A Low Complexity Cyclostationary Detector for OFDM Signals

Douglas Allan*, Louise Crockett and Robert W. Stewart
Department of Electronic and Electrical Engineering, University of Strathclyde
204 George Street, Glasgow, G1 1XW, Scotland, UK
Email: *d.allan@strath.ac.uk

***Abstract*—One of the key challenges for state of the art radio systems is enabling efficient utilisation of the Radio Frequency (RF) spectrum. Licensed frequency bands are often under-utilised in both time and geographical location and thus the opportunity exists for secondary users to transmit in these bands, provided that they do not interfere significantly with the operation of the primary licensed user. A proposed method for exploiting this opportunity is Cognitive Radio (CR) wherein the secondary user is able to modify its transmissions based on observation of the operating RF environment. Orthogonal Frequency Division Multiplexing (OFDM) is the enabling technology for many modern communications standards such as IEEE 802.11a (WiFi) and 4G Long Term Evolution (LTE). Therefore, facilitating robust and cost effective detection of OFDM signals is a key problem for the design of secondary user CR systems. In this paper, we derive and assess the performance of a low complexity detection scheme that exploits the inherent cyclostationarity of OFDM signals. We then present details of its implementation on a Xilinx Artix 7 FPGA and compare the resource cost of the proposed detector with another low complexity detection algorithm found in the literature.**

I. Introduction

OFDM is a multi-carrier modulation method that divides the available spectrum into a number of parallel narrowband sub-carriers. It is noted for its robust performance in multipath channels and the fact that modulation and demodulation can be performed efficiently in hardware through the use of the Fast Fourier Transform (FFT) algorithm. Due to these benefits, it has been widely adopted as part of modern communications standards such as LTE and the IEEE 802.11 family of standards. A common feature for all of these systems is a Cyclic Prefix (CP), that consists of a portion of the OFDM symbol which is copied from the end of the symbol and appended at the front of the symbol. This provides a level of immunity to Inter Block Interference (IBI) between OFDM symbols and significantly simplifies the process of channel equalisation in the receiver.

The CP is also a beneficial feature for the detection of OFDM signals in the sense that it makes the OFDM signal cyclcostationary. A signal is said to be wide-sense cyclostationary if its statistics are periodic with some fundamental period. In the case of OFDM, due to the presence of a CP, the autocorrelation function is periodic. Therefore, OFDM signals can be detected by testing for the presence of cyclostationarity. The authors in [1] presented an algorithm for detection of cyclostationarity using a Likelihood Ratio Test (LRT). This has since been modified and implemented using FPGAs as described by the authors in [2]. Similarly, in a previous paper we described an implementation of the same algorithm and showed how its hardware cost could be reduced by re-arranging the calculation to avoid division [3]. In [4], the authors describe a sub-optimal detection method that calculates the ratio of the squared magnitude of the Cyclic Autocorrelation Function (CAF) computed at a known cyclic frequency to the squared magnitude of the CAF calculated at a known non-cyclic frequency. The performance of this detection scheme is reduced when compared to the LRT algorithm but it uses a significantly simplified test statistic. Similarly, in [5], the authors suggest a detector which applies the spatial sign function to the input data, also leading to a simpler test statistic.

In this paper, we introduce a low complexity cyclostationary detection algorithm that is shown to perform well in comparison to the detectors in [3], [4] and [5]. We then implement the proposed detector on a Xilinx Artix 7 FPGA device and compare its resource cost and performance to an implementation of the detector in [4]. Therefore, our contribution is an addition to existing approaches to cyclostationary detection of OFDM signals that can be achieved at low cost in terms of FPGA hardware.

The rest of the paper is organised as follows. In Section 2 we derive the proposed cyclostationary detection algorithm and compare its performance to the existing solutions in [3], [4] and [5]. In Section 3 we provide details of the FPGA implementation and compare its cost to the detector in [4]. Finally, in Section 4, we draw conclusions.

