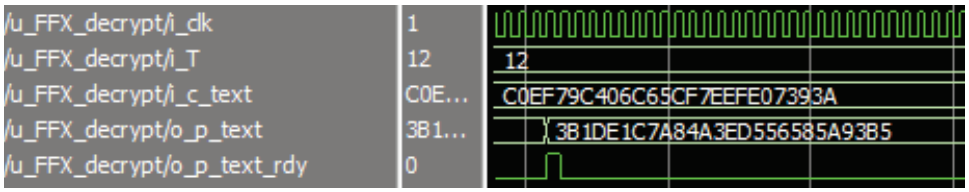


Figure 37. Screenshot of decryption component in ModelSim®.

`o_p_text_rdy` is set high.

4.2.5 PPM Modulation.

Figure 38. Screenshot of full modulation signal in Active-HDL®.

As with PPM demodulation, the verification for PPM modulation is focused on timing implementation. Figure 38 shows the full signal timing of the output, which is aligned to the output valid `o_valid` signal. Additionally, the period of the signal is correct at 120 μs (including the preamble). Figure 39 shows the same simulation, zoomed in on the preamble to verify its appropriate timing prior to bit modulation.

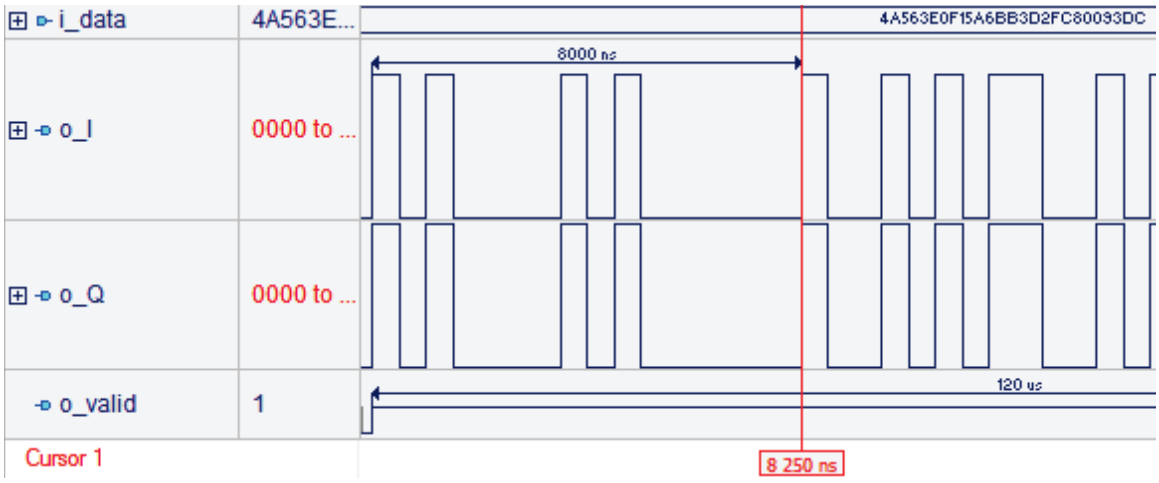


Figure 39. Screenshot of modulation signal in Active-HDL®, zoomed on preamble timing.

4.3 Performance Test Methodology

4.3.1 Evaluate End-to-End Latency.

Latency is a significant factor in achieving an operationally useful prototype for SADS-B. Evaluation of the end-to-end latency consists of measuring the operational latency of the SADS-B prototype in each mode (decrypt and encrypt) following the successful verification of the system. Because the design is fully pipelined, the longest propagation path (in units of clock cycles) can be determined through static VHDL code analysis. Alternatively, the FPGA component latency can be measured using a simulation environment as well. However, this does not account for the overhead latency of the signal propagation within the X310 before and after the designed SADS-B module.

Therefore, an additional and more robust measurement of operational latency requires the use of a dual input oscilloscope. In this case, a Teledyne LeCroy WavePro 760Zi-A is used, according to the setup shown in Figure 40. To perform the measurement, the signal generator is setup using MATLAB® to generate a valid ADS-B waveform (previously captured from an Aeroflex IFR 6000), as described in

Section 4.1.2. However, the primary difference in this case is the N5182B vector signal generator will not be setup in a repeating mode. Instead, it will perform a single playback of the ADS-B waveform upon a trigger button being manually pressed on the signal generator. The trigger could alternatively be set through an ACPI commands via MATLAB®. Before playback, the oscilloscope is setup to trigger appropriate on the input ADS-B preamble reaching a specified power level. Once triggered, the oscilloscope captures both channels at 40 GSPS. The data is then manually saved using the interface on oscilloscope to an attached USB drive. The results are evaluated using MATLAB® plots and manually inspection.

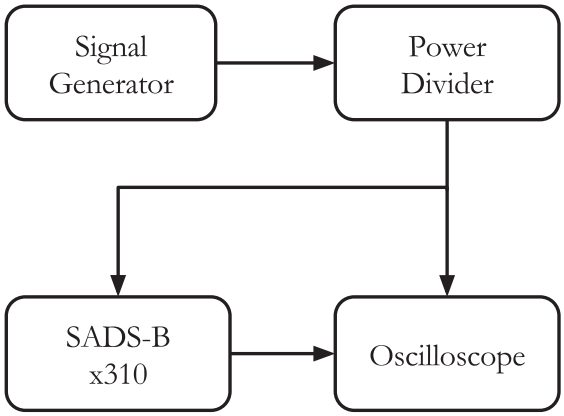


Figure 40. Test setup for end-to-end latency measurement.

