NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

IMPLEMENTATION OF A MULTIPLE LOW-RATE SAMPLER DETECTOR WITH DUAL ADC CARD AND FPGA USING VERILOG HDL

by

Canisio Barth Jr.

September 2020

Thesis Advisor: Ric Romero
Co-Advisor: Douglas J. Fouts

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2020	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
IMPLEMENTATION OF A MULTIPLE LOW-RATE SAMPLER DETECTOR WITH DUAL ADC CARD AND FPGA USING VERILOG HDL

5. FUNDING NUMBERS

6. AUTHOR(S) Canisio Barth Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

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

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

Multiple Low-Rate Sampling (MLRS) technique offers a method in which the detection of specific wideband signals can be achieved using an effective sampling rate lower than what is required by the Nyquist-Shannon theorem. This thesis presents a hardware implementation of a radar detector that utilizes a two-channel MLRS receiver. We apply the MLRS theory using commercial-off-the-shelf analog-to-digital converters and a field-programmable gate array (FPGA). MLRS is implemented using the Verilog hardware description language. Additionally, aiming to reduce the computational cost and the overall signal processing complexity, we propose a fixed-point composite detector based on a single threshold derived from the multiple spectral dominant parts of the signal. The composite detector is implemented in the FPGA and is evaluated using external analog signals obtained from EM-simulated target responses. Applying two matched filters yet using one modified threshold introduced by the composite detection approach, the MLRS detector significantly increases the probability of detection while compensating for the increased false alarm rate.

14. SUBJECT TERMS	**15. NUMBER OF PAGES**
low-rate sampling, Multiple Low-Rate Sampling, MLRS, Nyquist, matched filter, simulated target, fixed-point, hardware implementation, FPGA, ADC, Verilog, HDL	77
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

IMPLEMENTATION OF A MULTIPLE LOW-RATE SAMPLER DETECTOR WITH DUAL ADC CARD AND FPGA USING VERILOG HDL

Canisio Barth Jr.
Lieutenant, Brazilian Navy
BCE, Ponta Grossa State University—Brazil, 2009
BT, Federal University of Technology—Brazil, 2010

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2020

Approved by: Ric Romero
Advisor

Douglas J. Fouts
Co-Advisor

Douglas J. Fouts
Chair, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Multiple Low-Rate Sampling (MLRS) technique offers a method in which the detection of specific wideband signals can be achieved using an effective sampling rate lower than what is required by the Nyquist-Shannon theorem. This thesis presents a hardware implementation of a radar detector that utilizes a two-channel MLRS receiver. We apply the MLRS theory using commercial-off-the-shelf analog-to-digital converters and a field-programmable gate array (FPGA). MLRS is implemented using the Verilog hardware description language. Additionally, aiming to reduce the computational cost and the overall signal processing complexity, we propose a fixed-point composite detector based on a single threshold derived from the multiple spectral dominant parts of the signal. The composite detector is implemented in the FPGA and is evaluated using external analog signals obtained from EM-simulated target responses. Applying two matched filters yet using one modified threshold introduced by the composite detection approach, the MLRS detector significantly increases the probability of detection while compensating for the increased false alarm rate.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

2C	two's complement
ADC	analog-to-digital converter
CAD	computer-aided design
COTS	commercial-off-the-shelf
CS	compressed sensing
CST	computer simulation technology
DAC	digital-to-analog converter
DDC	digital down converter
DDR	double data rate
DSP	digital signal processing
FIR	finite impulse response
FMC	FPGA Mezzanine Card
FPGA	field-programmable gate array
HDL	hardware description language
IC	integrated circuit
IF	intermediate frequency
I/O	input/output
ILA	integrated logic analyzer
IP	intellectual property

LRT likelihood ratio test

LUT look-up table

LVCMOS low-voltage complementary metal oxide semiconductor

LVDS low-voltage differential signaling

MAC multiply-accumulate operation

MF matched filter

MLRS multiple low-rate sampling

MMCX micro-miniature coaxial

MSB most significant bit

NP Neyman-Pearson

NYFR Nyquist Folding Receiver

P_d probability of detection

P_{fa} probability of false alarm

PL programmable logic

PLL phase-locked loop

PROM programmable read-only memory

PS processor system

RAM random-access memory

RF radio frequency

RTL register-transfer level

SNR signal-to-noise ratio

SPI serial peripheral interface

SSE sum squared error

SoC system-on-chip

TI Texas Instruments

VIO virtual input/output (I/O)

VSG vector signal generator

WGN white Gaussian noise

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

I would like to recognize the invaluable contribution of my advisor, Dr. Ric Romero, who provided guidance and motivation during this challenging mission. It was a privilege working with such an extraordinary professional with immense theoretical knowledge and industry experience. I also would like to express my sincere appreciation to my co-advisor, Dr. Douglas Fouts, for offering his guidance. This work would not have been possible without his continuous support.

I would like to thank USN LT Adam Waymouth for his help in many experiments in this thesis and also for his camaraderie during my time at NPS. Additionally, I acknowledge the technical support of Mr. P. Balasubramanium during the experiments with the signal generators.

I would like to thank the Brazilian Navy, represented by the Brazilian Navy Research Institute, for funding this degree and giving me a unique opportunity that has greatly expanded my technical and life skills.

I wish to show my gratitude to my parents, Elizete and Canisio, for their inestimable contribution to this journey. All my achievements were made possible because of their unlimited support and guidance throughout my entire life.

Finally, I am extremely grateful to my girlfriend, Julienne, who has made countless efforts to help me get to this point. Thank you for your support.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

Modern radar and electronic warfare systems require the estimation and detection of radio frequency (RF) signals over an extremely wide frequency bandwidth. Even with state of the art receivers, there are physical limitations with the analog-to-digital converter (ADC) sampling rate, which restricts the amount of bandwidth that can be digitized and limits the ability to process the RF spectrum directly.

In an effort to reduce the sampling rate for certain wideband signals, a receiver architecture was proposed in [1] that takes advantage of the resonant frequency components of the signal of interest. It is called a multiple low-rate sampling (MLRS) receiver. The work presented here expands this technique by developing a comprehensive hardware model which is implemented with commercial-off-the-shelf (COTS) devices. We also investigate new techniques in the detection of a wideband signal using only its dominant components with the intent of reducing the overall data processing required.

1.1 Background

It is widely known that the ADC subsystem is one of the most critical components of an RF receiver. A traditional architecture usually consists of an analog pre-processing front-end, where first stages of filtering and amplification are applied, and an ADC block, where the continuous-time signal is sampled and quantized to be properly represented in the digital domain. Next, the discrete-time digital signal is transferred to a signal processing block, where various functions like detection, signal reconstruction and estimation are performed according to the application.

The minimal rate requirement to sample a continuous-time signal to the discrete-time signal is dictated by the Nyquist-Shannon sampling theorem. In summary, the sampling theorem states that the information contained in a signal can be completely reconstructed when the sampling rate is at least two times the highest frequency component present in the waveform. However, this definition relies on the concept of a perfectly bandlimited signal, that is, a signal with infinite time duration and zero energy outside of its bandwidth. Because a

theoretical signal like this does not practically exist, most receivers apply sampling rates greater than the Nyquist-Shannon limit in order to avoid aliasing and distortion. These greater sampling rates increase the complexity and cost of an ADC subsystem.

1.2 Previous Analysis

Given the limitations imposed by the analog-to-digital conversion, many surveys have addressed the difficulty of sampling large bandwidth signals. Exploiting the general sparsity of RF signals, Donoho [2] proposed the concept of compressed sensing (CS), in which the waveform is sampled based on the information content rather than the frequency domain characteristics. No prior knowledge of the signal is required by this method; however, the signal is assumed to be compressible by a known transform coding. Inspired by the CS development, the Nyquist Folding Receiver (NYFR), originally presented by Fudge et al. [3], proposed a non-uniform sampling scheme in which multiple frequency regions, bounded by the Nyquist-Shannon limit, are folded into a narrow bandwidth before entering the ADC. While this approach uses a wideband pre-selection filter in place of the traditional anti-aliasing filter, unambiguous recovery is made possible by applying frequency modulated pulses that compress the RF signal.

Thus, under certain circumstances the CS and NYFR principles significantly reduce the sampling rate, which bypasses the Nyquist-Shannon limit at the cost of increasing the digital signal processing (DSP) burden in information recovery. Considering the full bandwidth of the signals of interest, MLRS also effectively samples below the Nyquist rate. It is important to mention, however, that with respect to the resonant frequency components only, the ADCs of the receiver proposed by the MLRS method do not run at a sampling frequency under the Nyquist-Shannon limit. As detailed in Chapter 2, the effective reduction comes from the discarding of low level energy portions of the signal environment.

1.3 Objective

Following the discussion in [1] and [4], the primary goal of our study is to revisit the MLRS analysis and develop a more complete hardware model for the receiver architecture using re-configurable logic and the Verilog hardware description language (HDL). Our approach differs from the preceding implementation of MLRS in various aspects. First, the test signals

are captured from an analog source and sampled by a dual-channel ADC before being processed in the field-programmable gate array (FPGA). Second, the hardware description and logic routines for the communication between the ADC and the FPGA, as well as the DSP applied by the MLRS detector, are coded directly in Verilog, with no automatic HDL generation. Third, our design is exclusively implemented in the programmable logic of the FPGA, with no utilization of a microprocessor. Finally, we propose a composite detector with a single threshold, attempting to improve the *probability of detection (P_d)* while maintaining the desired *probability of false alarm (P_{fa})* in a simplified scheme without the use of the full wideband signal as reference.

1.4 Thesis Organization

This thesis is organized as follows. Chapter 2 presents a review of the MLRS method with an overview of its architecture and the signals of interest. A background of the detection theory is shown in Chapter 3, along with the definition and simulation of a composite detector for the MLRS approach. Chapter 4 depicts the characteristics of the hardware parts used in our work. Chapter 5 describes the numbering format and details of the fixed-point representation required for the hardware implementation. In Chapter 6, we describe the development of the HDL-based version of the MLRS method applied to a detector. Finally, a summary of the results and suggestions for future research is presented in Chapter 7.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Review of MLRS

The multiple low-rate sampling (MLRS) technique, first investigated in [1] and [4], explores wideband signals that are not sparse in the time domain with the objective of reducing the sampling rate needed for this category of signals. Although the sampling theorem requires a sampling rate slightly greater than twice the bandwidth (Nyquist-Shannon limit), practical applications utilize higher sampling frequencies for detection and reconstruction of signals due to the nature of real-world signals and degradation effects such as noise. Thus, wideband signals imply the utilization of ADCs with high sampling rates, which may introduce technical and budgetary constraints.

Another investigation applying the MLRS concept was performed in [5]. The segmented band approach was explored in conjunction with a cognitive radar technique [6], in which improvements in the P_d are achieved by creating radar waveforms based on particular target responses. Furthermore, the work in [5] depicts practical target responses that corroborate the use of MLRS.

The receiver proposed in [1] was able to successfully lower the effective sampling rate for signals with multiple dominant portions at certain frequency bands. The architecture and procedures presented by [1] are summarized as follows. First, the wideband type signal was treated as a collection of dominant sub-regions. Next, each sub-region centered at its respective dominant frequency passed through a chain of filters and the baseband representation of each individual channel was obtained. The channels were upconverted to their original frequencies and added for reconstruction. The performance of the proposed architecture was evaluated using two parameters: sum squared error (SSE) and probability of detection compared to a matched filter (MF) receiver. MATLAB and Simulink models for the MLRS method provided similar results in comparison to a conventional high-rate sampler. The results from the FPGA simulation demonstrated a slight degradation in the P_d when compared to the theoretical MF. However, the lower effective noise variance helped to improve the performance in signal recovery. Thus, the loss in the P_d can be seen as the trade-off between the MLRS and a conventional system.

There are three major differences between the MLRS implementation of [1] and the work herein. First, the test signals are simulated using an analog signal generator and sampled by a two-channel ADC board, as opposed to the internal simulations performed in [1]. Second, the hardware is developed using the Verilog HDL instead of being generated by Simulink tools. Third, the detection is performed using each dominant component using a composite detector and without utilizing the original wideband signal as the single matched filter for the receiver.

2.1 Signals of Interest

The waveforms for this study must satisfy the conditions presented in [1]. Let $x(t)$ be a wideband time domain signal of interest, with frequency spectrum $X(f)$. Assuming $X(f)$ contains M dominant regions across its frequency domain, it can be approximated as a summation of each M existing sub-bands

$$X(f) \approx \sum_{i=1}^{M} \psi_i(f), \tag{2.1}$$

where $\psi_i(f)$ is the i_{th} dominant portion of $X(f)$, with bandwidth B_i centered at a given f_i sub-carrier.

It is evident that with the approximation, the effective bandwidth B_E considered for sampling is

$$B_E = \sum_{i=1}^{M} B_i \leq B_W, \tag{2.2}$$

which is less or equal to the total bandwidth B_W.

For the purposes of this study, we assume a signal (e.g., a target response) such that the values M, ψ_i and B_i can fit the characteristics of the hardware available for our implementation, which will be detailed in Chapter 4. The top graph in Figure 2.1 shows the one-sided frequency spectrum of a test signal with $M = 2$, $\psi_1 = 20$ MHz, $\psi_2 = 25$ MHz and $B_1 = B_2 \approx 2$ MHz.