II. Derivation of Detector Algorithm

OFDM signals exhibit non-conjugate symbol rate cyclostationarity as discussed in [3]. In order to detect cyclostationarity, we compute the CAF as defined below,

$$\hat{R}^{\alpha}_{xx}[\nu] = \frac{1}{N} \sum_{n=0}^{N-1} x[n]x^{*}[n-\nu]e^{-j2\pi\alpha n} \tag{1}$$

where $x[n]$ is the discrete input signal, N is the observation interval, ν is the discrete autocorrelation lag, and α represents the cyclic frequency of interest normalised to the input sampling frequency, f_s. For IEEE 802.11a OFDM operating at f_s = 20MHz, ν = 64 and the fundamental cyclic frequency is $\alpha_0 = 0.0125$. The FFT of the autocorrelation scaled by a factor of $1/N$ is as follows,

$$X[k] = \frac{1}{N}\sum_{n=0}^{N-1} x[n]x^*[n-\nu]e^{-j2\pi kn/N} \tag{2}$$

where $X[k]$ is a single FFT bin and k is the bin index. Inspecting (1) and (2), it can be seen that the CAF corresponds to a single bin of the FFT where $\alpha = k/N$. Note that in the case of IEEE 802.11a the cyclic frequency does not correspond to an exact FFT bin so some spectral leakage occurs. Since the peak corresponding to α_0 is significant when the signal of interest is present, we propose to compute the following test statistic for the detector,

$$\hat{T} = \frac{\left|\hat{R}_{xx}^{\alpha_0}[\nu]\right|^2}{\frac{1}{N}\sum_{k=0}^{N-1}\left|X[k]\right|^2} \tag{3}$$

noting that $\hat{R}_{xx}^{\alpha_0}[\nu]$ is equal to $X[\alpha_0 N]$. This measures the strength of $\left|\hat{R}_{xx}^{\alpha_0}[\nu]\right|^2$ relative to the average squared magnitude of the FFT bins. The detection problem is formulated as a binary hypothesis test where the null hypothesis, H_0, states that the received signal is complex white noise. Conversely, the alternative hypothesis, H_1, states that the received signal consists of the signal of interest plus complex white noise. By determining the probability distribution of (3) under H_0, we can set a threshold η for the detector based on a desired Probability of False Alarm (P_{fa}) as follows [3],

$$\eta = P^{-1}(1 - P_{fa}) \tag{4}$$

where P represents the Cumulative Distribution Function (CDF). The authors in [4] prove that the following is true under H_0,

$$\frac{2N}{\sigma^4}\left|\hat{R}_{xx}^{\alpha_0}[\nu]\right|^2 \sim \chi_2^2 \tag{5}$$

where σ is the standard deviation of the white noise signal. This means that the quantity on the left hand-side of (5) is χ_2^2 distributed under H_0. Using this result, we can state the following,

$$\frac{2N}{\sigma^4}\sum_{k=0}^{N-1}\left|X[k]\right|^2 \sim \chi_{2N}^2 \tag{6}$$

since the sum of N χ_2^2 random variables is χ_{2N}^2 distributed. We will now denote the left hand-sides of (5) and (6) as A and B respectively. Now re-writing (3) in terms of A and B, the test statistic is expressed as,

$$\hat{T} = \frac{\frac{\sigma^4}{2N}A}{\frac{\sigma^4}{2N}B/N}. \tag{7}$$

Removing the common factor, this can be simplified to,

$$\hat{T} = \frac{A}{B/N}. \tag{8}$$

Therefore, under H_0, we compute the ratio of a χ_2^2 random variable to a χ_{2N}^2 random variable divided by a factor of N. It is known that the ratio of two χ^2 random variables each divided by their respective degrees of freedom follows an F distribution. From (5) and (6), it can be seen that the degrees of freedom of the numerator and the denominator are 2 and $2N$ respectively. It is clear that dividing by 2 on the numerator and $2N$ on the denominator is equivalent to dividing by 1 on the numerator and N on the denominator as in (8). Therefore, the test statistic in (3) is $F(2, 2N)$ distributed under H_0. For sufficiently large N, the $F(2, 2N)$ distributed random variable is equivalent to a $\chi_2^2/2$ random variable [6]. This is a gamma distributed random variable with a shape parameter of 1 and a scale parameter of 1. Therefore, (3) is in fact $\Gamma(1, 1)$ distributed under H_0 for a sufficiently large N. Note that in cognitive radio applications, the assumption of a large N is typically valid as sensing needs to be achieved at very low Signal to Noise Ratios (SNRs). Notice that we are still computing the test statistic in the frequency domain, which would require an FFT. This can be resolved by invoking Parseval's theorem [7] and realising that the denominator of (3) can be re-written in the time domain as follows,

$$\frac{1}{N}\sum_{k=0}^{N-1}\left|X[k]\right|^2 = \frac{1}{N^2}\sum_{n=0}^{N-1}\left|x[n]x^*[n-\nu]\right|^2. \tag{9}$$

The N^2 term is present because $X[k]$ is in fact the true FFT scaled by a factor of $1/N$. Therefore, the time domain equivalent of (3) is,

$$\hat{T} = \frac{N\left|\hat{R}_{xx}^{\alpha_0}[\nu]\right|^2}{\frac{1}{N}\sum_{n=0}^{N-1}\left|x[n]x^*[n-\nu]\right|^2}. \tag{10}$$