The data collection is saved to a local file on the oscilloscope, off-loaded to a PC and plotted using MATLAB®. Because the captures are synchronized, the latency in seconds is simply a $\Delta t = \Delta s \times F_s$, where Δs is the number of samples between the start of each preamble and F_s is the sample rate of the oscilloscope.

The process is repeated for each mode (encryption and decryption). Finally, the results are evaluated against the threshold latency of 50ms based on the ADS-B specification [21].

4.3.2 Evaluate Performance of SADS-B Components.

Demodulation Error.

Demodulation is evaluated against the lower limit bit error rate (BER) for a 2-PPM signal, which is the same as on-off keying (OOK) and shown in Equation 8.

$$BER \geq \frac{1}{2} * e^{-E_b/N_0} \tag{8}$$

A Monte Carlo simulation is performed by using the `randn` in MATLAB® to generate and add noise to a random stream of bits, according to the various levels of E_b/N_0, the SNR. The BER for the implemented algorithm is evaluated against the ideal formula curve on a plot of SNR vs. BER.

Preamble Detection Error.

A Monte Carlo simulation is used to evaluate detection performance, using two simulations. The first simulation evaluates how well each value of α performs when a preamble signal is present. It begins with an ideal preamble signal and uses the `randn` function within MATLAB® to add additive white Gaussian noise (AWGN) to the signal to a specified SNR. Next, the resulting signal is input to the preamble detector algorithm. The number of correct detections are recorded. This is repeated for a variety of settings for α. The false negative rate (FNR) is then used to evaluate the performance across different settings of α.

The second simulation is similar, but begins with a half-matching false preamble. The false preamble is a signal with four pulses that align to the preamble pulses. It also has four pulses that do not align. This is similar to many "8-bit windows" PPM. Visually, the two general cases where this is possible are shown in Figure 41.

In a normal ADS-B signal, this bit pattern is likely to occur in approximately

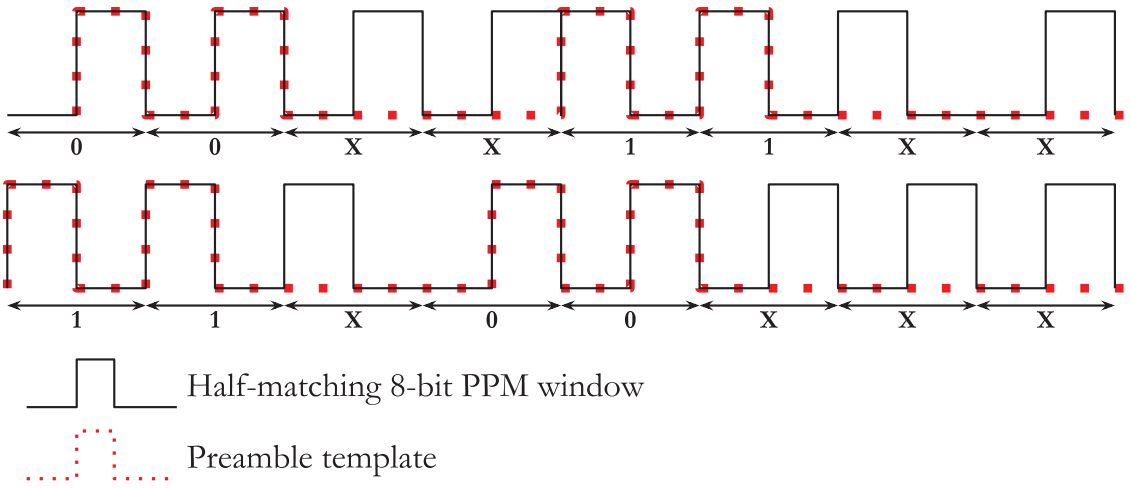


Figure 41. Preamble template aligning with 8 bits from a PPM signal.

12.5% of the time. It will either occur with [1,1,X,0,0,X,X,X] pattern (as discussed later in Section 3.5), or with a [0,0,X,X,1,1,X,X]. An 'X' in these patterns represents either a '1' or a '0', and are where a pulse is placed for this simulation. Each of these events have approximately a $\frac{1}{2^4} = 6.25\%$ chance to occur, combining to approximately a 12.5% of either of them occurring in any given 8-bit window in the ADS-B PPM signal.

If the demodulation results correlate with ideal BER formula given by Equation 8, then Figure 42 will approximate the minimum E_b/N_0 required to receive a signal error-free 80% of the time.

Let SNR_{thr} be the threshold value of E_b/N_0 determined in Figure 42. The preamble detection results will be used to determine the optimal value of penalty factor α based on the following criteria:

(1) successful detection of real preambles (TPR) $\geq 90\%$, $\forall E_b/N_0 \geq SNR_{thr}$

(2) successful rejection of normal ADS-B signals as preambles (TNR) $\geq$ 90%, $\forall E_b/N_0 \geq SNR_{thr}$

$$
\begin{aligned}
(1 - BER)^{112} &= 80\% \\
BER &= 1 - \sqrt[112]{.8} \\
BER &= 1.99 * 10^{-3} \\
BER &= \frac{1}{2} * e^{-E_b/N_0} \\
\frac{1}{2} * e^{-E_b/N_0} &= 1.99 * 10^{-3} \\
E_b/N_0 &= 5.526 \\
E_b/N_0(\text{dB}) &= 10 * \log_{10} 5.526 = 7.424\text{dB}
\end{aligned}
$$

Figure 42. E_b/N_0 that will be allow error-free reception 80% of the time.