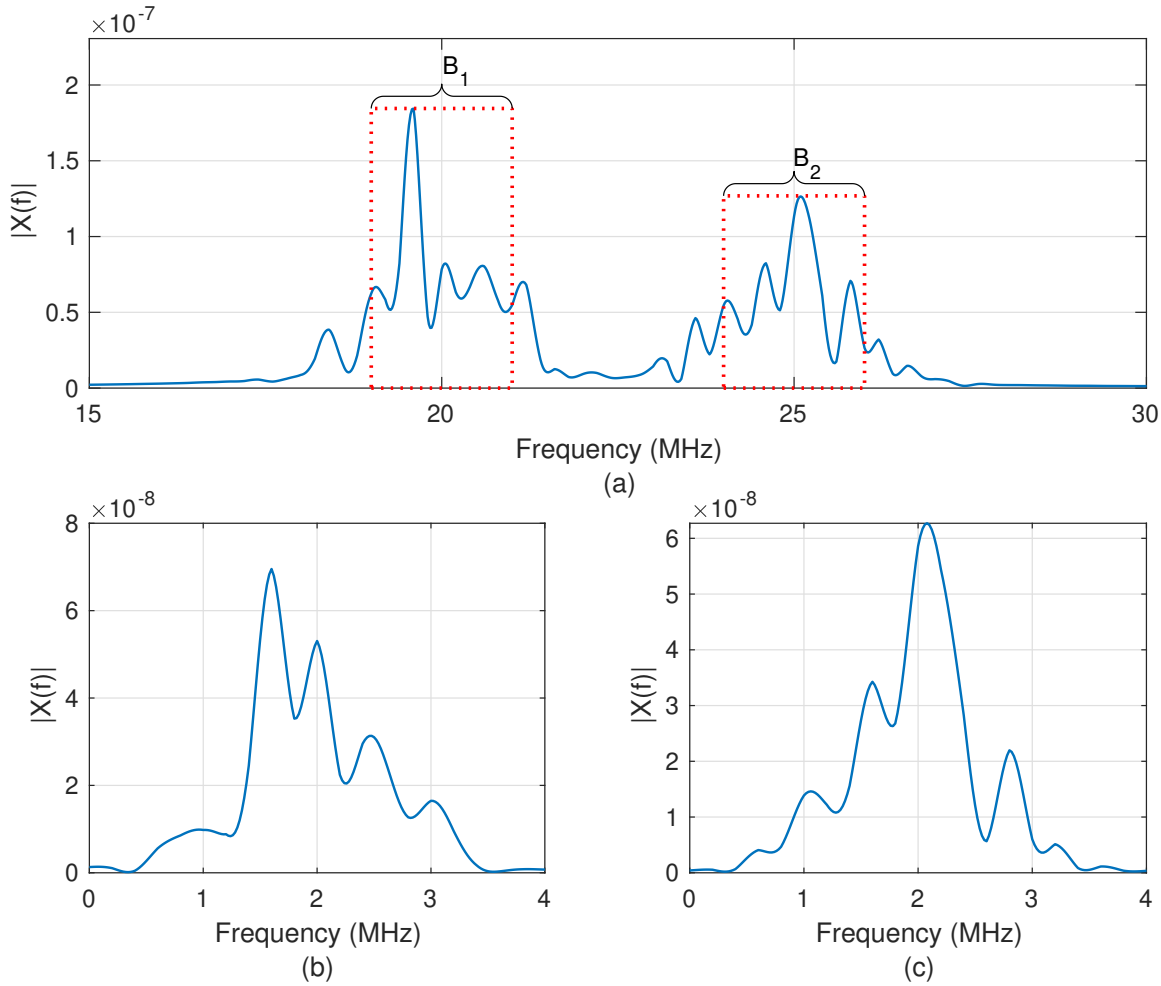


Figure 2.1. Generic test signal with $M = 2$. Original (a) , separated IF 1 (b) and separated IF 2 (c)

2.2 Dual-Channel MLRS Detector Architecture

The block diagram for the system architecture proposed in our research is presented in Figure 2.2. We assume that the main signal is already split into M dominant bands, which are downconverted to the required intermediate frequency (IF) in a previous stage. The bottom graphs on Figure 2.1 show both the dominant portions of the test signal downconverted to an $IF = 2$ MHz. In our implementation, the reception of the sub-carriers is implemented by connecting the ADC inputs to a dual-channel vector signal generator (VSG). We also presume that the undesired frequency components have been previously filtered and the

signal was amplified to the required levels before entering the sampling block. Thereafter, each sub-band is individually sampled by two separate ADCs and the digitized signals are sent to the FPGA block.

The baseband representation of each sub-carrier is then obtained through a digital down converter (DDC) block and subsequently processed by a MF detector. For design simplicity, the traditional low-pass filters are not implemented. Next, the detection process takes place, comparing the outputs of the MF against a reference threshold. The reference signals for detection are obtained from the baseband representation of the two dominant signals.

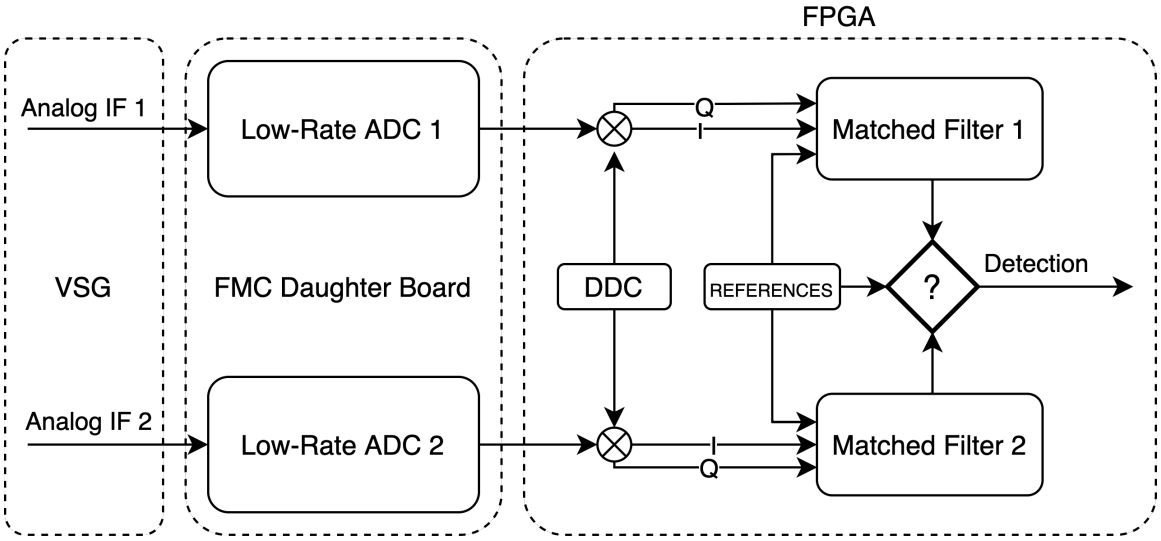


Figure 2.2. Block diagram of a dual-channel MLRS detector

2.3 Use of EM-Simulated Target Response for a Dual-Channel MLRS Detector

The signal shown in Section 2.1 was specifically designed to have two major dominant bands for a dual-channel MLRS receiver. However, we can use previous works in cognitive radar and target signature identification to evaluate the presence of dominant components in practical targets. The research presented in [7] demonstrates the frequency responses for several ground vehicles. These frequency responses were obtained using a computer

simulation technology (CST) software that applies *ray tracing* technique against full-sized computer-aided design (CAD) targets. In particular, the BMW M6 sedan shown in Figure 2.3 has the frequency responses (depending on aspect angle) that match well the two dominant-band scenario and consequently the dual-channel MLRS receiver.

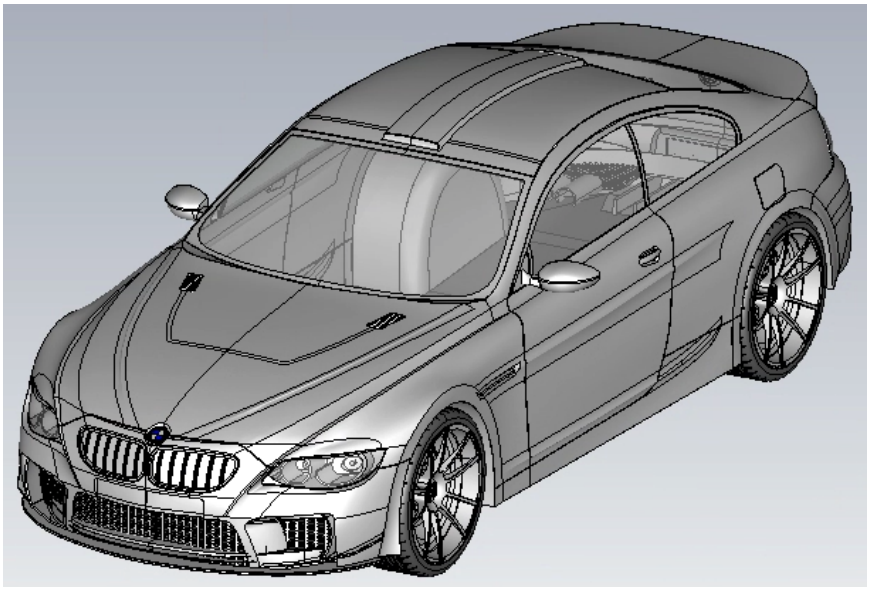


Figure 2.3. BMW M6 CAD model

Figure 2.4 shows the squared magnitude of the electric field versus the normalized frequency response at a specific aspect angle for the BMW M6. Thus, to use this target response in our test hardware, we first need to scale the frequencies of each dominant portion to the proper IF, filtering the components outside of the bandwidth of interest. Next, the time-domain waveforms can be obtained by applying the inverse Fourier transform. Finally, the amplitudes of the time signals are adjusted according to the desired signal energy. As mentioned earlier, we assume that these steps are done outside of the proposed architecture shown in Figure 2.2, with the DSP steps executed in MATLAB and the generation of the analog signals performed by the VSG.

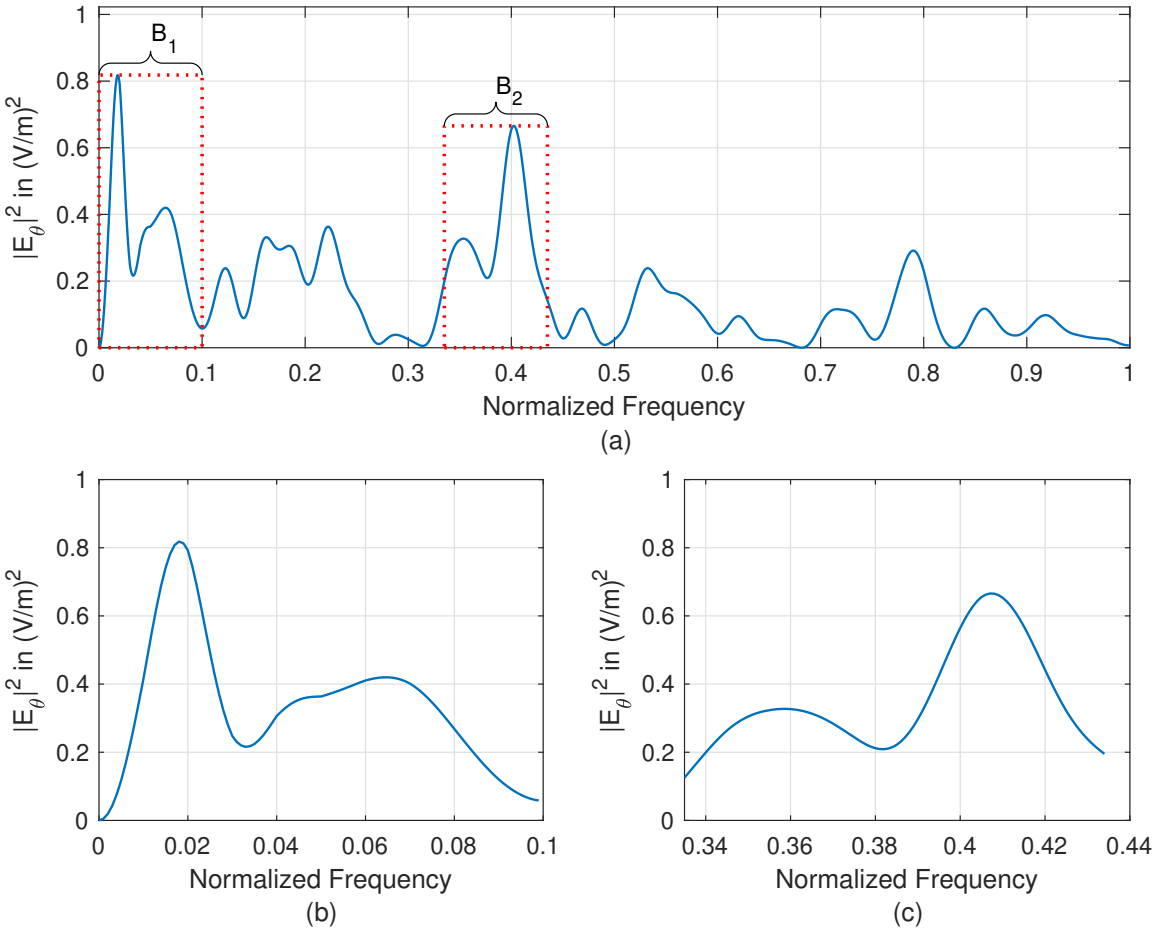
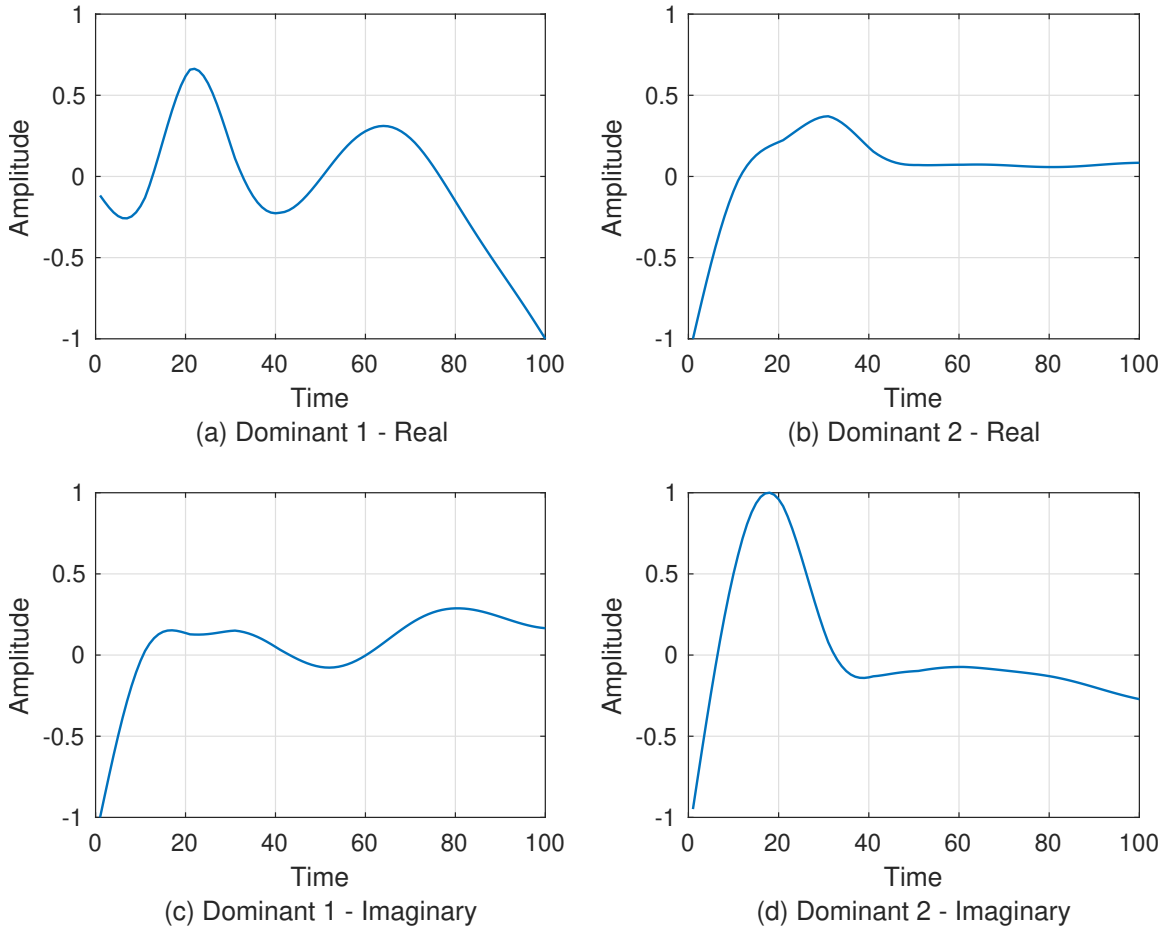


Figure 2.4. Practical target frequency response. Original (a), separated IF 1 (b) and separated IF 2 (c). Azimuth $\theta_{az} = 45°$ and Elevation $\theta_{el} = 0°$

Figure 2.5 shows the complex time domain signal of each dominant portion of this target response, which can be transformed in the analog signals by the VSG.