Notice that we have scaled both numerator and denominator by N to remove the N^2 term. From this point forward, we will denote the denominator of (10) as C to simplify the analysis that follows. We also define the following two quantities,

$$\begin{aligned} R &= \frac{1}{N}\sum_{n=0}^{N-1}\Re(x[n]x^*[n-\nu])^2 \\ I &= \frac{1}{N}\sum_{n=0}^{N-1}\Im(x[n]x^*[n-\nu])^2 \end{aligned} \tag{11}$$

where $C = R + I$. At this stage, we propose a method by which we compute two modified CAFs in a new test statistic. The first modified CAF considers the real part of the autocorrelation,

$$\hat{R}_{rr}^{\alpha}[\nu] = \frac{1}{N}\sum_{n=0}^{N-1}\Re(x[n]x^*[n-\nu])e^{-j2\pi\alpha n} \tag{12}$$

Similarly, the second modified CAF considers the imaginary part,

$$\hat{R}_{ii}^{\alpha}[\nu] = \frac{1}{N}\sum_{n=0}^{N-1}\Im(x[n]x^*[n-\nu])e^{-j2\pi\alpha n} \tag{13}$$

Using (10) and substituting terms from (11), (12) and (13), we propose the test statistic,

$$\hat{T} = \frac{N\left|\hat{R}_{rr}^{\alpha_0}[\nu]\right|^2}{R} + \frac{N\left|\hat{R}_{ii}^{\alpha_0}[\nu]\right|^2}{I}. \tag{14}$$

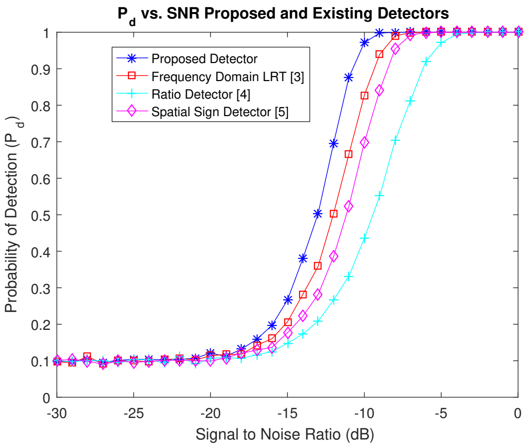

Fig. 1. P_d vs. SNR for fixed point proposed and LRT detectors

Each of the terms retains a $\Gamma(1,1)$ distribution as in (10), meaning that (14) is $\Gamma(2,1)$ distributed. In order to simplify the test statistic, we make the assumption that $R \approx I \approx C/2$ which is true under pure noise conditions. After making substitutions based on this assumption and after some mathematical manipulation, we arrive at a final test statistic,

$$\hat{T} = \frac{2N\left(\left|\hat{R}_{rr}^{\alpha_0}[\nu]\right|^2 + \left|\hat{R}_{ii}^{\alpha_0}[\nu]\right|^2\right)}{C}. \qquad (15)$$

The benefit of using (15) can be quantified by plotting the Probability of Detection (P_d) vs. SNR for the above detector and comparing its performance to other solutions in the literature. Fig. 1 shows P_d vs SNR curves for the proposed detection scheme and the solutions in [3], [4] and [5]. For the monte carlo simulations a total of 1000 trials were conducted at each SNR level and the test signal was IEEE802.11a OFDM in an Additive White Gaussian Noise (AWGN) channel. The detector is set up with an observation interval of $N = 16384$ and $P_{fa} = 0.1$ It can be seen that the proposed detector gives the best detection performance of the considered solutions, achieving a P_d of almost 100% at an SNR as low as -$9dB$. This represents an improvement of approximately $2dB$ over the frequency domain LRT detector in [3]. It can also be seen that the proposed detector significantly out performs the low complexity solutions in [4] and [5]. Having derived the detector and established its applicability for the detection of OFDM signals, we will now go on to discuss its FPGA implementation.