With these criteria in mind, the maximum allowable false positive rate (FPR) (1-true negative rate (TNR)) that will satisfy condition (2) is calculated in Figure 43. Therefore a correctly implemented preamble detector will have a FPR $\leq 7.95 * 10^{-3}$, $\forall E_b/N_0 \geq SNR_{thr}$.

Let $P(FP)$ be the probability of false preamble detection across a 112-bit ADS-B signal. Let $P(HM)$ be the probability of a false preamble detection during an 8-bit ADS-B signal that matches one of the templates described in Section 4.3.2.

$$
\begin{aligned}
(1 - P(FP))^{(112-6)} &= 90\% \\
P(FP) &= 1 - \sqrt[106]{.9} \\
P(FP) &= 9.935 * 10^{-4} \\
P(FP) &= P(HM) \times FPR \\
P(HM) &= 2 \times \frac{1}{2^4} = .125 \\
FPR &= \frac{9.935 * 10^{-4}}{.125} = 7.95 * 10^{-3}
\end{aligned}
$$

Figure 43. FPR that will allow successful rejection 90% of the time.

4.4 Performance Results

4.4.1 Operational Latency.

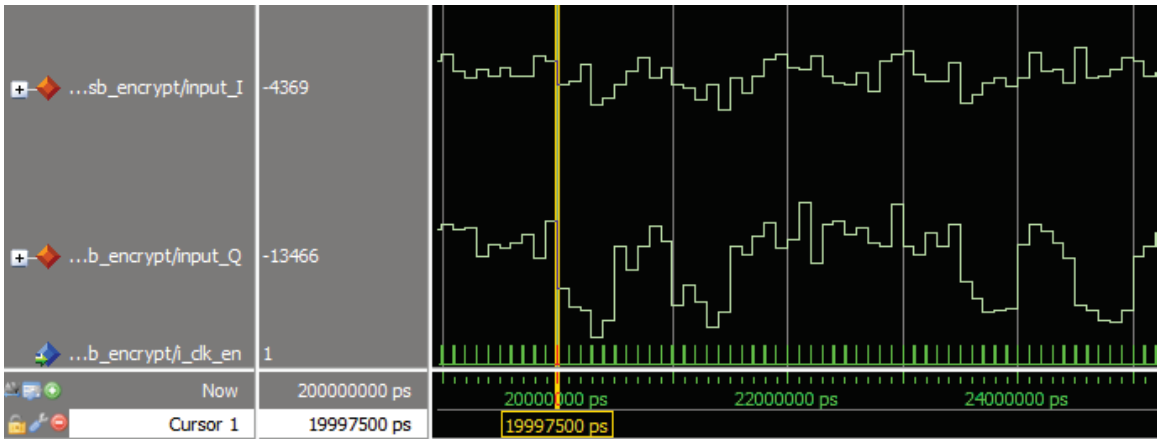


Figure 44. Timing of input sample for SADS-B encryption module.

Figure 44 shows arrival timing of the first preamble sample as input into the SADS-B encryption module from the DDC on the FPGA. Though not visible, note that this time is aligned with the rising edge of the clock `i_clk` when the clock enable signal `i_clk_en` is high.

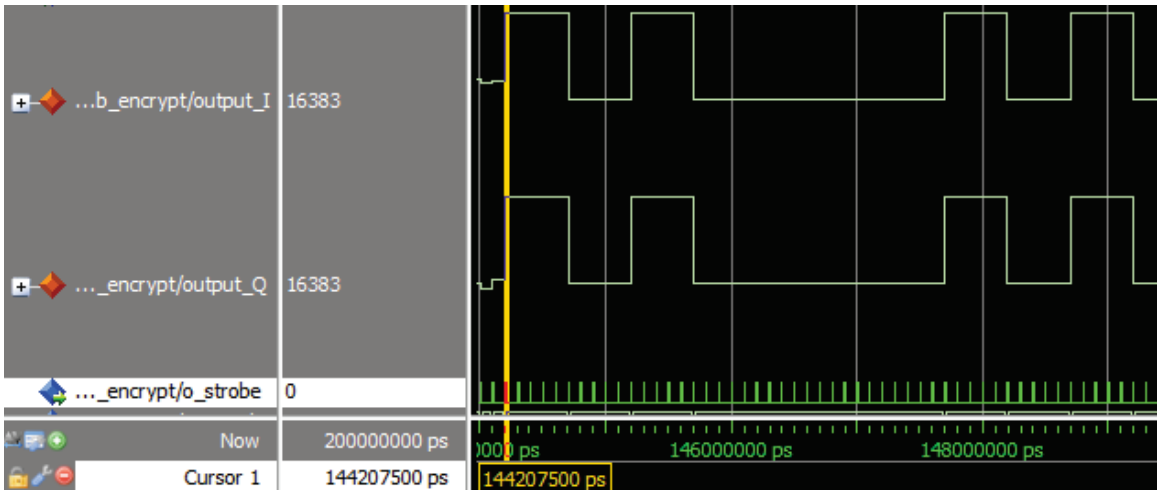


Figure 45. Timing of input sample for SADS-B encryption module.

Figure 45 shows the output timing of the corresponding output preamble sam-

ple from the SADS-B encryption module to the DUC on the FPGA. The time is aligned with the rising edge of `i_clk` when the output enable signal `o_strobe` is high. The resulting latency of the SADS-B module is 144207500 - 19997500 ps = 124.21 μs of latency.