(a) Dominant 1 - Real

(b) Dominant 2 - Real

(c) Dominant 1 - Imaginary

(d) Dominant 2 - Imaginary

Figure 2.5. Time domain signal from dominant portions

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
MLRS Composite Detector

The final product of our work is a basic radar detector that takes advantage of the MLRS method. As our architecture utilizes a dual-channel MLRS receiver, we propose a novel approach in which the detection procedure is performed simultaneously on each dominant frequency component of the target response, as opposed to a traditional detector that acts on a single stream of the given received signal. This adds a new layer in the decision process, which we intend to explore in later sections of this chapter after reviewing the relevant background from conventional detection theory.

3.1 Detection Theory Review

The groundwork for the MLRS detector proposed in this study is based on the approach presented by [8], in which we assume a known deterministic signal embedded in white Gaussian noise (WGN). As the goal is to maximize the probability of detection (P_d), conditioned to a constant probability of false alarm (P_{fa}), the Neyman-Pearson (NP) criterion is ideal to our case. To solve the detection problem, our receiver must be able to distinguish between two fundamental hypotheses:

$$\begin{aligned} &\mathcal{H}_0 : target\ not\ present \\ &\mathcal{H}_1 : target\ present. \end{aligned}, \tag{3.1}$$

If $s[n]$ is the known discrete time signal to be detected, with N samples, we can mathematically rewrite (3.1) as

$$\begin{aligned} &\mathcal{H}_0 : x[n] = w[n] \qquad &n = 0, 1, \ldots N-1 \\ &\mathcal{H}_1 : x[n] = w[n] + s[n] \qquad &n = 0, 1, \ldots N-1 \end{aligned}, \tag{3.2}$$

where $x[n]$ is the received signal and $w[n]$ is assumed to be WGN with variance σ^2. In other words, under $\mathcal{H}_0$ only noise exists on the receiving channel whereas under $\mathcal{H}_1$ a version of the sent signal $s[n]$ corrupted by noise is present. It is well known from the NP theorem that

the detector must decide $\mathcal{H}_1$ based on the likelihood ratio test (LRT)

$$L(\boldsymbol{x}) = \frac{p(\boldsymbol{x}; \mathcal{H}_1)}{p(\boldsymbol{x}; \mathcal{H}_0)} > \gamma, \tag{3.3}$$

where $\boldsymbol{x}$ is the vector form of $x[n]$, the threshold γ is a scalar value (which is function of the desired P_{fa}) and $L(\boldsymbol{x})$ is denoted as the *likelihood ratio*.

It follows that, for the WGN case, $p(\boldsymbol{x}; \mathcal{H}_0)$ and $p(\boldsymbol{x}; \mathcal{H}_1)$ are both Gaussian distributions with known parameters. As noted in [8], the NP detector is simplified to a threshold being compared to the *test statistic* $T(\boldsymbol{x})$, where $\mathcal{H}_1$ decided when

$$T(\boldsymbol{x}) = \sum_{n=0}^{N-1} x[n]s[n] > \gamma'. \tag{3.4}$$

In summary, the target is deemed *present* when $T(\boldsymbol{x})$ in (3.4) exceeds γ', which is a reformulated threshold obtained during the simplification process of the LRT. Obviously, when $T(\boldsymbol{x}) \leq \gamma'$ then target is deemed *NOT present*. As discussed in [8], γ' is a function of σ^2, the expected received energy $\mathcal{E}$ and the desired P_{fa}

$$\gamma' = \sqrt{\sigma^2 \mathcal{E}} Q^{-1}(P_{fa}), \tag{3.5}$$

with $Q^{-1}(\cdot)$ being the *inverse Q-function*.

The detection test of (3.4) is known as the replica correlator due to the fact that the received signal is correlated with a replica of the sent signal. Another interpretation that yields the same result is to consider $\boldsymbol{x}$ as the input to a finite impulse response (FIR) filter. If the impulse response of the filter is set to be a “mirrored" version of the original signal $\boldsymbol{s}$, the convolution at the instant $N - 1$ (when the last sample of the signal is processed) is equal to the summation portion of (3.4). This implementation of the NP detection using the FIR approach is know as the *matched filter (MF)*. The block diagram of Figure 3.1 summarizes the detection process.

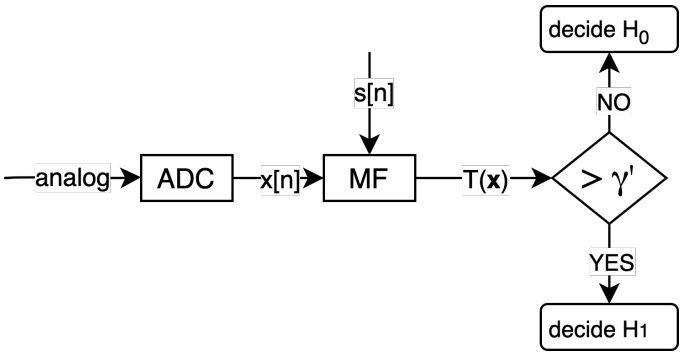


Figure 3.1. Block diagram of a traditional detector using a MF

A well-known property of the MF is that it maximizes the signal-to-noise ratio (SNR) at the FIR output. Thus the P_d is also optimal in the detection of a known deterministic signal in WGN, and as defined in [8], the performance of the MF can be evaluated for a given P_{fa} using

$$P_d = Q\left(\frac{\gamma' - \mathcal{E}}{\sqrt{\sigma^2 \mathcal{E}}}\right). \tag{3.6}$$

Usually, we wish to evaluate the detector in terms of the SNR for a desired P_{fa}; thus we can simply retain σ^2 constant and vary the signal energy based on the given SNR. As detailed by [8], the SNR at the MF output is

$$SNR = \frac{\mathcal{E}}{\sigma^2}. \tag{3.7}$$

Skipping the algebraic details, a combination of (3.5), (3.6) and (3.7) leads to

$$P_d = Q\left(Q^{-1}(P_{fa}) - \sqrt{SNR}\right). \tag{3.8}$$

For the purpose of completing this section, it is worth mentioning that from the same derivation of (3.6), the theoretical P_{fa} can be calculated for a given threshold with

$$P_{fa} = Q\left(\frac{\gamma'}{\sqrt{\sigma^2 \mathcal{E}}}\right). \tag{3.9}$$

3.2 Detection Approach for Dual-Channel MLRS

The objective of our detector is still to decide between the hypotheses shown in (3.1). However, we cannot apply directly the basic matched filter detector in (3.4) because we have two separated signals rather than one measurement vector $\boldsymbol{x}$. Available to our detector are two waveforms that are part of a single transmitted signal. One could argue that the *M-ary* detector described by [8] and [9] is the method to use, but that is better suited for classification than for the detection problem. Since we have two separate signals, it follows to use two MFs, one for each dominant band. However, we only have one threshold obtained from the original reference signal which is an issue and the topic of the next section.

3.2.1 MLRS Single Threshold

As shown on the top part of Figure 2.1, a certain amount of the signal energy is clearly lost in the process of selecting the dominant signal portions. Hence, to calculate our single threshold, it seems reasonable to take into consideration only the energy of the dominant parts as opposed to the entire signal energy, especially in the case that the target frequency response is assumed to be known.

A logical approach to solve this problem is to sum up the energy contained in both dominant regions and use (3.5) to obtain our desired single threshold. A part of this approach is to apply the MFs separately on each component and use the addition of the outputs in the test statistic of (3.4). As we shall see in the simulations of the next section, this approach confirms the theoretical P_d. Now, we investigate an alternative approach that improves P_d.

Our alternative approach is then to derive the threshold from the average energy of the dominant regions and to channelize the MLRS detector into two intermediate detectors and finally decide $\mathcal{H}_0$ or $\mathcal{H}_1$ based on the intermediary outputs. Therefore, if $\mathcal{E}_A$ and $\mathcal{E}_B$ represent the energy of the signal contained in each resonant region A and B, respectively, of the transmitted signal $\boldsymbol{s}$, then we can reformulate (3.5) as

$$\gamma'_{avg} = \sqrt{\sigma^2 \left(\frac{\mathcal{E}_A + \mathcal{E}_B}{2} \right)} Q^{-1}(P_{fa}). \tag{3.10}$$

3.2.2 Double Dominant Band MF Detector

At this point, we are able to apply the test statistic on each dominant band using the average threshold γ'_{avg}, and (3.4) is expanded to

$$\begin{aligned} T(\boldsymbol{x}_A) &= \sum_{n=0}^{N-1} x_A[n] s_A[n] > \gamma'_{avg} \\ T(\boldsymbol{x}_B) &= \sum_{n=0}^{N-1} x_B[n] s_B[n] > \gamma'_{avg} \end{aligned}, \tag{3.11}$$

where the subscripts A and B identify each sub-band component. Although we use a single threshold in both tests, we still need two different reference vectors to apply the dual MF. In our implementation we assume that $\boldsymbol{s}_A$ and $\boldsymbol{s}_B$ are obtained a priori from the original wideband signal.

It is important to recall that the *test statistic* $T(\boldsymbol{x})$ output is ultimately a logical value defined by the operator ">" in (3.4). However, by extending the test to a composite formulation, we create an additional decision layer connecting the test statistic and the final choice between the elementary hypotheses $\mathcal{H}_0$ and $\mathcal{H}_1$. Therefore, we need to define what logical operator to apply to the output of each detection test before finally deeming *target present* or *target not present*. Figure 3.2 shows the difference between the traditional detection scheme in Figure 3.1 and the composite detector we propose in our study.

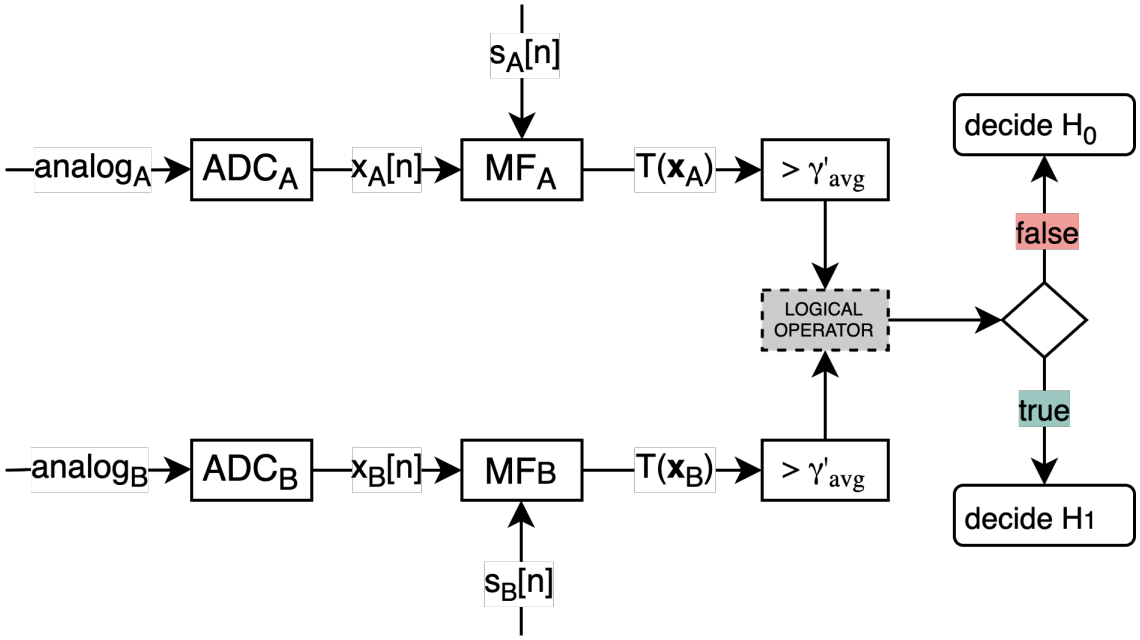


Figure 3.2. Block diagram of the MLRS composite detector

To fill in for the logical operator in the dashed box of Figure 3.2, the most intuitive assumption is to apply the *AND* function since the signal of interest is known to contain both components at the same time. This is clearly the more cautious approach with respect to the false alarm, since both detectors must reach the threshold whether it is truly $\mathcal{H}_0$ or $\mathcal{H}_1$. On the other hand, if we select the operator to be an *OR*, the final detector is relaxed and the primary assumption is that the P_d increases. From the same presumption, however, the P_{fa} would also increase.

3.3 Monte Carlo Simulations

To evaluate the proposed strategy and have a numerical baseline for the trade-off between the P_d, P_{fa} and the logical operator, we set up a simulation scheme using the *Monte Carlo* method, in which we evaluate the P_d against the SNR using simulation with multiple trials. The results for the P_d are shown in Figure 3.3, and are compared to the theoretical values from (3.6). The desired P_{fa} was set to 1×10^{-3} with 1×10^{6} trials.

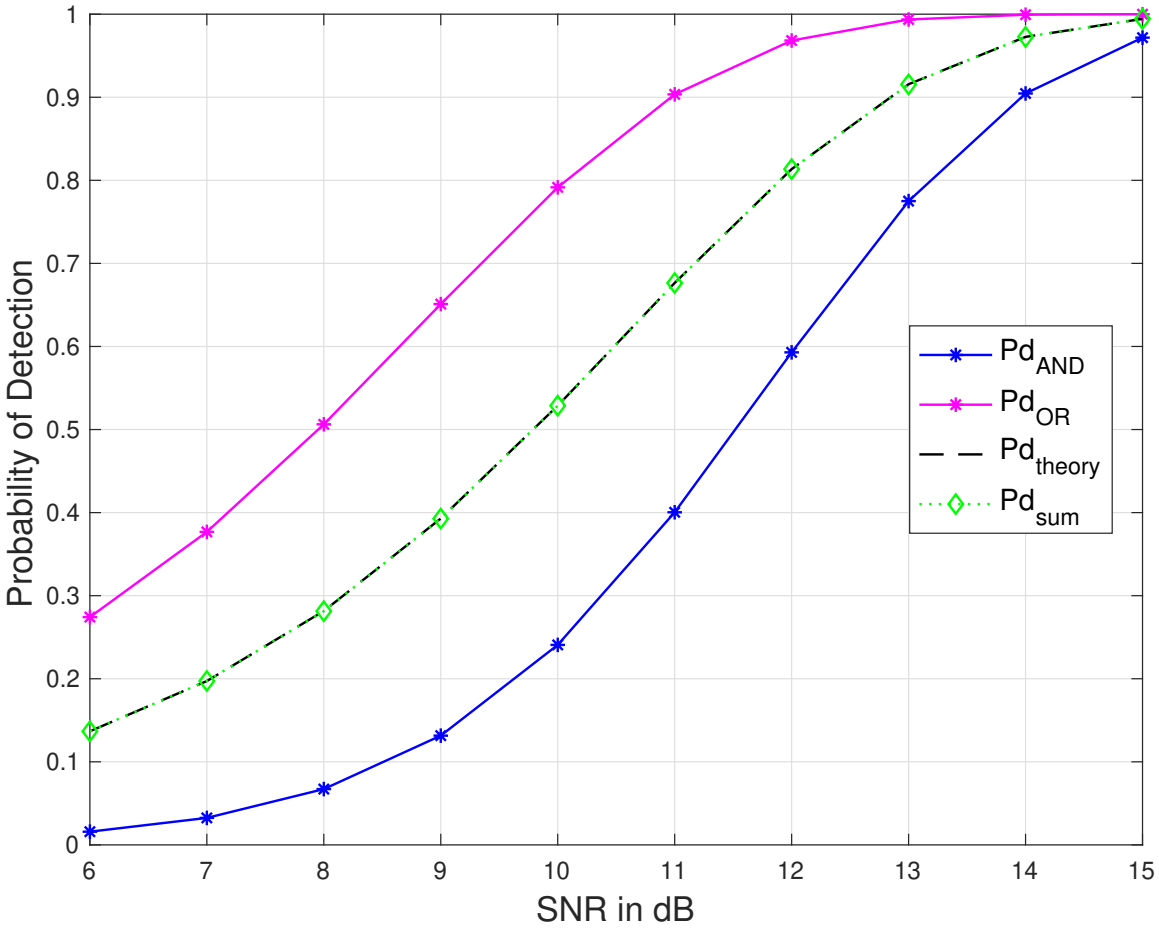


Figure 3.3. Composite detector P_d vs. SNR (dB). *AND* detector (blue), *OR* detector (magenta) and sum of components (green)

As mentioned earlier, the simulation results using the total energy of the relevant components agrees with the theoretical values. And as expected, the conservative *AND detector* lowers the P_d below the theoretical values, while the *OR detector* greatly increases it. Nevertheless, we must evaluate the impact of these detection schemes on the resulting P_{fa}. Figure 3.4 illustrates the same simulation scenario in which only the noise signal was input into the receiving channel.