III. FPGA Implementation and Verification

We will now discuss the implementation of the proposed algorithm in MathWorks *HDL Coder* [8] using the FPGA-in-the-loop workflow with an Artix 7 xc7a100t csg324 FPGA. The first stage involves computing the complex autocorrelation lag product $x[n]x^*[n-\nu]$ which requires four multipliers to implement the multiplication, a complex conjugate operation and a delay of 64 (as $\nu = 64$ for IEEE 802.11a signals). Following this, we split the autocorrelation lag product into its real and imaginary parts in order to compute the two modified CAFs in (12) and (13). To compute both of the CAFs, we require three ingredients; a COordinate Rotational Digital Computer (CORDIC) unit operating in rotation mode for the frequency shift by α_0, a re-settable integrator to calculate the summation and a divider to implement the division by N. In our implementation, we elect to use a power of 2 observation interval of $N = 16384$ samples such that the

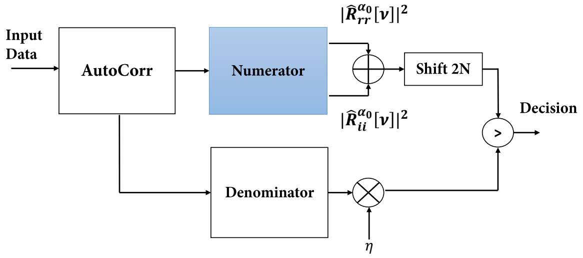
Input Data
AutoCorr
Numerator
$|\hat{R}_{rr}^{\alpha_0}[\nu]|^2$
$|\hat{R}_{ii}^{\alpha_0}[\nu]|^2$
Shift 2N
Decision
>
Denominator
η

Fig. 2. Block Diagram of Detector Implementation

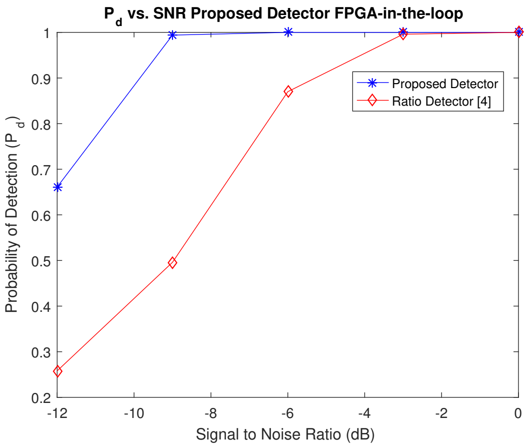

Fig. 3. P_d vs. SNR for Proposed Detector using FPGA-in-the-loop

TABLE I
Resource Utilisation of Proposed Detector on Xilinx Artix 7 xc7a100t csg324 FPGA

FPGA Resource	No. Used	No. Available	% Used
Flip Flops	2,254	126,800	2
LUTs	2,269	63,400	4
BRAMs	0	135	0
DSP48E1s	14	240	6

TABLE II
Resource Utilisation of Detector [4] on Xilinx Artix 7 xc7a100t csg324 FPGA

FPGA Resource	No. Used	No. Available	% Used
Flip Flops	1,764	126,800	1
LUTs	4,104	63,400	6
BRAMs	0	135	0
DSP48E1s	14	240	6

division can be replaced by a simple binary shift operation. For the test statistic in (15) each of the CAFs then undrgoes an $|.|^2$ operation which requires two multipliers and an adder. Since the same functions are computed on both the real and imaginary parts of the autocorrelation, we propose to share the hardware required for both of them using multi-channel techniques. The two channels in this case are the real and imaginary parts of the autocorrelation. The hardware for the CORDIC block, re-settable integrator and $|.|^2$ operation are then shared between the channels, meaning that only one instance of these components is required rather than two in parallel. In order to achieve this, these blocks must run at twice the input sampling frequency and all delays within the shared hardware must be scaled by a factor of 2. Since the input sampling frequency rate is $20MHz$ for IEEE 802.11a, the required clock frequency is $40MHz$ which is achieved by our design. The two channels are multiplexed together at the input to the shared hardware and de-multiplexed at the output. The de-multiplexed channels are then added together as required for the numerator of (15). The result is multiplied by a factor of $2N$ which is implemented as a binary shift since N is a power of 2. The denominator of (15) requires two multipliers and an adder to implement the $|.|^2$ and a re-settable integrator to implement the summation. The division in (15) can be avoided by noting that the test statistic is compared to a constant,

$$2N\left(\left|\hat{R}_{rr}^{\alpha_0}[\nu]\right|^2+\left|\hat{R}_{ii}^{\alpha_0}[\nu]\right|^2\right)>\eta C. \qquad (16)$$