4.4.2 Demodulation Performance.

Figure 46 shows the results from the Monte Carlo simulation of the PPM Demodulator component of the SADS-B Modules. The same component was used for encryption and decryption, so the simulation was only performed once. 10,000,000 randomly generated bits (using the `randi` MATLAB® function) were converted to their ideal PPM signal, with a pulse value of $\sqrt{2} + \sqrt{2}i$, across the number of samples. AWGN was added to each sample, according to a specified E_b/N_0, by using the `randn` MATLAB® function and scaling it. The detection algorithm was then used to determine the bit value of the PPM signal according to the algorithm discussed in Section 3.6. The resulting bit was compared to the known vector and the number of errors was recorded for that particular value of E_b/N_0. The results show that the algorithm performs closely to the ideal BER rate for a 2-PPM signal.

4.4.3 Preamble Detection Characteristics and Performance.

With the demodulation results closely correlating to ideal 2-PPM BER, the threshold $SNR_{thr} = 7.424dB$ from Section 4.3.2 can be used as an approximation for the threshold SNR for the criteria for successful preamble detection. Recalling these criteria, they are:

(1) successful detection of real preambles (TPR) $\geq 90\%$, $\forall E_b/N_0 \geq SNR_{thr}$

(2) successful rejection of normal ADS-B signals as preambles (TNR) $\geq$ 90%, $\forall E_b/N_0 \geq SNR_{thr}$

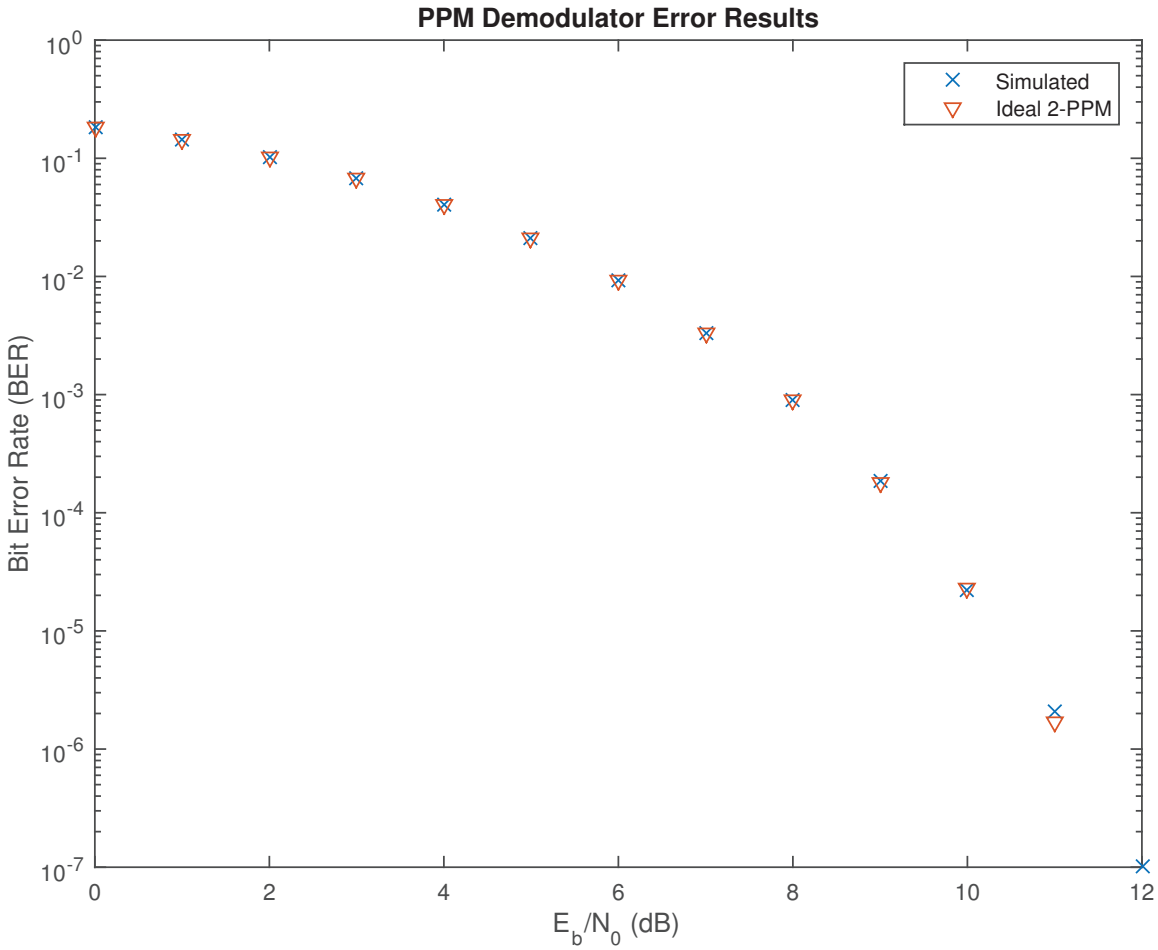


Figure 46. BER results of Monte Carlo simulation.

The first simulation was conducted according to Section 4.3.2 and used 1,000,000 randomly generated samples per trial run. Each sample consists of an ideal PPM signal with AWGN added using the MATLAB® `randn` function, scaled to achieve the desired value of E_b/N_0. The results in Figure 47 show the rate at which valid preambles are succesfully detected (i.e. the true positive rate (TPR)) based on a corresponding value of E_b/N_0. This simulation is repeated across several different settings for the penalty factor, α, as described by the Equation 6.

