

AN INTERFERENCE SEPARATION APPROACH FOR COMMUNICATION RECEIVERS WITH SINGLE ANTENNA

by

Mehmet Efruz Karabulut

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of
Master of Science

in

Electrical & Electronics Engineering

Koç University

September, 2015

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Mehmet Efruz Karabulut

and have found that it is complete and satisfactory in all respects,

and that any and all revisions required by the final

examining committee have been made.

Committee Members:

Assoc. Prof. Alper Tunga Erdoğan

Prof. Alper Demir

Asst. Prof. Önder Şuvak

Date: _____

Acknowledgements

I would like especially to give thank to my supervisor Assoc. Prof. Alper Tunga Erdoğan for all of his guidance and support during this process. I am very lucky because he always shared his extensive academic experiences with me. I would like to thank to my committee members Prof. Alper Demir and Asst. Prof. Önder Şuvak for their guidances. I would also like to thank to dear Ülkü Oktel, dear Erhan Yılmaz and ASELSAN (HBT) for their contributions to the process.

I would like to thank to Koç University, ASELSAN and Ministry of Science, Industry and Technology for providing me with financial support which made this study possible (This thesis was supported in part by SAN-TEZ 01203.STZ.2012-1).

I would also like to thank all my friends, my family especially to my sister for their support and encouragement.

ABSTRACT

In communication receivers, the system capacity and quality is often degraded by the interference signal. Even though antenna arrays have been the common method to separate interference signal, they may not be feasible for the receivers especially in the direction of down-link because of the physical and economic limitations. Therefore smart algorithms for interference cancellation have been developed that are based only on one received signal delivered by a single receive antenna.

In this thesis, we proposed a complete solution that addresses the single-antenna receiver interference scenario with two co-channel users using $\pi/2$ -BPSK modulation scheme. We divided the proposed solution into cascade of different algorithm blocks for performing distinct tasks. First block targets to estimate the channel corresponding to the initial user, the one who established the first communication link, within a predefined time interval in which most probably interference signal has not been received. The next block monitors the receive signal to detect the start of the interference event. We proposed alternative approaches to detect interference event based on the signal statistics. Once the interference event is detected the third block which is responsible for separating interference from the first user is activated. As a part of the algorithm for this block, we proposed a transformation that converted complex receiver samples to two real signals so that 2×1 complex under-determined source separation problem is converted to 2×2 real determined source separation problem. The main algorithm for the third block is based on Bounded Component Analysis (BCA) proposed as an efficient solution to 2×2 blind source separation problem obtained by the afore-mentioned transformation. The basic merit of this approach is its low complexity and its ability to achieve high separation performance for relatively low number of samples. The final block is responsible for detecting relative phase offset between sources as well as their symbols based on the Maximum Likelihood approach.

The proposed blocks were implemented and tested in both floating point and fixed point MATLAB implementation. We also proposed VHDL implementation guidelines for the hardware realization of the overall system.

ÖZETÇE

Haberleşme alıcılarında, alıcı sistem kapasitesi ve kalitesi girişim sinyali tarafından sıklıkla indirgenir. Anten dizileri girişim sinyallerinin ayırt edilmesini sağlayan yaygın bir yöntem olmasına rağmen, özellikle aşağı-bağlantı yönündeki haberleşme sistemleri için boyut ve ekonomik sınırlamalar sebebiyle uygulanabilir değildir. Bu yüzden sadece tek antenli alıcı tarafından alınan karışım sinyalinden girişim sinyalini ayırtırmayı hedef alan akıllı algoritmalar geliştirilmektedir.

Bu tezde $\pi/2$ -BPSK modülasyonunu kullanan iki eş kanal kullanıcıları tek antenli alıcı girişim senaryosunu ele alan tam bir çözüm önerdik. Önerilen çözüm yöntemini farklı görevler üreten kademeli algoritma bloklarına ayırdık. Birinci blok girişim sinyalinin pratik olarak alınmadığı bir zaman aralığında, ilk haberleşme bağlantısını kuran birinci kullanıcının kanal vektörünün kestirimini hedefler. Bir sonraki blok, girişim olayının başlangıç noktasını sezimlemek için alıcı sinyalini sürekli olarak izler. Girişim sinyalini sezimlemek için sinyal istatistiklerine dayanan alternatif yaklaşımlar sunduk. Girişim olayı sezimlendiği anda girişim kullanıcılarını asıl kullanıcıdan ayırmaktan sorumlu üçüncü blok aktifleştirildi. Bu bloktaki algoritmanın bir parçası olarak, karmaşık alıcı örneklerini iki gerçel sinyale çeviren ve 2×1 karmaşık kesinlik kazanmamış kaynak ayırtırma probleminden 2×2 gerçel belirlenmiş kaynak ayırtırma probleminin elde edilmesini sağlayan bir dönüşüm yöntemi önerdik.

Üçüncü bloktaki asıl algoritma söz konusu olan dönüşümden elde edilen 2×2 kör kaynak ayırtırma problemine etkili bir yöntem olarak sunulan Sınırlı Bileşenler Algoritmasına (BCA) dayanır. Bu yaklaşımın temel değeri sınırlı bileşenler algoritmasının daha düşük kompleksli ve veri sayısının az olduğu durumlarda ayırtırma performansının daha yüksek olmasından ileri gelir. Sonuncu blok kaynaklar ve onların sembolleri arasındaki göreceli faz farkını Maximum Likelihood yaklaşımı ile sezimlede görevlidir.