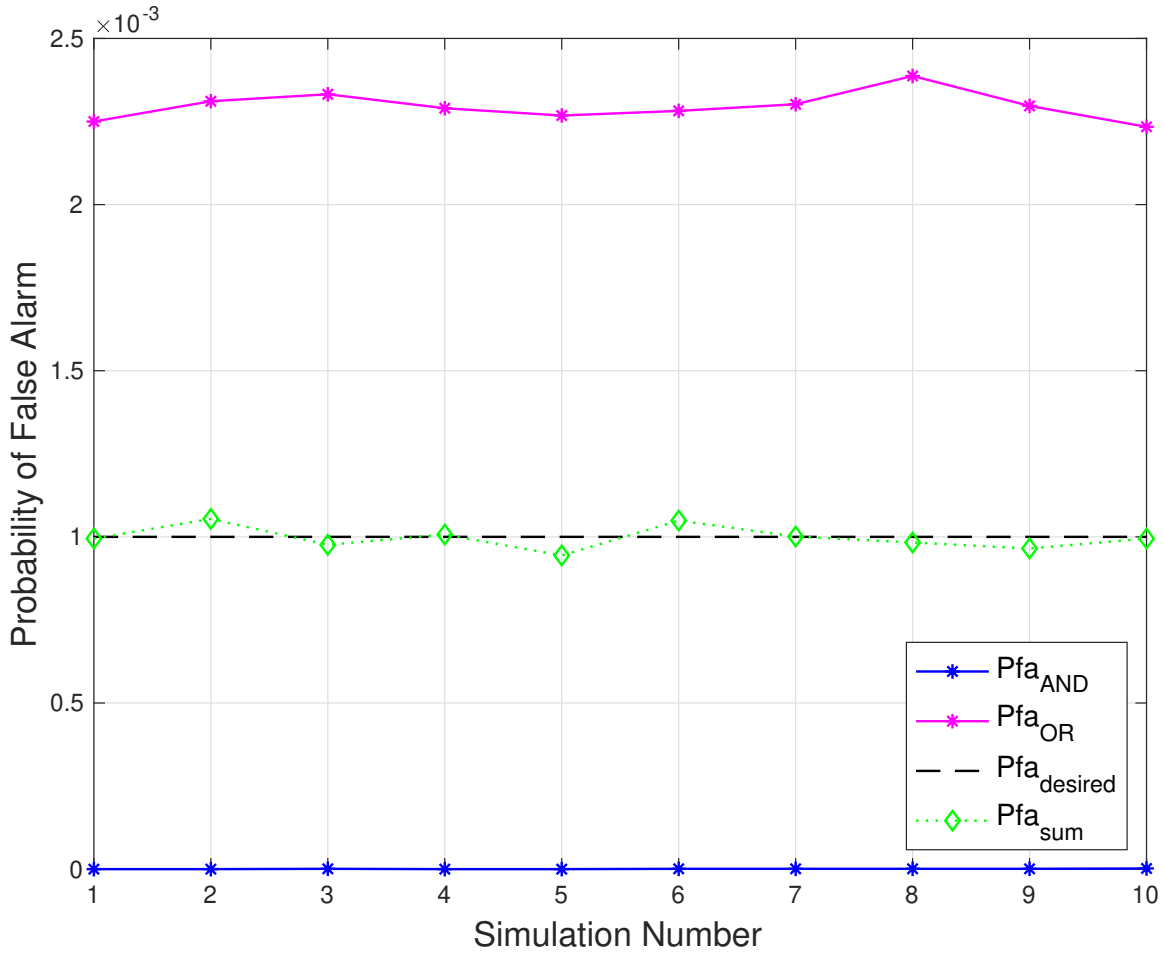


Figure 3.4. Composite detector P_{fa} vs. simulation number. *AND* detector (blue), *OR* detector (magenta) and "sum of components" approach (green)

The above mentioned "summed energy" approach is also simulated as a baseline comparison for P_{fa}. As anticipated in the previous section, it agrees with the specified value for the P_{fa}. The *AND detector* considerably lowers the false alarm rate while the *OR detector* degrades it to more than the double of the required probability.

3.3.1 Modified Threshold for the OR Detector

Given that the *OR detector* indeed improves the P_d but at the same time increases the P_{fa}, we now seek out a way of restoring the P_{fa} to the required value. One of the main differences between the traditional MF detector and the scheme proposed in this study is the calculation

of the detection threshold. Thus, trying to compensate for the loss in the P_{fa}, we propose a *scaled threshold* to be applied in the dual test statistic of (3.11) given by

$$\gamma'' = \kappa\gamma'_{avg}. \tag{3.12}$$

Using (3.10) we have

$$\gamma'' = \kappa\sqrt{\sigma^2\left(\frac{\mathcal{E}_A + \mathcal{E}_B}{2}\right)}Q^{-1}(P_{fa}), \tag{3.13}$$

where κ is a decimal value greater than one that seeks to restore required P_{fa}. By scaling up the average threshold, we aim to again meet the P_{fa} yet hopefully keep the P_d above the theoretical value. Table 3.1 shows the average false alarm rate for various κ.

Table 3.1. κ Multiplier comparison

$\boldsymbol{\kappa}$	$\boldsymbol{P_{fa} \times 10^{-3}}$		
	AND	OR	desired
1.000	0.0006	2.389	1.0
1.075	0.0001	1.138	
1.090	0.0001	0.990	
1.100	0.0001	0.892	

The closest value to restore the P_{fa} to 1×10^{-3} is $\kappa = 1.09$. However, we need to look at the P_d and P_{fa} curves with this new threshold. Figures 3.5 and 3.6 show the new results for the *OR* false alarm and detection curves, respectively.

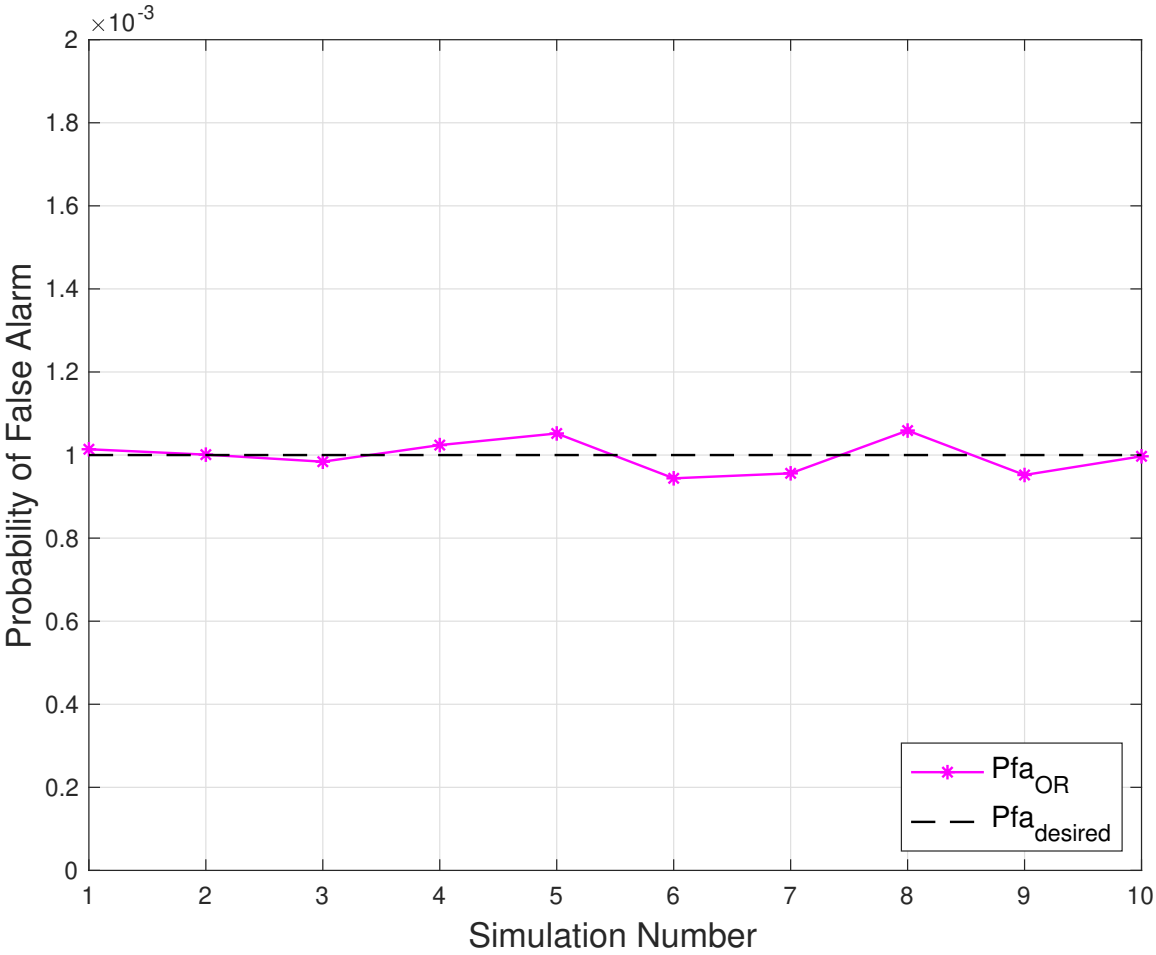


Figure 3.5. Composite *OR* detector P_{fa} vs. simulation number for $\kappa = 1.09$. *OR* detector (magenta), theoretical (black)

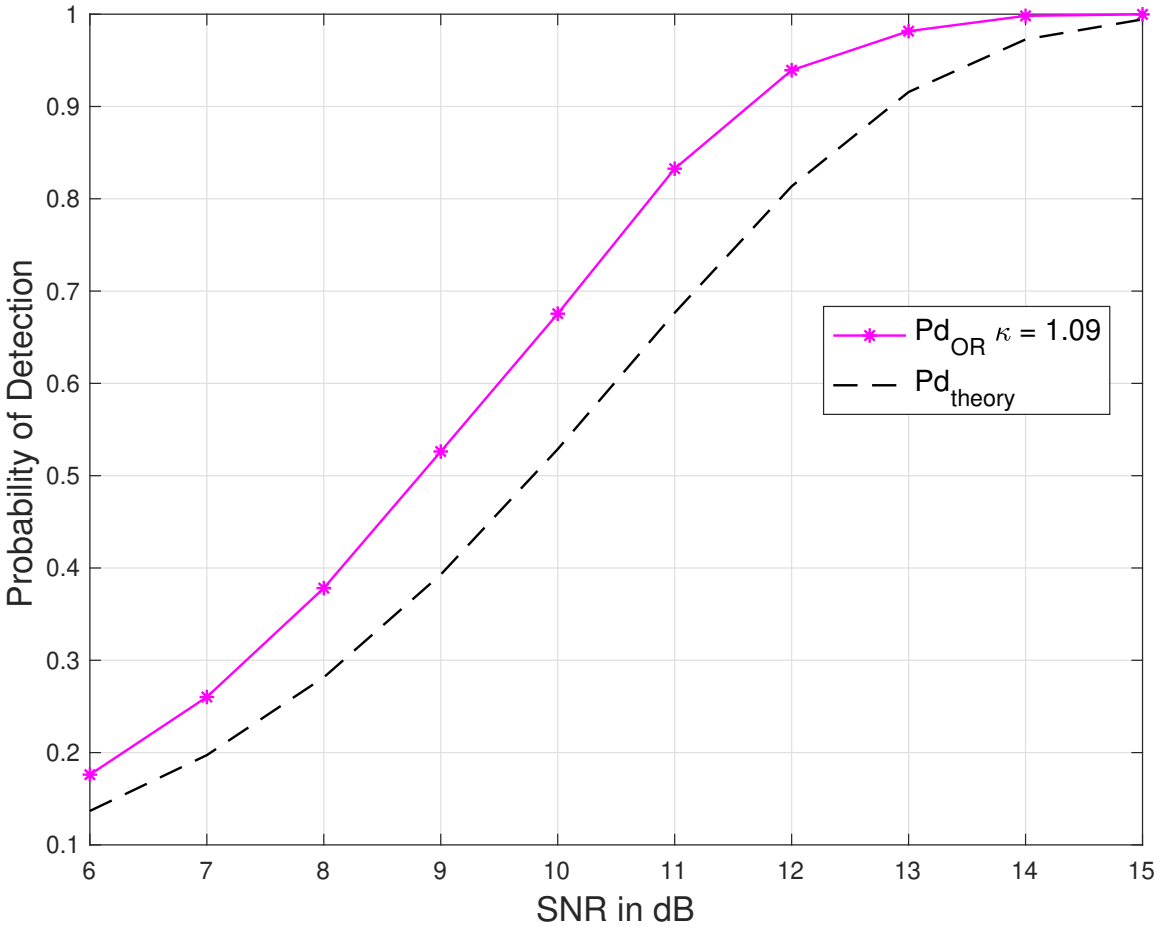


Figure 3.6. Composite *OR* detector P_d vs. SNR (dB) for $\kappa = 1.09$. *OR* detector (magenta), theoretical (black)

As can be seen in the last two figures, the new threshold coupled with the *OR* detector not only restores the desired P_{fa} but also increases P_d.

Considering the performance results from the simulations, we choose to keep the *OR detector* because it appears to be the worthwhile approach in terms of P_d. Yet it may be possible that the *AND detector* with a κ value less than one could lead to similar outcomes.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Hardware Components

As several aspects of the receiver implementation depend on the attributes of the chosen elements, in this chapter we highlight the most relevant electronic and physical characteristics of the components used in our work. With respect to our design, the architecture shown in Figure 2.2 can be divided in two main parts: the FPGA evaluation board, denoted as the carrier board, and the daughter board, which contains the ADC modules. The two circuit boards are connected by a hybrid interface used for configuration and data transfer. In summary, the FPGA sends control parameters to the daughter card, which digitizes the analog channels accordingly and streams the digital data back to the FPGA.