From criteria (1) above, the cutoff TPR = 90% at $SNR_{thr} = 7.424$ dB is shown by the intersection of the dotted lines in the figure. Curves that pass through the upper left quadrant meet the criteria from (1) for successful preamble detection. As

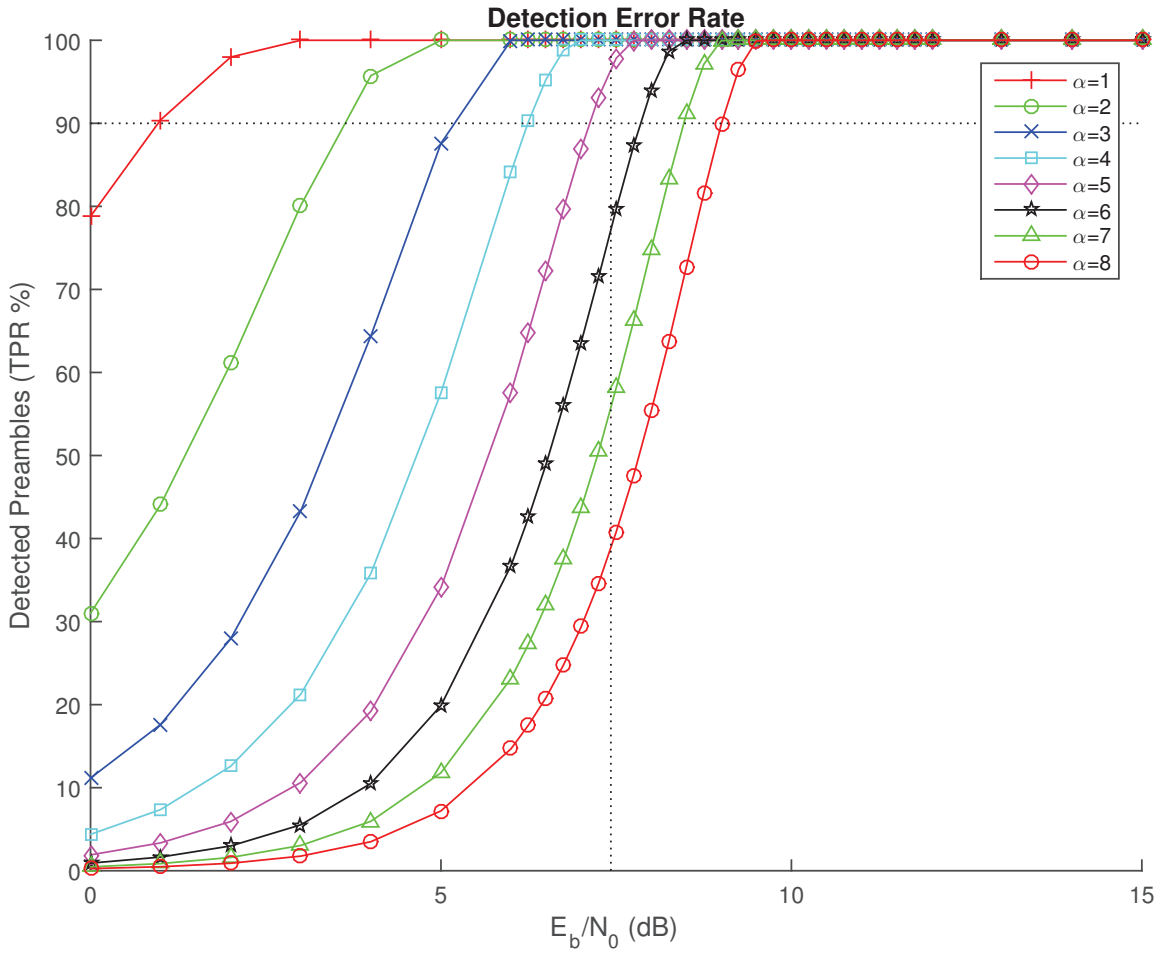


Figure 47. TPR results of Monte Carlo simulation for various α.

shown in Figure 47, only values where $1 \leq \alpha \leq 5$ stay within the criteria.

The second simulation was also conducted according to Section 4.3.2 and used 1,000,000 randomly generated samples per trial run. Each sample uses a randomly generated signal template from the false preambles templates, shown in Figure 41, with noise added using the `randn` MATLAB® function according to a specified level of E_b/N_0. Figure 48 shows the FPR across different values of E_b/N_0, for the penalty factors $1 \leq \alpha \leq 5$ which met criteria (1). The cutoff FPR at SNR_{thr} is shown by the intersection of the dotted lines in the figure. All curves that pass through the lower left quadrant meet criteria (2) for successful preamble detection. As shown in Figure 48, the α values of 3,4, and 5 meet the criteria.