Therefore, we have replaced the division with a scaling of the denominator by the threshold η. Fig. 2 illustrates the processing chain for the proposed detector. The numerator block encompasses all functionality that is shared i.e. the calculation of the terms in (12) and (13) and the $|.|^2$ operation. It has been highlighted in blue to indicate that the hardware is shared. The denominator block covers all processing required to calculate C. We use a 12 bit ADC at the input detector. The wordlength grows to 24 bits after the complex multiplication for the autocorrelation. On the numerator, the CORDIC frequency shift is implemented with 10 iterations and the output wordlength is 24 bits. This is then passed to the re-settable integrator where the wordlength grows to 38 bits (since the observation interval is $N = 16384$) before scaling by $1/N$, after which the wordlength is quantised to 24 bits again. Finally the wordlength is allowed to grow to 48 bits for the $|.|^2$ operation to ensure accuracy. On the denominator, the output of the $|.|^2$ is quantised to 25 bits to limit the bit growth in the integrator to 39 bits and after the scaling by $1/N$ the wordlength is quantised to 25 bits. The result is then multiplied by the constant η. We chose an $\eta = 3.8897$ to guarantee a P_{fa} of 0.1 and this was represented with an unsigned wordlength of 18 bits and 16 fractional bits.

Table 1 shows the cost of this implementation of the proposed detector in terms of Flip Flops, Look Up Tables (LUTs), Block Random Access Memories (BRAMs) and DSP48E1s on the FPGA. In terms of the FPGA fabric, the design is very cost effective consuming only 4% of LUTs. In terms of BRAMs, which are the scarcest resource on the FPGA, our design uses 0%. The design consumes 14 DSP48E1s in its current configuration, which is an acceptable cost. We also implemented the low complexity detector in [4] using equivalent word length choices throughout and applying the same re-arrangement as (16). It can be seen from Table 2 that the cost of both detectors is broadly similar due to the use of hardware sharing in our proposed architecture. Finally, Fig. 3 shows P_d vs. SNR curves for both the proposed detector and [4] generated using the FPGA-in-the-loop feature in HDL Coder. We can see that our detector achieves a P_d of almost 100 % for an SNR of -9dB, thus verifying that the design works as expected on the FPGA. The detector in [4] achieved a P_d of 100 % at -3dB, revealing a 6dB performance gap with no advantage in terms of hardware cost when compared to the proposed detector.

IV. Conclusions

In conclusion, this paper has introduced a low complexity cyclostationary detector for OFDM signals. The derivation of this detector relies on splitting the autocorrelation into its real and imaginary parts and calculating two modified CAFs, before combining them in a final test statistic. It has been shown that the proposed detector performs well compared to existing solutions. An FPGA implementation has also been discussed and the cost in terms of resources has been shown to compare favourably with another low complexity detector coupled with performance advantages. Finally, the implementation has been successfully verified using FPGA-in-the-loop.

References

[1] A. V. Dandawate and G.B. Giannakis. "Statistical tests for presence of cyclostationarity,"*IEEE Transactions on Signal Processing*, vol.42, no. 9, pp. 2355-2369, Sep. 1994.

[2] V. Turunen, M. Kosunen et al, "Implementation of Cyclostationary Feature Detector for Cognitive Radios," in *Cognitive Radio Oriented Wireless Networks and Communications, 2009. CROWNCOM 2009. 4th International Conference on*, Jun. 2009, pp. 1-4.

[3] D. Allan, L. Crockett et al, "FPGA Implementation of a Cyclostationary Detector for OFDM Signals," in *Signal Processing Conference, 2016. EUSIPCO 2016. 24th European*, Aug. 2016, pp. 647-651.

[4] H. Arezumand, P. Azmi et al, "A Robust Reduced-Complexity Spectrum Sensing Scheme Based on Second-Order Cyclostationary for OFDM-Based Primary Users," in *Electrical Engineering, 2011. ICEE 2011. 19th Iranian Conference on*, Jul. 2011, pp. 1-6.

[5] J. Lunden, S. Kassam et al, "Nonparametric Cyclic Correlation Based Detection for Cognitive Radio Systems," *Cognitive Radio Oriented Wireless Networks and Communications, 2008. CrownCom 2008. 3rd International Conference on*, May. 2008.

[6] Stata Data Analysis and Statistical Software, "Relationship between the chi-squared and F distributions,"[Online]. Available: http://www.stata.com/support/faqs/statistics/chisquaredandfdistributions [Accessed: May. 24, 2017]

[7] Steven W. Smith, "Fourier Transform Properties," in *The Scientist and Engineer's Guide to Digital Signal Processing* 2nd ed. San Diego, USA: California Technical Publishing, 1999, ch. 10, sec. 7, pp. 208.

[8] MathWorks, "HDL Coder: Generate Verilog and VHDL Code for FPGA and ASIC designs,"[Online]. Available: http://uk.mathworks.com/products/hdl-coder. [Accessed: May. 24, 2017].