4.1 ZC706 FPGA Evaluation Board

The ZC706 is a hardware evaluation board utilizing the Xilinx XC7Z045 system-on-chip (SoC) devices. The XC7Z045 is a mid-range SoC, comprised of both a processor system (PS) and a programmable logic (PL) subsystem [10]. In addition to the PS and PL, the ZC706 provides features common to the developing and evaluation of embedded systems, namely various memory and input/output (I/O) devices, which implement the most common industry standards. One of them is the FPGA Mezzanine Card (FMC) expansion interface, an ANSI compliance form factor interface used in the interconnection of FPGAs and external circuit boards. Furthermore, users of the ZC706 also have access to the vendor development software package, the *Vivado Design Suite*, including a variety of intellectual property (IP) cores.

For what is proposed in this study, we will mostly be focused on the PL and the FMC interface. The PL contains the FPGA that will implement our HDL design for the MLRS system. The FMC will serve as the main interface between the FPGA and the two ADC modules.

4.2 FMC150 Daughter Board

The FMC150 daughter card provides a two-channel analog-to-digital converter (ADC) and digital-to-analog converter (DAC) compliant to the ANSI 57.1 standard (FMC). The ADC module is based on Texas Instruments (TI) ADS62P49 ADC, which is a dual channel 14-bit output device with a variable sampling rate. The FMC150 sampling system can be clocked either by an internal phase-locked loop (PLL) or by an external reference. All clock related functions are implemented by the TI CDCE72010 integrated circuit (IC). A voltage and temperature monitor device and an external trigger circuit are also available in the package [11]. Figure 4.1 shows the ZC706 with the FMC150 attached to one of its expansion connectors.

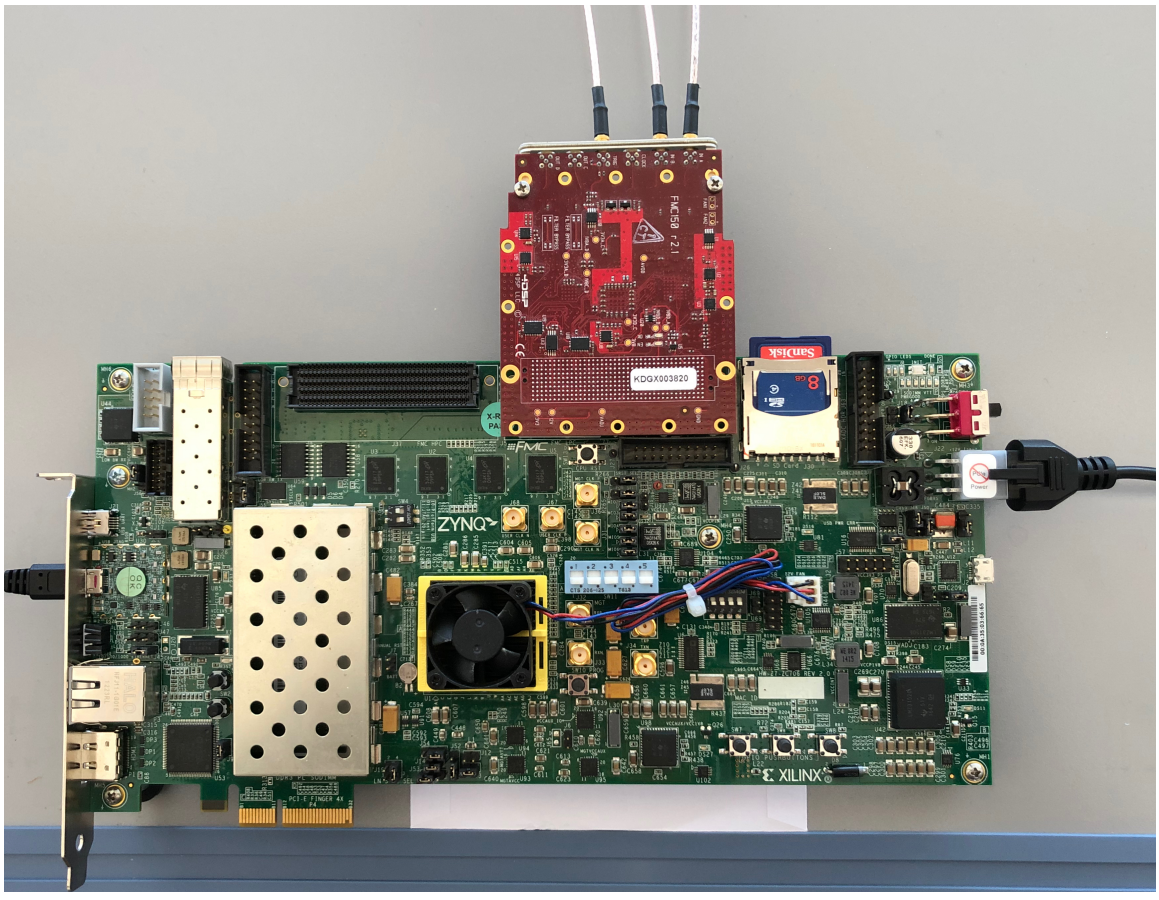


Figure 4.1. ZC706 FPGA evaluation board with FMC150 daughter card

4.3 Connection Standards and Protocols

In Figure 4.2, we show an overview of the standards used between the FPGA and the external board, as well as the connections with the analog environment. The analog channels are connected through micro-miniature coaxial (MMCX) plugs and the input signals are AC-coupled with load impedance matched to 50 Ohms. The digital outputs of the ADC blocks and respective control signals are available to the FPGA device via the FMC connector. The data stream produced by each channel of the ADS62P49 is presented to the FPGA using a double data rate (DDR) bus on the FMC interface, along with a reference clock. The configuration and control functions must be realized by the FPGA via serial peripheral interface (SPI).

The digital signaling for the data, clock and trigger uses the low-voltage differential signaling (LVDS) standard, whereas the control signals are transmitted using low-voltage complementary metal oxide semiconductor (LVCMOS) technology. Given the differences, the respective pins on the FPGA side must be mapped accordingly in the HDL design.

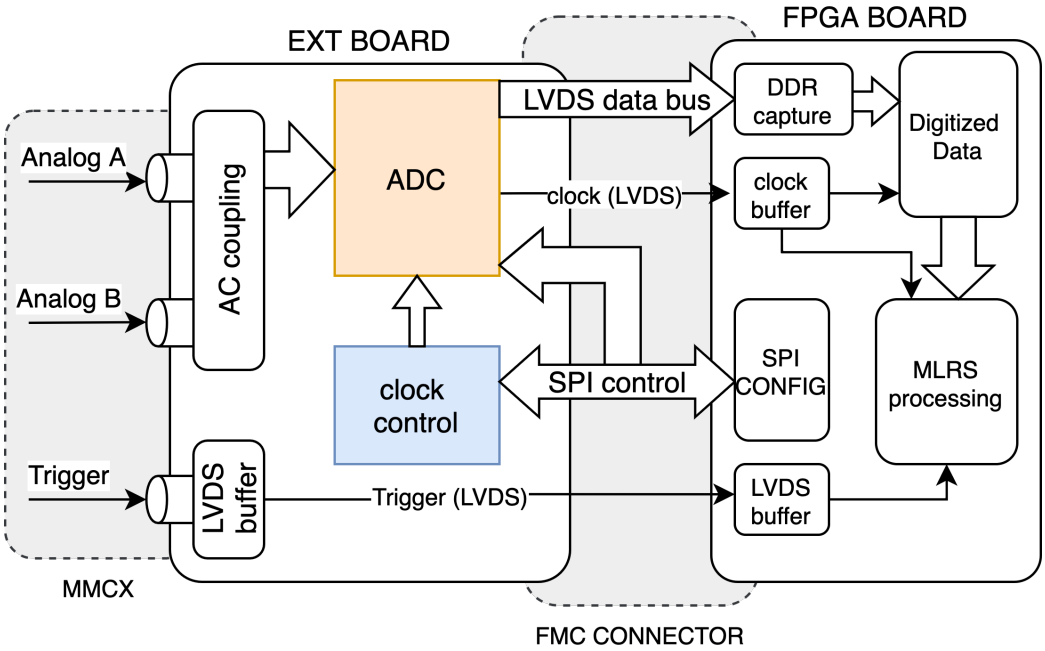


Figure 4.2. Block diagram of interfaces

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Fixed-Point Representation

In computer arithmetic, there are two major schemes to represent real numbers: Fixed-point notation, in which the integer and fractional parts of the real value have a fixed number of bits, and floating-point notation, that uses a standardized format to allow for variations in the position of the decimal point, similar to scientific notation. The floating-point scheme assures a much higher dynamic range and can normally provide higher accuracy [12], but its complex organization increases hardware cost, power consumption, and execution time. On the other hand, the simplicity of the fixed-point format is translated to more cost-effective and faster hardware. With fixed-point representation, all data manipulations such as addition and multiplication can be reduced to more simple (but faster) arithmetic functions such as binary add and shift.

As our scenario has a predictable and relatively small data set, we can properly specify our fixed-point scheme with no disadvantage regarding the range. Furthermore, a fixed-point representation with two's complement (2C) numbering format is used for the reasons that follow. First, it is popular in DSP implementations [12], as it has a unique representation for zero and the summation/subtraction operations are consistently performed using the same hardware [13]. Second, it is recommended by Xilinx when using DSP48E1, an inherit FPGA primitive for math operations [14]. Finally, the ADS62P49 implements the 2C format directly in its digital outputs [15].

Our fixed-point representation parameters are shaped by the characteristics of the ADC. We first need to generate the signals obeying the analog input voltage range. Next, we have to create the 2C scaled version of the signal according to the channel resolution, following the format of the expected captured signals at the output of the ADC channels. Finally, we address overflow conditions and re-scaling of data during some arithmetic operations.

5.1 Signal Characteristics

The amplitude of the analog inputs of the ADC must assume values between -1 and 1 V; thus the test signals are scaled to have a dynamic range of 2 Vpp (Volts peak-to-peak) referenced

to 0 V. The waveforms are actually created separately with respect to each dominant band to match the dual-channel architecture mentioned in section 2.2. At this point, we have two real-valued discrete arrays $\boldsymbol{s}_1^{\mathbb{R}}[t]$ and $\boldsymbol{s}_2^{\mathbb{R}}[t]$ that can be loaded into the VSG to serve as IF analog inputs for the ADC. Following the derivations in Chapter 3, here again we make use of bold face letters to represent discrete vector signals and normal face to indicate a scalar sample.

5.2 Fixed-Point Encoding

The dynamic range for signed integers in 2C format is given by $[-2^{(N-1)}, 2^{(N-1)} - 1]$, where N is the number of bits in the binary word. Additionally, a *N-bit* word provides a resolution of 2^N. Equation 5.1 generalizes the 2C encoding of an arbitrary *N-bit* signed integer s, where i represents the bit index going from 0 to $N-1$ and s_i denotes the i_{th} bit of s,

$$s^{2C} = \begin{cases} \sum_{i=0}^{N-2} s_i \times 2^i & \text{if } s \geq 0 \\ -2^{N-1} + \sum_{i=0}^{N-2} s_i \times 2^i & \text{if } s < 0 \end{cases}. \tag{5.1}$$

The word size of our ADC is $N = 14$; therefore, the analog test signal is encoded in the decimal range $[-8192_D, +8191_D]$ with 16384_D quantization levels, where the subscript D is used to emphasize the decimal (base-10) format. Moreover, each bit of the digital word permits a minimum voltage step of $2/2^N = (2/16384)\,V$. As our real numbers are comprised in the -1.0 to $+1.0$ range, we are allowed to position the decimal point right after the sign bit. In other words, the integer part is implied to be 0 and the remaining 13 digits represent the fraction. This can be denoted "Q0.13" number format as stated by [13]. Therefore, to obtain the Q0.13 format of a discrete-time real-valued signal $\boldsymbol{s}^{\mathbb{R}}[t]$, we simply use (5.1) with s being each individual sample of $\boldsymbol{s}^{\mathbb{R}}[t]$. Performing the sum for $N = 14$ and simplifying by using decimal representation, we have

$$s^{\mathbb{Z}} = \begin{cases} \lfloor s^{\mathbb{R}}[t] \times (8191)_D \rfloor & \text{if } s^{\mathbb{R}}[t] \geq 0 \\ \lfloor s^{\mathbb{R}}[t] \times (8192)_D \rfloor & \text{if } s^{\mathbb{R}}[t] < 0 \end{cases}, \tag{5.2}$$

where $s^{\mathbb{Z}}$ represents each sample of the Q0.13 signal array and $\lfloor \cdot \rfloor$ is the floor operator.

Figure 5.1 shows an example of a real-valued signal encoded in this format.

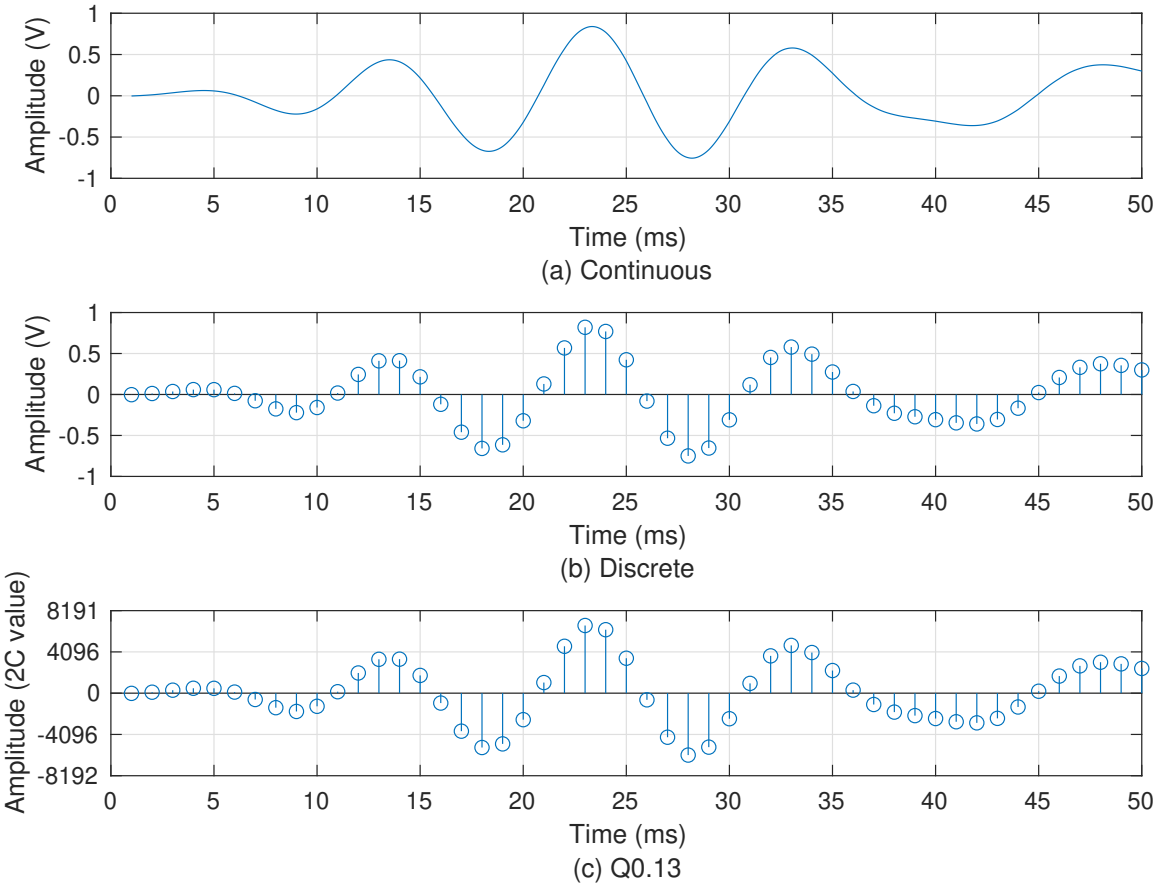


Figure 5.1. Example of an arbitrary real-valued signal encoded in the Q0.13 format. (a) Continuous, (b) Discrete and (c) Q0.13

5.3 Arithmetic and Overflow Control

The 2C format is consistent with binary arithmetic [13], but integer overflow can always occur. The addition of two M-*bit* numbers might produce a sum with length $M + 1$ while a multiplication can potentially generate a $2 \times M$ product. Although a 28-*bit* register is enough for our scenario, we chose a 32-*bit* word, which is a common base-2 standard in computer architecture. Furthermore, we sign-extend the outputs of the ADC from 14 to 32 bits to maintain coherence in our HDL modules.