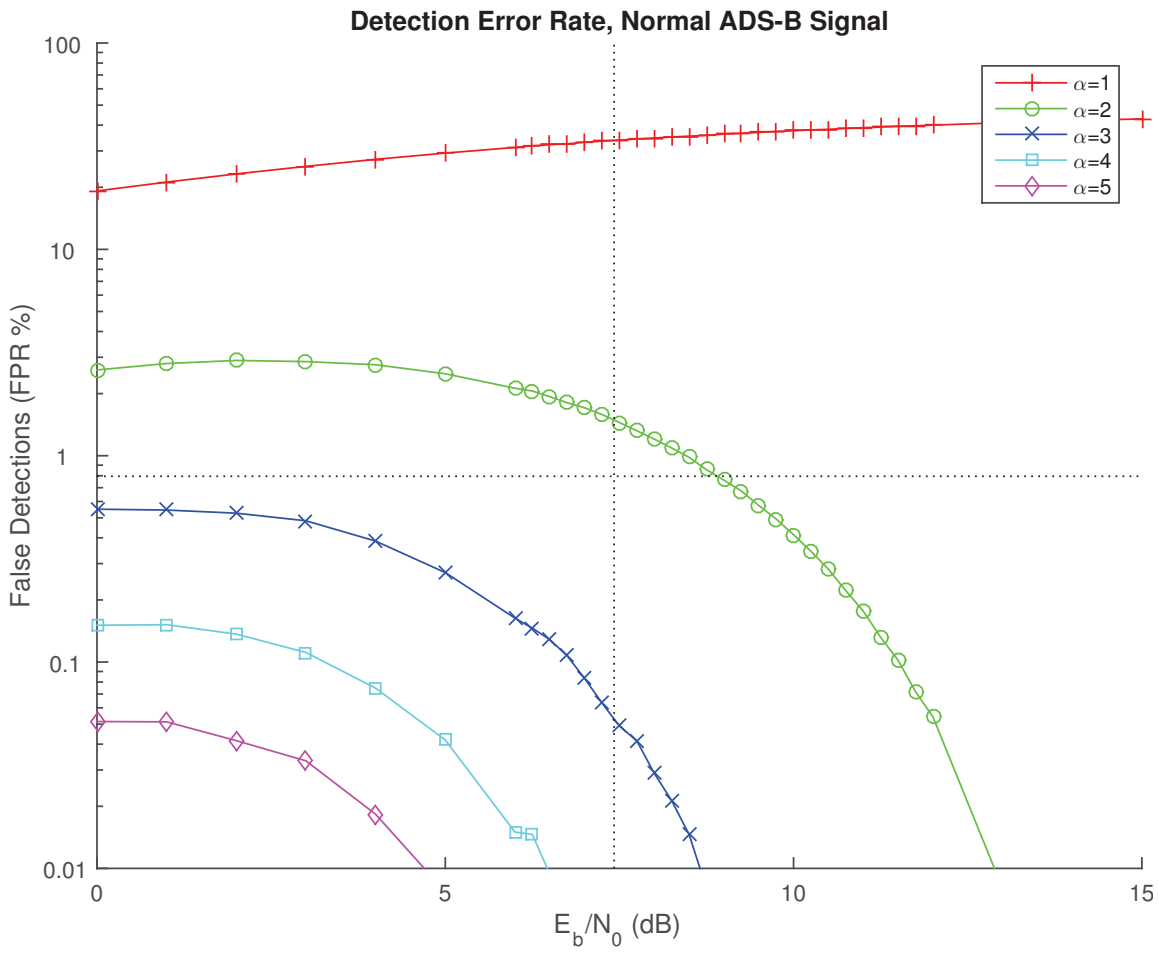


Figure 48. FPR results of Monte Carlo simulation, $1 \leq \alpha \leq 5$.

4.4.4 FPGA Resource Usage.

The Xilinx Kintex-7 is a fairly modern FPGA, compared to many existing FPGA components, especially. The overall FPGA resource consumption expressed by slices, 6-input LUTs, random access memory (RAM) (Kb), shift registers (Kb), and flip-flops. Table 2 shows the total available for each resource type on the Kintex-7 (specifically the Kintex-7K410T) onboard the USRP™ X310.

Based on the system design and implementation, Table 3 shows the required FPGA resource utilization, reflecting the portion of the usage attributable to the SADS-B encryption module designed during the course of this research, and also

Table 2. Total Available Kintex-7K410T Resources

Resource Type	Quantity Available
Slices	63,550
6-input LUTs	254,200
RAM (Kb)	5,663
Flip-Flops	508,400

the total including the overhead from the inherited USRP™ top level design.

Table 3. Kintex-7K410T Resources Used by SADS-B

Resource Type	SADS-B Module Usage	Total Prototype Usage
Slices	34,316	54,219
6-input LUTs	121,465	174,560
RAM (Kb)	299	804
Flip-Flops	96192	136,000

V. Conclusions

5.1 Research Summary

The goal of this research was to prototype and characterize an implementation of BITW security for the ADS-B system. Given the unique structure and small size of ADS-B information messages, and based on previous research, the FFX algorithm was implemented on an FPGA-based platform. The prototype was designed, implemented, and verified using system engineering principles and an incremental approach.

The components were individually verified through the modeling and simulation, hardware design simulation, and hardware implementation using various techniques. The verification criteria were based on the intended operating environment. The performance of the system as a whole system was characterized in terms of latency, throughput, and the strength of the detection and demodulation schemes. The latency and throughput of the system are more than adequate in meeting the requirements for a transparent BITW security system for ADS-B and would be expected to perform transparently in an operating context. The FPGA utilization is sufficient for implementation of a single SADS-B module (encryption or decryption), but not both on a single USRP™ X310 prototype device without further optimization or utilization of a larger FPGA.

Based on sub-millisecond latency and a verified end-to-end design, the results of this research suggest that FPE (specifically FFX), when implemented in a BITW architecture, is a suitable mechanism for layering security onto the existing ADS-B infrastructure.

5.2 Future Work

The practical next step in this area of research from a system perspective is to miniaturize the prototyped SADS-B system for implementation on an operational Unmanned Aerial Vehicle (UAV). To do this, would either be miniaturizing the HDL design for inclusion in an already existing on-board electronics package, or reducing the overall size, weight and power required and analyzing the results.

Several cryptographic enhancements or modifications could be be proposed for use with the current SADS-B prototype. A cryptographic recommendation from Jochum was to preclude the CRC bits in the message from being included, then recalculating them for the new packet/frame prior to transmission [15]. This would allow the receiver to validate the encrypted SADS-B packet before decryption, processing, and transmittal to the ADS-B IN receiving device. The DF-19 military channel has no readily available specifications for the use of the associated CA codes. This field could be used in conjuction with a public key infrastructure (PKI) scheme to securely distributed updated symmetric keys via asymmetric cryptography, similar to Secure Sockets Layer (SSL) in the internet space. Additionally (or alternatively) the addition of an integrity check such as a message authentication code (MAC) could be included in place of the CRC check or as an augmentation of it.

The performance standards that built the basis for the preamble detector have a section on enhanced preamble detection [21]. This material could be used to optimize the SADS-B prototype in the decryption configuration to maximize performance in the context of closer-to-real-world conditions.

5.3 Contributions

This research generated a real-time ADS-B prototype with verification results that supports its operational feasibility and utility. It validated the prototype against performance requirements specified for ADS-B 1090ES, supporting the feasibility of adding the SADS-B layer of security on top of ADS-B in a military context.

It also made strides towards a development framework for BITW encryption, including further enhancements to ADS-B security or BITW implementation in other communications contexts or applications. This framework includes the contribution more than 4,000 lines of MATLAB®, VHDL, and Verilog code.

Appendix A.

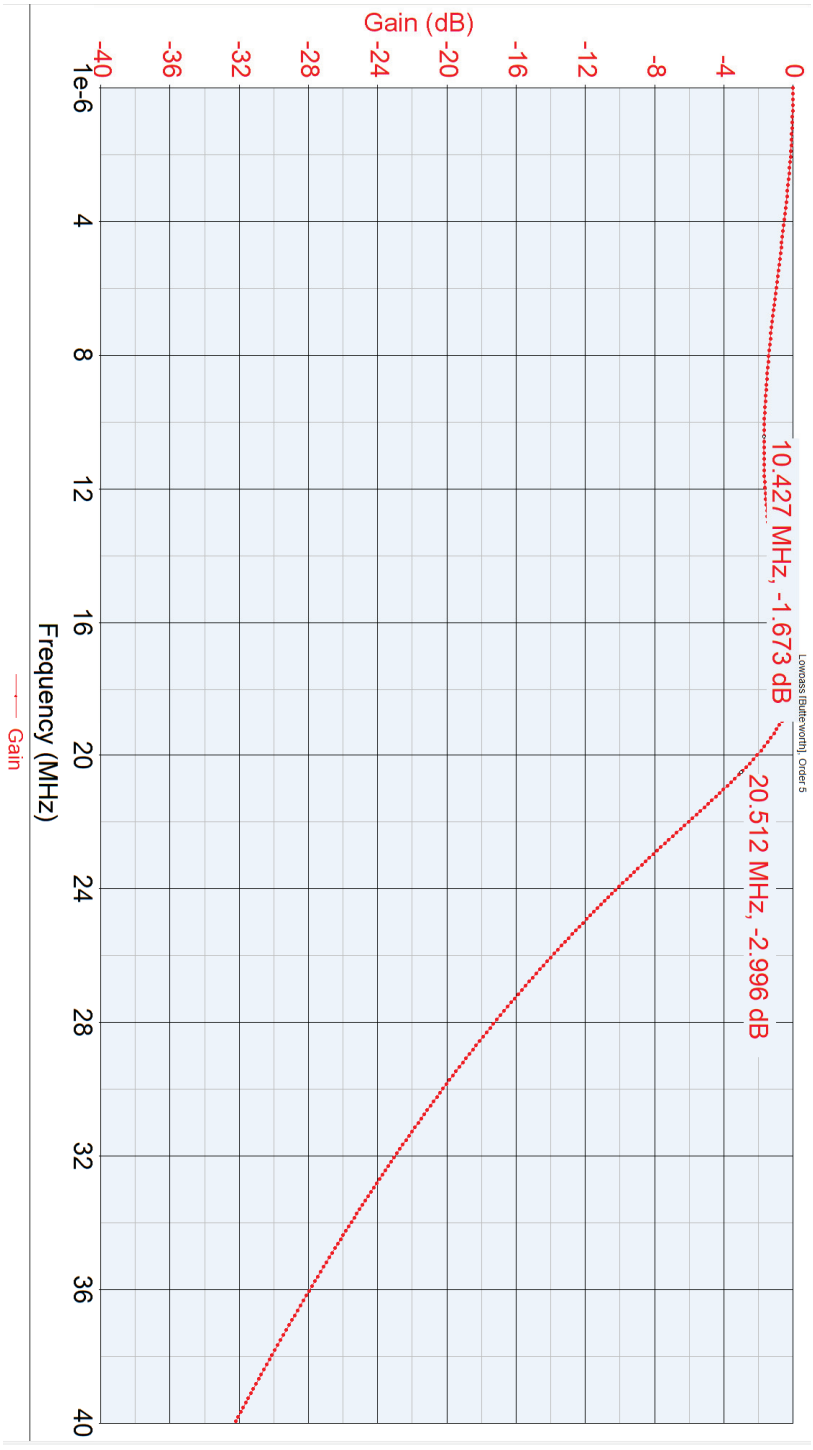


Figure 49. Frequency response of analog front end on USRP™ SBX-40 daughterboard.

Bibliography

1. Agbeyibor, Richard C. *Secure ADS-B: Towards Airborne Communications Security in the Federal Aviation Administration's Next Generation Air Transportation System*. Master's thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH, March 2014.

2. Bellare, Mihir, Phillip Rogaway, and Terence Spies. "The FFX Mode of Operation for Format-Preserving Encryption", February 2010. Report to NIST describing FFX algorithm.