To help prevent overflow from repeated mathematical computations (e.g., iteration loops, multiply-accumulate operation (MAC)), each intermediate product in the down-conversion module (see Figure 2.2) is right-shifted by the factor $N-1$. This comes from the fact that any number x multiplied by a unit sinusoidal stays in the range $[-x, x]$. In our specific case, if we multiply any sample denoted by (5.2) by a sinusoidal wave, the resulting product must lie in the interval shown $[-8192, 8191]$. Table 5.1 exemplifies this procedure with three multiplications of an arbitrary sample by a given sinusoidal value.

Table 5.1. Examples of re-scaling after a multiplication. The Q0.13 format uses an intermediate product prior to re-scale by the N-1 factor

	sample	**sin value**	**intermediate**	**final result**
Real	1	1	1	1
Q0.13	8191	8191	67092481	8191
Real	1	-0.5	-0.5	-0.5
Q0.13	8191	-4096	-33550336	-4096
Real	-0.35	-0.875	-0.30625	-0.30625
Q0.13	-2868	-7168	20557824	2509

It is worth pointing that, as we are working with signed numbers and the sign bit must be taken in consideration, the re-scaling is made using the Verilog operator **>>>**, which applies an arithmetic shift [16]. Nonetheless some accuracy is lost in this procedure, but we assume that the error is small enough to be ignored at this point.

5.4 Validation of the Fixed-Point Format for MLRS

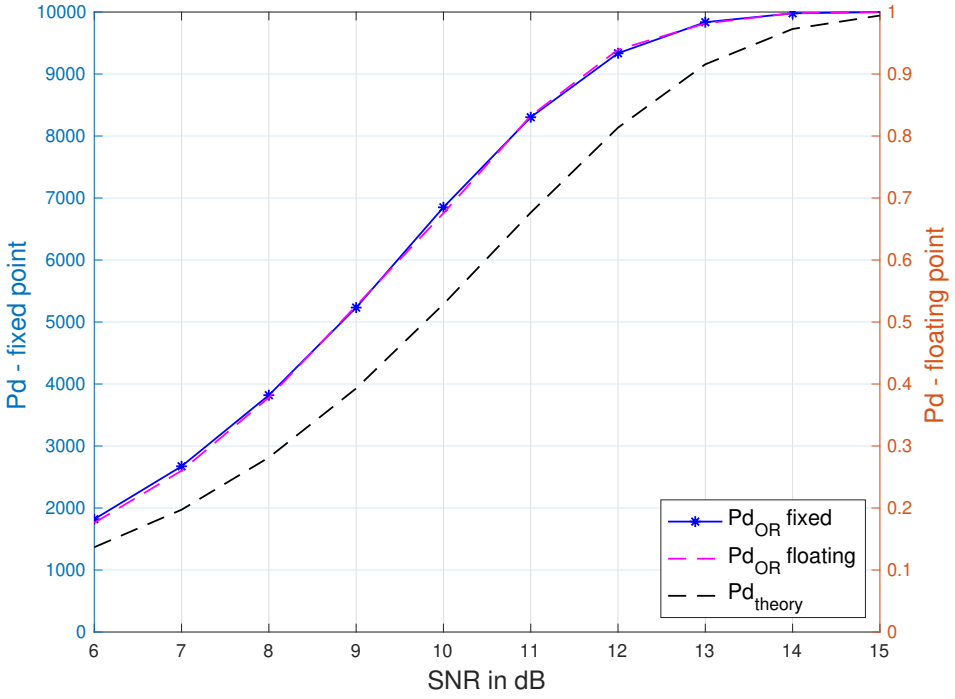
Before moving to the FPGA implementation and to evaluate the behavior of the number representation proposed here with the MLRS detector shown in Chapter 3, we recreate the simulations shown in Section 3.3 using only fixed-point operations. This can be done in MATLAB by first encoding signals into the fixed-point format using the function *uencode*

$$\boldsymbol{s}_{INT} = uencode(\boldsymbol{s}_{FLOAT}, N, r, \text{“}signed\text{”}); \tag{5.3}$$

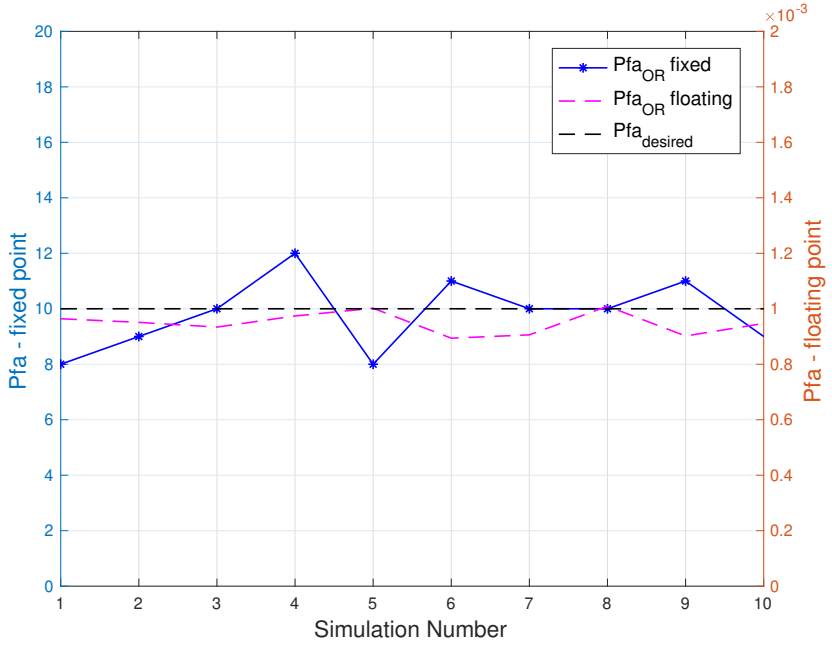
where $\boldsymbol{s}_{FLOAT}$ is a real valued vector signal with samples in the $\pm r$ range, N is the number of bits in the fixed-point representation and the keyword *signed* accounts for the 2C format. Next, given that the default data type for many MATLAB functions is *double*, we also make use of the functions *cast* and *int32* to convert between data types. Furthermore, as we intend to implement this same simulation in the FPGA using HDL, we assume that only the integer division is available. Because any probability results in a value between 0 and 1, we also have to encode the P_d and P_{fa} during the *Monte Carlo* trials using a fixed-point format. The simplest way to achieve this is to apply the equation

$$P_{d_{INT}} = \frac{(total\ number\ of\ detections) \times S}{R}, \tag{5.4}$$

where R is the total number of runs (or trials) and S an integer scaling factor, which forces the result of the division to be between 0 and S. Note that S also determines the precision in the result of (5.4), in which the maximum resolution is when $S = R$. To simplify our implementation, we let S and R to be powers of 10, thus the resulting probability can be read directly from the FPGA registers. Figure 5.2 compares the results using fixed-point encoding with $N = 14$ and $S = 10^4$ to the default floating-point method. Thus, if for a given SNR the detection probability in the fixed-point simulation is $Pd_{FIXED} = 9981$, the true percentage value is 99.81%.



(a) P_d



(b) P_{fa}

Figure 5.2. Fixed-point simulation performance results of composite detector for $\kappa = 1.09$

Here, we have set $R = 1 \times 10^5$ in preparation for the hardware simulation. Given that we need to recreate an IF analog version of the signal plus noise stream using the VSG, a simulation with a greater number of trials could be demanding in terms of file size and complexity.

As shown in Figure 5.2, the differences between the two methods are very subtle and they can be explained by the errors in the quantization process introduced by the *uencode* function and the rounding of the integer division operations.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 6:
A Verilog Hardware Description Language MLRS Receiver

With the intent to show the particular aspects of a hardware implementation, this chapter describes the full development of our version for the MLRS detector using Verilog HDL in the *Xilinx Vivado* environment. Following the two-channel architecture suggested in Section 2.2, the HDL project can be divided in two main parts based on their functionality. The first part is the configuration of the external ICs and the receiving of the digitized data, which are performed by the *FMC* block shown in Figure 6.1. The second part is the processing of the digital signals and calculation of detection results, which is realized by the *MLRS* block.

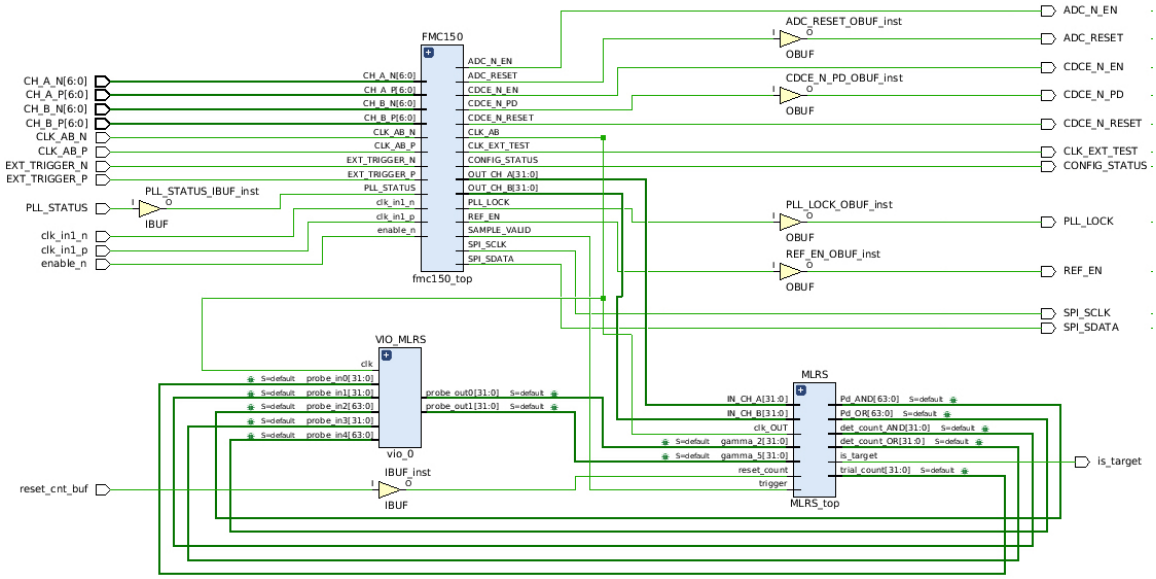


Figure 6.1. HDL top level schematic: FMC module receives the signal from the external board. MLRS module processes data and computes the detection results

Moreover, a virtual I/O (VIO) module is added to the top level project to serve as an auxiliary interface (*VIO_MLRS* block). Through this module we can input configuration parameters, visualize and capture results in real time.

6.1 Configuration of the FMC and Capturing of the Digital Signal

Before the digital signals are ready in the output of the capture block, it is necessary to configure the logic functions inside the FMC card using a SPI protocol [15] [17]. The FPGA needs to configure each module sequentially since the SPI connection lines in the FMC interface are shared among all the external ICs. As the system initializes, the first configuration required is in the clock management system (the TI's CDCE). In this step, parameters such as the sampling frequency, clock reference and signaling modes are read from a block memory in the FPGA and communicated to the CDCE via the SPI protocol. After the configuration is done, the clock status is verified by monitoring a feedback line: the PLL lock indicator. At this point, we have the clock running on the external board, and the configuration of the ADC circuits can be started.

Similarly to the previous procedure, the initial parameters for the ADC (data format, power mode, offset correction, etc.) are stored in a programmable read-only memory (PROM) as coefficient files. The FPGA acts as an SPI leader, transferring the desired parameters to their respective register in the ADC side. When the SPI state machine transmits the last register in the block memory, the configuration is over and the digital outputs are ready to be captured in the DDR bus of the FMC connector. Figure 6.2 shows a flowchart of the configuration procedure.

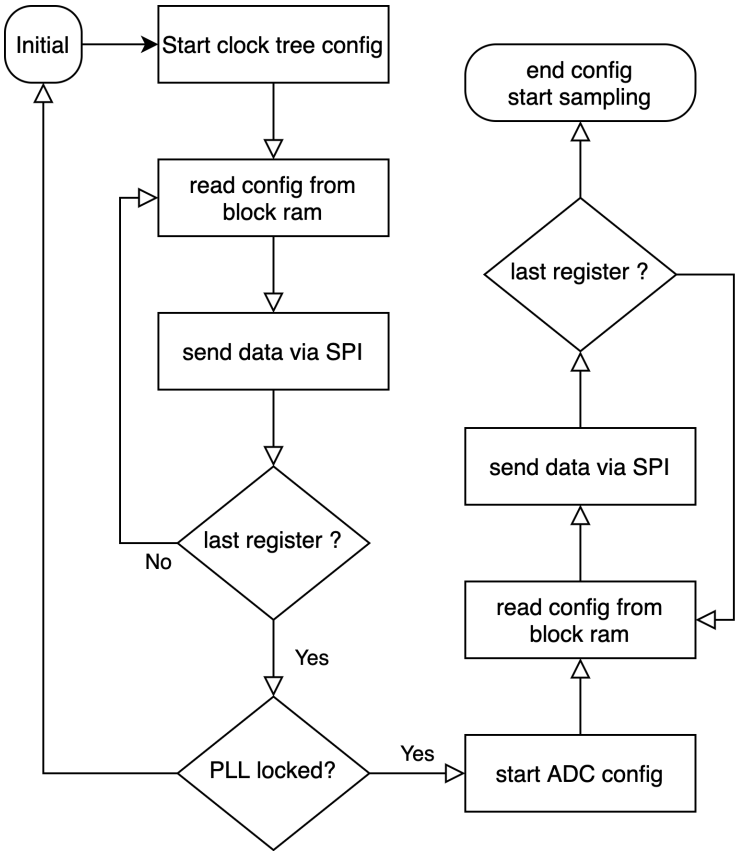


Figure 6.2. Flowchart of external board configuration

Although the CDCE and the ADC have different configuration characteristics, like register size and data polarization, the SPI Verilog module is developed with parametric inputs, allowing for the reuse of the hardware model based on different instantiation parameters. The finite state machine controlling the SPI protocol is implemented using a modified version of the hardware description suggested by [18]. The block memory and clock functions used in the SPI modules are implemented by instantiation of customized Xilinx IP blocks.