3. Cenciotti, David. "U.S. airborne communication plane could be tracked on the Web for 9 hours during air strike that killed Taliban leaders in Afghanistan". `http://theaviationist.com/2014/08/13/bacn-supports-air-strike-afghanistan/`, August 2014.

4. Daemen, Joan and Vincent Rijmen. *The Design of RijndaeL:AES-The Advanced Encryption Standard*. 2002.

5. Drange, Geir. "Ultimate CRC", 2013. Available at `http://opencores.org/project,ultimate_crc`.

6. "USRP™ Bandwidth". World Wide Web Page, 2015. Available at *http://www.ettus.com/kb/detail/usrp-bandwidth*.

7. Federal Aviation Administration. "Chapter 1: IFR Operations in the National Airspace System". *FAA-H-8261-1A Instrument Procedures Handbook*, 1–32. URL `http://www.faa.gov/regulations_policies/handbooks_manuals/aviation/instrument_procedures_handbook/media/CH-01.pdf`, 2007.

8. Federal Aviation Administration. "Automatic Dependent Surveillance-Broadcast (ADS-B) Out Performance Requirements to Support Air Traffic Control (ATC) Service; Final Rule". *Federal Register*, 75(103):30106 – 30195, May 2010.

9. Federal Aviation Administration. "JO 7110.65V, Air Traffic Control". *Air Traffic Organization Policy*, Chapter 2, 2014.

10. Finke, Cindy, Jonathan Butts, and Robert Mills. "ADS-B encryption: confidentiality in the friendly skies". *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, 9. ACM, 2013.

11. Finke, Cindy D. *Format Preserving Encryption: Evaluating FFX for Use Within the NextGen Air Traffic Control System*. Master's thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH, March 2013.

12. "NATO air strike 'kills two children' in south-eastern Afghanistan". http://www.theguardian.com/world/2013/mar/30/nato-air-strike-children-afghanistan, August 2014.

13. Headquarters U.S. Air Force, USAF XOR-GANS. "Military Unique Applications for ADS-B". URL http://adsb.tc.faa.gov/WG6_Meetings/Meeting4/242A-WP-4-10%20Military%20Apps.pdf, 2001.

14. International Telecommunication Union. "Recommendation M.1371-5: Technical Characteristics for an Automatic Identification System using Time-Division Multiple Access in the VHF Maritime Mobile Band". *ITU-R*, 2014.

15. Jochum, John R. *Encrypted Mode Select ADS-B for Tactical Military Situational Awareness*. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, April 2002.

16. Koziel, Eric A. *Effects of Architecture on Information Leakage of a Hardware Advanced Encryption Standard Implementation*. Master's thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH, September 2012.

17. Magazu, Dominic. *Exploiting the Automatic Dependent Surveillance-Broadcast System via False Target Injection*. Master's thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH, March 2012.

18. McCallie, Donald L. *Exploring Potential ADS-B Vulnerabilities in the FAA's NextGen Air Transportation System*. Master's thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH, June 2011.

19. Parhi, K. K. *VLSI Digital Signal Processing Systems: Design and Implementation*. John Wiley, 1999.

20. Purton, L, Hussein Abbass, and Sameer Alam. "Identification of ADS-B system vulnerabilities and threats". *Australian Transport Research Forum, Canberra*. 2010.

21. RTCA Special Committee 186. "DO-260B with Corrigendum 1, Minimum Operational Performance Standards for 1090 MHz Extended Squitter Automatic Dependent Surveillance - Broadcast (ADS-B) and Traffic Information Services - Broadcast (TIS-B)". *ADS-B 1090 MOPS*, 2011.

22. Shirey, R. "Internet Security Glossary, Version 2". *IETF RFC4949*, 2007.

23. Wiglesworth, Jim. "NextGen Overview". *Part 145 Safety Summit*. Southern Region NextGen Branch, March 2014.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
26–03–2015	Master's Thesis	Sep 2013 — Mar 2015

4. TITLE AND SUBTITLE

Design and Characterization of a Secure Automatic Dependent Surveillance-Broadcast Prototype

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Heruska, Benjamin N., Captain, USAF

5d. PROJECT NUMBER

N/A

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-15-M-041

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory
Avionics Vulnerability Mitigation Branch
2241 Avionics Cir
WPAFB, OH 45433-7334
POC: Mr. Steven E. Stokes, Program Manager
(937) 528-8035 steven.stokes@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RYWA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

During sensitive military operations, such as air operations in theater, the broadcasting of ADS-B messages poses a serious security concern, but the small message payload of ADS-B transmissions renders encryption standards such as AES unsuitable. Format-preserving encryption (FPE), a technique already in use on small message sizes such as credit card numbers, provides a suitable implementation of strong symmetric key encryption for the military context. This research proposes and details a Secure ADS-B (SADS-B) system which utilizes a bump-in-the-wire approach for encryption and decryption of ADS-B messages using FPE, going on to then characterize the prototype's performance using the following metrics: detection and error rates, operational latency, and utilization of resources on the underlying field programmable gate array (FPGA) hardware. Findings include sub-millisecond operational latency, full data rate throughput capability for ADS-B, and approximately 50% utilization of the available FPGA resources. Overall, findings suggest that a layered security system such as SADS-B is suitable for adding confidentiality to ADS-B.

15. SUBJECT TERMS

NextGen, ADS-B, FAA, FPGA, format preserving encryption, FFX

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Robert F. Mills, PhD (ENG)
U	U	U	UU	78	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636 x4527 Robert.Mills@afit.edu