The communication between the ADC outputs and the FMC interface is performed through a DDR interface with LVDS standard. In our system, this capacity is implemented by using two dedicated Xilinx built-in primitives, the *IBUFDS* and the *IDDR* [19]. First, each differential pair of the ADC outputs must terminate in a LVDS buffer (the IBUFDS primitive) in the FPGA side, which will convert the differential signal to the single-ended standard used in the internal routing. Second, the outputs of the IBUFDS are de-multiplexed

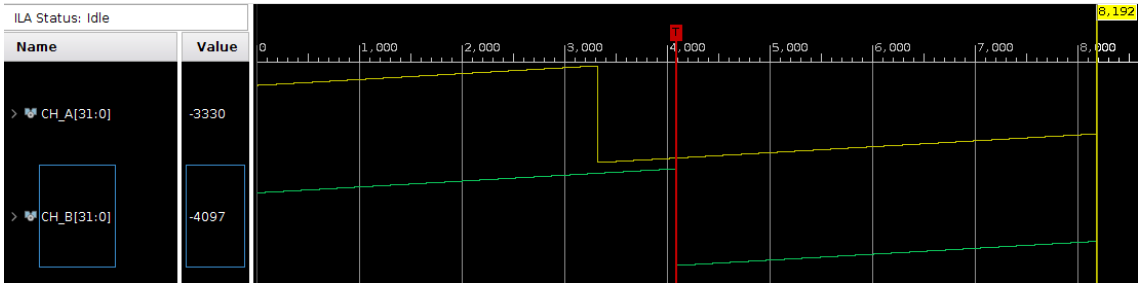
by passing through the IDDR primitive, which will alternately capture the even and odd bit positions of the digitized word in the rising and falling edges of the clock, respectively [15]. This process is executed synchronously with the CDCE clock, which is already configured during the initial stages. A total of 14 bits per channel are then available to the FPGA logic at the end of each clock cycle. To avoid code repetition and have a clean construction of the DDR capture module, we make use of the Verilog *generate* capability [20].

Finally, for each IDDR bus, the 14-bit word is extended to 32-$bits$. As the sign bit must be preserved, this can be done by a conjunction of the *replication* and *concatenation* Verilog operands [21],

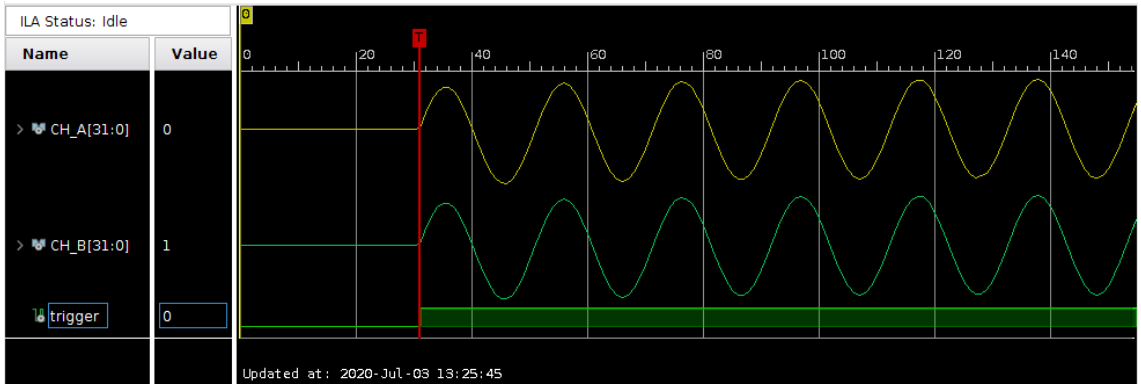
$$size32bit[31:0] = \{\{18\{size14bit[13]\}\}, size14bit[13:0]\}; \tag{6.1}$$

where the most significant bit (MSB) is replicated 18 times and concatenated with the original 14-bit word.

To isolate the complexity of configuring and capturing the signals through the FMC components from the actual MLRS processing, here we define a checkpoint in our HDL project to validate the ongoing design in two forms. First, we use the *test pattern* functionality provided by the ADC. With this mode, instead of converting the analog inputs, the ADC circuits will simply output a digital pattern (e.g., ramp or sawtooth signal) in the DDR interface. The second form is through inputting a *sinusoidal* originated by a VSG in both analog channels. In this test, the ADC operates in normal mode and the entire analog-to-digital path is validated. Figure 6.3 shows a snapshot of both test scenarios, in which the signals are captured and verified using Vivado's integrated logic analyzer (ILA). Although the logic analyzer is configured to output an analog style waveform, the column *Value* shows the respective fixed-point discrete value.



(a) Digital Ramp generated by the ADC



(b) Sinusoidal generated by the VSG

Figure 6.3. Validation of signal digitization

6.2 MLRS Processing and Detection

Once the analog-to-digital conversion is performed by the capturing block, the digital dual-channel stream must be processed simultaneously by the next block. According to what is proposed by the MLRS technique and before applying the detection tests, we need to perform the matched filtering in the baseband components of each dominant frequency component (see Figure 2.2).

6.2.1 Downconversion

To derive the in-phase and quadrature portions for each received channel, we develop a simple DDC using a look-up table (LUT). As the range of the sine and cosine waves conveniently match the dynamic range of our input signals, we apply the same floating-point to fixed-point encoding detailed in Subsection 5.2. Even though in this first version we

assume an IF of 2 MHz, our DDC LUT stores a 1 MHz cosine as shown in Table 6.1. With this LUT organization, we allow for some flexibility in the IF parameter as any multiple of 1 MHz can be used, as long as we keep a constant sampling frequency. Running at 20 MHz (sampling clock rate) the cosine values repeat every 20 clock cycles; therefore, our LUT needs to have only 20 distinct values.

To get the in-phase component, we multiply the channel by a 2 MHz *cosine*. Using the 1 MHz LUT and instead of stepping through each value of the table sequentially, we simply read every other value of the array with the sinusoidal period repeating every 10 clock cycles. In other words, in the same clock period but with a step size of 2, the phase change for the 2 MHz wave is twice that of the 1 MHz wave. Thus, for example, if we were to tune to an IF of 5 MHz, we would have to increase the step size to 5.

Next, we need to obtain the quadrature component of the channel. With an identical step size of 2 for the chosen IF, we can use the same table to generate a *negative sine*. However, to introduce the 90-degree shift, the initial phase must be half cycle apart from that of the cosine. Therefore, for the quadrature component, we start reading the table in its 6th value (see 3rd column of Table 6.1). A similar approach can be applied when using another IF.

Table 6.1. Derivation of cosine and negative sine from a single LUT

sample #	LUT	cos	-sin
1	8191	8191	0
2	7791	6627	4816
3	6627	2531	7792
4	4815	-2532	7792
5	2531	-6628	4816
6	0	-8192	1
7	-2532	-6628	-4815
8	-4816	-2532	-7791
9	-6628	2531	-7791
10	-7792	6627	-4815
11	-8192	repeat ...	repeat ...
12	-7792		
13	-6628		
14	-4816		
15	-2532		
16	-1		
17	2531		
18	4815		
19	6627		
20	7791		

It is noted that using a single LUT allows re-utilization of hardware modules, saving resources and simplifying the design.

6.2.2 Reference Signals

In view of the MF, we need access to the reference signals used in the correlation process. The coefficients are generated in MATLAB from the known signals, obeying the number format described in Section 5.2 and saved in a *coefficients file*. During the HDL design flow,

a *block memory generator* tool is used to read the reference files and store each of them in an FPGA random-access memory (RAM). Synchronously with the sampling clock, the addresses of the block memory are read sequentially, making the coefficient values available as outputs of the memory module. In addition, the size of the individual block and the end of the sequential reading are controlled by the number of samples in one signal length.

6.2.3 Matched Filter

Following the derivation shown in section 3.1, for a discrete real signal $\boldsymbol{x}$ with N samples in normalized sampling time, the output of the MF can be defined as

$$MF_{out}(\boldsymbol{x}) \equiv y[N-1] = \sum_{k=0}^{N-1} s[k]x[k], \tag{6.2}$$

where $\boldsymbol{s}$ is the filter coefficient array, which in our implementation is known as the reference signal. This is clearly a simplified version of the discrete convolution process where we are interested only in the filter output at the time instant $N-1$, the middle point of the entire convolution result. Furthermore, from the same definition, $\boldsymbol{s}$ turns out to be the noise-free version of $\boldsymbol{x}$.

As we are proposing to work with the in-phase and quadrature components in a dual-channel MLRS receiver (see Figure 2.2), our detector will actually utilize two MF outputs, as presented in Chapter 3. Regarding the hardware implementation of real signals, we need to define the two quantities $MF_{out}(\boldsymbol{A})$ and $MF_{out}(\boldsymbol{B})$ as the MF outputs of our two dominant signals $\boldsymbol{A}$ and $\boldsymbol{B}$:

$$\begin{aligned} MF_{out}(\boldsymbol{A}) &= MF_{out}(\boldsymbol{A_I}) + MF_{out}(\boldsymbol{A_Q}) \\ MF_{out}(\boldsymbol{B}) &= MF_{out}(\boldsymbol{B_I}) + MF_{out}(\boldsymbol{B_Q}) \end{aligned}, \tag{6.3}$$

which is nothing more than (6.2) expanded for the baseband representation of two different signals. In other words, to obtain the MF output without reconstructing the signal, we propose to add the MF outputs of its baseband components.

Another task in our implementation of the MF is to find the starting point of the received

signal. Particularly, we need to define the sample with index $k = 0$ for $\boldsymbol{x}$ in (6.2). This is performed in our Verilog modules by monitoring a *trigger signal* sent by the VSG. In other words, we determine the starting point of signals $\boldsymbol{A}$ and $\boldsymbol{B}$ as the sample obtained from the ADC at the clock cycle where the trigger signal from the waveform generator goes high (logic level *true*). Although a synchronization technique can be used for the radar receiver, signal synchronization is not part of our implementation scope.

To summarize the above concepts in the FPGA implementation, we have four simultaneous MAC operations: one for each baseband component of both channels. Other than the sampling clock, the process is synchronized with the help of an external trigger, outputting the respective MF results at every realization (end point) of the received signals.

6.2.4 Detection Test and Statistics

As discussed in Chapter 3, having two simultaneous processing channels yields a variety of possibilities in how we can perform the detection test. With respect to the hardware implementation, given that we have two MF outputs, we simply need to verify if their values exceed an arbitrary threshold. As this procedure does not depend on the clock and the synchronization of the signal has already taken place in the previous steps, we simply apply a Verilog *conditional assignment* in the outputs of the MF, comparing the current value with the desired threshold, which in turn is configured via the VIO interface. Although our Verilog implementation is focused in the *OR detector* shown in Section 3.2, we allow flexibility in future experiments by implementing two different tests given by

$$\begin{aligned} \boldsymbol{assign} \quad & T_{AND} = (MF_{out}(\boldsymbol{A}) \geq \gamma'_A) \,\&\, (MF_{out}(\boldsymbol{B}) \geq \gamma'_B) \,?\, 1'b1 \,:\, 1'b0 \\ \boldsymbol{assign} \quad & T_{OR} = (MF_{out}(\boldsymbol{A}) \geq \gamma'_A) \mid (MF_{out}(\boldsymbol{B}) \geq \gamma'_B) \,?\, 1'b1 \,:\, 1'b0 \end{aligned}, \tag{6.4}$$

where γ'_A and γ'_B are obtained by using (3.5) by applying the energies of the dominant signals A and B, respectively. T_{AND} and T_{OR} are the variables holding the logical result of each regular expression for the current signal realization. Moreover, to utilize the modified threshold defined in Section 3.3.1, we simply set

$$\gamma'' = \gamma'_A = \gamma'_B. \tag{6.5}$$

Completing the assembly of our HDL-based MLRS detector, the P_d (and eventually the P_{fa} for a noise only test) is obtained by accumulating the outcome of T in a given number of runs. The trial count is controlled by the number of times the detector has reached the end of the reference signal array while performing the MF. Assuming that only the fixed-point division is available for our implementation, as detailed in Section 5.4, we scale up the value of T by an integer factor to have a percentage result with two digits precision after the decimal point.

6.3 Debug and Simulation Interfaces

To validate our hardware project, we perform debug tasks and collect simulation results. The following procedures are used.

6.3.1 Behavioral Simulation

The *behavioral simulation* is widely used in digital designs to verify if the register-transfer level (RTL) code is functioning as expected. This is the first validation of the Verilog project and it is performed prior to synthesis and implementation steps. To simulate the analog channels of our system, the 2C signals used in the MATLAB scripts are converted to the binary format and saved in external files. Therefore, in the Verilog test bench environment, the system functions *$fopen* and *$fscanf* can be applied to open and read the files respectively. In summary, the *$fscanf* steps sequentially through the external file and outputs one binary sample at every simulation clock cycle.

Next, the binary sample is manipulated in the test bench program to reproduce the DDR output of the ADC and create the adequate LVDS signaling to be received by the FPGA buffers. In the *positive edge* of the clock, the *even bits* of the binary sample are presented to the positive pair of the DDR channel while the *complement* of the even bits are presented to the negative pair. Conversely, in the *negative edge*, the *odd bits* of the data and their complement are copied to the DDR positive and negative differential pairs respectively.

Other signals like clocks, triggers, and constants are simulated directly in the test bench program. Figure 6.4 shows a snapshot of the behavioral simulation environment. Although only one channel and a few variables are shown here, any digital element created in the RTL design can be accessed and verified at a given simulation time instant.

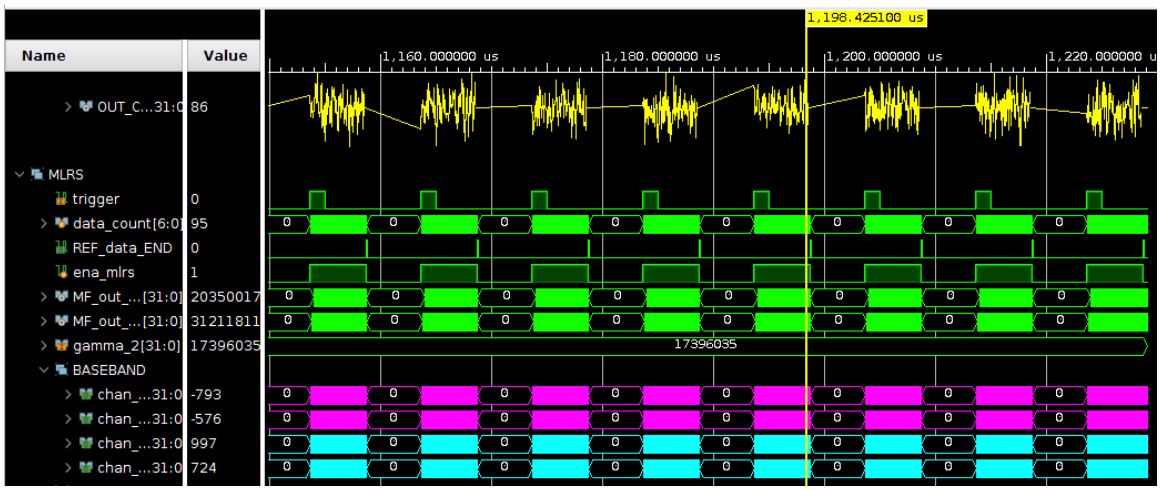


Figure 6.4. Behavioral simulation environment

6.3.2 ILA And VIO Interfaces

After passing the behavioral validation and having the design running in the FPGA, the integrated logic analyzer (ILA) and virtual I/O (VIO) modules were inserted into the project. As the ILA is more demanding in terms of hardware resources, it is employed only to validate signals and debug specific problems that do not exist in the behavioral simulation, particularly in the ADC details and the triggering process that now use external signals. On the other hand, the VIO works as an interface, allowing the configuration of the detection threshold and the acquisition of the probability results for a given SNR level. Figure 6.5 shows a simulation environment with both modules, where the ILA displays the output of the ADC channels and the VIO shows the simulation parameter and result statistic.

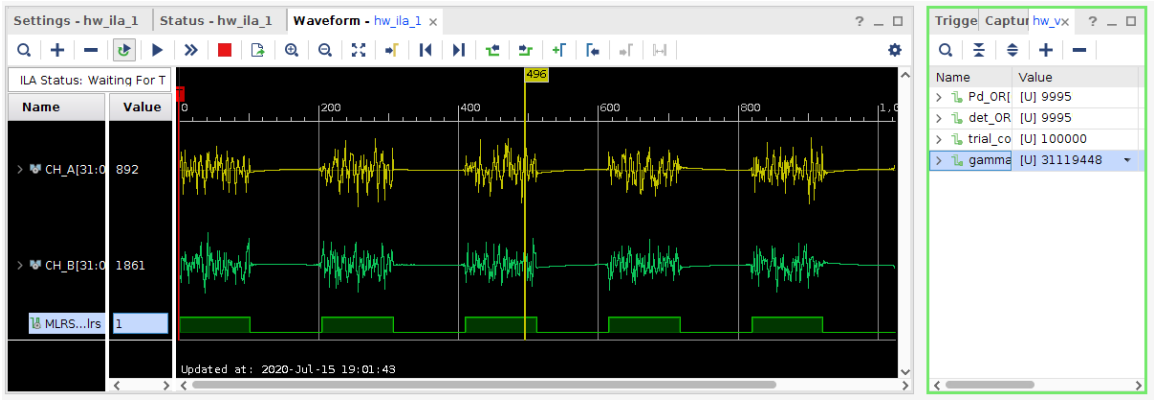


Figure 6.5. ILA (left) and VIO (right) interfaces

It is worth mentioning that there exist other important steps between the behavioral simulation and the synthesis/implementation process, like the constraining of the I/O pins and the timing closure analysis. These details are pertinent to any FPGA design but not necessarily unique to the MLRS technique and as such will not be discussed further here.

6.4 Hardware Utilization

Having a baseline for the hardware cost of our HDL implementation is important. In Table 6.2 we show a breakdown of the FPGA resources utilized by their logic function. This information is extracted from the design tool after synthesis and implementation.

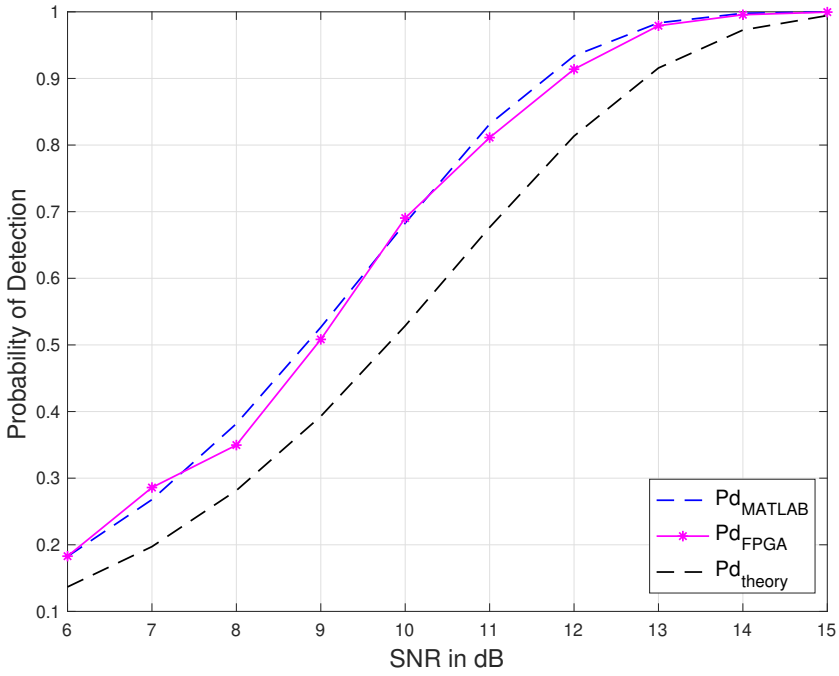
Table 6.2. Hardware utilization report

Resource	**Available**	**Utilization**	**Utilization %**
LUT (as logic)	218600	5892	2.70
LUT (as RAM)	70400	206	0.29
Flip-Flops	437200	4054	0.93
RAM blocks	545	35	6.42
DSP48 blocks	900	24	2.67
IO pins	362	49	13.54
Clock Management	8	1	12.5

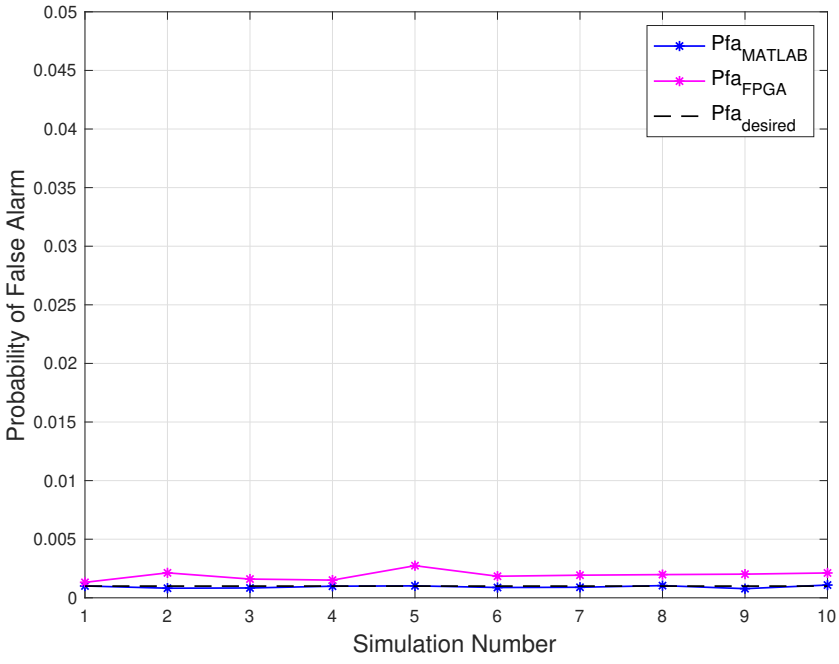
Even with extra logic from the debug and interface blocks, the design has a low utilization rate relative to the specific FPGA model. The overall usage among all instances types is approximately 1.4%, which is reasonable considering our proposal for a simplified receiver with low computational cost.

6.5 HDL Detector Performance Results

The simulations from Section 3.2 are recreated in the FPGA environment using the following scenario. Each dominant band component is created in MATLAB and the respective noise level is added according to the SNR. Next, each channel is processed by a special function provided by the VSG vendor, transforming the digital stream in the appropriate format used in the generator. The generator in turn sends each component in separate IF analog channels that are sampled by the ADC (see block diagram in Figure 2.2). The digital signal is then processed as detailed in Section 6.2. Figure 6.6 shows the results for the P_d and P_{fa} compared to the MATLAB floating point simulations. The κ multiplier for the modified threshold is maintained as 1.09 and the number of trials is 1×10^5.



(a) P_d



(b) P_{fa}

Figure 6.6. Hardware detector performance

While the results for the P_d in the hardware implementation look very similar to the MATLAB scenario, the P_{fa} shows a slight increase. This can be explained mostly by the finite quantization process in the ADC channels. Other factors that may contribute to these errors are the added thermal noise present in the analog environment, the conversion path from MATLAB to the analog signal output by the VSG, and the loss in the dynamic range introduced by the scaling of the signal energy in the SNR simulations.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 7:
Conclusion

To conclude this work, we summarize the results of our research and suggest future topics that can be explored in the MLRS field.

7.1 Summary of Results

By developing a hardware model for a dual-channel detector, we demonstrated that the MLRS technique can be implemented using COTS ADCs and FPGAs. The use of a fixed-point format allowed for the optimization of the hardware resources, leading to a compact design. The hardware was entirely described in the Verilog language using the FPGA development environment with no dependency on third party tools for code generation, which permitted full control of the low-level primitives and allowed the debug of the code throughout the entire HDL project.

Depending on the target response signals used, the two-channel MLRS composite detector was able to increase the P_d compared to a traditional MF detector. The concept of a scaled threshold value was devised to compensate for the increased P_{fa} introduced by the extra decision layer on each dominant band detector. Furthermore, the low sampling rate and the use of multiple MFs reduced the computational cost of the detection process.

Overall, it had been shown that a low-cost MLRS detector is feasible. Furthermore, detection performance is increased depending on target frequency response used.

7.2 Future Work

Due to the characteristics of the ADC board used in our implementation, we utilize two dominant sub-carriers centered at the same IF. This condition introduces complexity to the pre-processing block which has to downconvert the coupled waveforms using different parameters for each dominant portion. An alternative approach using distinct center frequencies can be explored, transferring the complexity of dealing with multiple center frequencies to the FPGA receiver. This requires the sampling rate of each ADC channel to

be controlled independently, reflecting a more flexible scenario. Although the IF utilized in the analog test is relatively low for commercial radar environment, there is no restriction for the detector to work with higher IFs, provided that the sampling rate is adjusted accordingly. Further explorations can be in the ADC configuration and DDR capture in order to allow higher frequencies and even real time adjustments of this parameter.

The analog front end can be adjusted by creating the specific RF front-end hardware for splitting of the dominant bands. In this work, the first downconversion stages were performed in MATLAB. Furthermore, field experiments with actual targets can also be performed.

The hardware available to us had only two digitizing channels, however the detector developed in our project can be easily expanded. Although the complexity of the hardware increases with the additional channels, the utilization report indicates that current FPGA devices can support a significant increase of the implemented hardware. An HDL generic framework for the digitization and the use of multiple MF channels in the detection process is already in place. Thus, future implementations can make use of our parametrized modules combined with Verilog special constructions to expand the detector to more than two channels. Additionally, alternative configurations to the binary decision of the composite detector must be proposed in order to access the tradeoff between P_d and P_{fa}.

Our approach to define the κ scaling factor that would recover the P_{fa} was empirical, based on the Monte Carlo trials. A more detailed evaluation of the changes introduced by the *OR detector* and its mathematical effects in the detection theory can possibly be made. This assessment could help us to define an optimal value of κ with respect of the signal energy or SNR.

Although the hardware implemented is this work has used the MF approach for the detection problem, a variety of adaptive filters in DSP theory can potentially take advantage of the proposed channelized processing, further evaluating the MLRS method for other RF applications.

List of References

[1] M. A. Johnson, "Signal recovery and detection of certain wideband signals using multiple low-rate ADC," M.S. thesis, NPS, Dept. of Elec. and Comp. Eng., Monterey, CA, USA, 2018 [Online]. Available: http://hdl.handle.net/10945/60415

[2] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[3] G. L. Fudge, R. E. Bland, M. A. Chivers, S. Ravindran, J. Haupt and P. E. Pace, "A Nyquist folding analog-to-information receiver," pp. 541–545, 2008.

[4] M. Johnson and R. A. Romero, "Signal recovery and detection of certain wideband signals using multiple low-rate ADCs," in *2019 Aerospace Conference*, Big Sky, MT, USA, 2019, pp. 1–9.

[5] A. Berger, "Low bandwidth matched-illumination radar transmitter-receiver design for certain wideband targets," M.S. thesis, NPS, Dept. of Elec. and Comp. Eng., Monterey, CA, USA, 2019.

[6] R. A. Romero, "Matched waveform design and adaptive beamsteering in cognitive radar applications," Ph.D. dissertation, Univ. of Arizona, Dept. of Elec. and Comp. Eng., Tucson, AZ, USA, 2010.

[7] Q. J. O. Tan and R. A. Romero, "Ground vehicle target signature identification with cognitive automotive radar using 24–25 and 76–77 GHz bands," *IET Radar, Sonar Navigation*, vol. 12, no. 12, pp. 1448–1465, 2018.

[8] S. M. Kay, *Fundamentals of Statistical Signal Processing Vol. II - Detection Theory*. Upper Saddle River, NJ, USA: Prentice Hall, 1998.

[9] H. L. Van Trees, K. L. Bell, and Z. Tian, *Detection, Estimation, and Modulation Theory Part I: Detection, Estimation, and Filtering Theory*. Hoboken, NJ, USA: Wiley, 2013.

[10] *ZC706 Evaluation Board for the Zynq-7000 XC7Z045 SoC User Guide*, Xilinx. Co., San Jose, CA, USA, 2019.

[11] *FMC150 User Manual*, Huntsville, AL, USA: Abaco Systems, 2013.

[12] U. Meyer-Baese, *Digital Signal Processing With Field Programmable Gate Arrays*, 4th ed. Berlin, Germany: Springer-Verlag, 2014.

[13] W. T. Padgett and D. V. Anderson, *Fixed-Point Signal Processing*. San Rafael, CA, USA: Morgan & Claypol, 2009.

[14] *7 Series DSP48E1 Slice User Guide*, San Jose, CA, USA: Xilinx. Co., 2018.

[15] *Dual Channel 14-/12-Bit, 250-/210-MSPS ADC With DDR LVDS and Parallel CMOS Outputs Datasheet*, Dallas, TX, USA: Texas Instruments Inc., 2019.

[16] P. P. Chu, *FPGA Prototyping by Verilog Examples: Xilinx Spartan-3 Version*. Hoboken, NJ, USA: Wiley, 2008.

[17] *Ten Output High Performance Clock Synchronizer, Jitter Cleaner, and Clock Distributor Datasheet*, Dallas, TX, USA: Texas Instruments Inc., 2020.

[18] C. Unsalan and B. Tar, *Digital System Design with FPGA: Implementation using Verilog and VHDL*. New York, NY, USA: McGraw-Hill, 2017.

[19] *7 Series FPGAs Select IO Resources User Guide*, San Jose, CA, USA: Xilinx. Co., 2018.

[20] S. Brown and Z. Vranesic, *Fundamentals of Digital Logic with Verilog Design*, 3rd ed. New York, NY, USA: McGraw-Hill, 2014.

[21] J. Cavanagh, *Verilog HDL Design Examples*, Boca Raton, FL, USA: CRC Press, 2017.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California