Sunulan bloklar kayan nokta ve sabit nokta MATLAB uygulamaları olarak uygulandı ve test edildi. Buna ek olarak, bütün sistemin donanım gerçekleştirilmesi için VHDL implementasyon önerileri sunuldu.

Contents

1	Introduction	1
1.1	Outline	8
2	Single Antenna Interference Cancellation Setup	10
2.1	Mathematical Model	10
2.2	Simulation Model	16
2.3	VHDL Implementation Issues	17
2.3.1	Design Specifications	17
2.3.2	Table of Signals	18
2.3.3	Controller Design	19
2.4	The proposed Interference Separation System	20
3	Estimation of The First User's Channel and Noise Variance	23
3.1	Mathematical Setup	23
3.2	Fixed Point Implementation Issues	24
3.3	VHDL Implementation Issues	24
3.3.1	Design Specifications	24
3.3.2	Table of Signals	26
3.3.3	Datapath Design	28
3.3.4	Controller Design	30
3.4	Conclusion	32
4	Interference Event Detection	33
4.1	Mathematical Setup	33
4.2	Adaptive Algorithm	35
4.3	Fixed Point Implementation Issues	37

4.4	Simulation Results	39
4.5	VHDL Implementation Issues	47
4.5.1	Design Specifications	47
4.5.2	Table of Signals	48
4.5.3	Datapath Design	49
4.5.4	Controller Design	51
4.6	Conclusion	53
5	Source Separation Algorithm	54
5.1	BCA Setup	54
5.2	BCA Approach [43]	55
5.3	Adaptive Algorithms	57
5.4	Fixed Point Implementation Issues	59
5.5	Simulation Results	64
5.5.1	Algorithm Convergence	68
5.5.2	Comparison of BCA with state of algorithms	68
5.6	VHDL Implementation Issues	71
5.6.1	Design Specifications	71
5.6.2	Table of Signals	72
5.6.3	Datapath Design	74
5.6.4	Controller Design	76
5.7	Communication through Input Buffer Block-BCA Unit-ML Unit	78
5.8	Conclusion	82
6	Interference Offset and Symbol Detection	83
6.1	Mathematical Setup	83
6.2	Fixed Point Implementation Issues	89
6.3	VHDL Implementation Issues	90
6.3.1	Design Specifications	90
6.3.2	Table of Signals	91
6.3.3	Datapath Design	93

6.3.4	Controller Design	94
6.4	Conclusion	96
7	Conclusion	97
	Bibliography	98

List of Figures

2.1	Constellation diagram where phases of the constellation points are shown in terms of indexes.	12
2.2	Constellation points after complex to real transformation.	13
2.3	Signal to Interference Ratio as a function of phase offset for channel form I.	15
2.4	Signal to Interference Ratio as a function of phase offset for channel form II.	15
2.5	Simulation model developed in MATLAB.	16
2.6	Illustration of block diagram for VHDL design of setup phase.	17
2.7	External control signals, data output signals and registers in VHDL design of setup phase.	18
2.8	Control and status signals in VHDL design of setup phase.	19
2.9	Illustration of state diagram for VHDL controller design in setup phase.	20
2.10	Phases of The Interference Separation Algorithm.	20
3.1	Illustration of block diagram for VHDL design of first phase of the interference separation system.	25
3.2	External control signals, data output signals and registers in VHDL design of first phase of the interference separation system.	27
3.3	Control and status signals in VHDL design of first phase of the interference separation system.	27
3.4	Counter and accumulator design blocks utilized in the first phase of the interference separation system.	28
3.5	Circuit blocks for calculating the channel estimates in the first phase of the interference separation system.	29
3.6	Circuit blocks for calculating the noise variance estimates in the first phase of the interference separation system.	29
3.7	Illustration of state diagram for VHDL controller design of first phase of the interference separation system.	32

4.1	Maximum Eigenvalue as a function of sampling index for $\lambda_1(k)$: $SIR = 0dB$, $\alpha = \frac{1}{10}$.	40
4.2	Minimum Eigenvalue as a function of sampling index for $\lambda_2(k)$: $SIR = 0dB$, $\alpha = \frac{1}{10}$.	40
4.3	$\ \hat{\mathbf{R}}_{\mathbf{y}}(k) - \hat{\mathbf{R}}^{(ni)}\ _F$: $SIR = 0dB$, $\alpha = \frac{1}{10}$.	41
4.4	Maximum Eigenvalue as a function of sampling index for $\lambda_1(k)$: $SIR = 0dB$, $\alpha = \frac{1}{50}$.	42
4.5	Minimum Eigenvalue as a function of sampling index for $\lambda_2(k)$: $SIR = 0dB$, $\alpha = \frac{1}{50}$.	42
4.6	$\ \hat{\mathbf{R}}_{\mathbf{y}}(k) - \hat{\mathbf{R}}^{(ni)}\ _F$: $SIR = 0dB$, $\alpha = \frac{1}{50}$.	43
4.7	Ratio of Eigenvalues as a function of sampling index $\beta_{2,b}(k) = \frac{\lambda_1(k)}{\lambda_2(k)}$: $SIR = 0dB$, $\alpha = \frac{1}{50}$.	44
4.8	Probability of false alert as a function of SNR in case of no interference.	45
4.9	Probability of missing the interference event as a function of SIR for different SNR values.	46
4.10	Averaged estimated index as a function of SIR for different SNR values.	46
4.11	Illustration of block diagram for VHDL design of the second phase of the interference separation system.	47
4.12	External control signals, data output signals and registers in VHDL design of the second phase of the interference separation system.	48
4.13	Control and status signals in VHDL design of the second phase of the interference separation system.	48
4.14	Circuit blocks for covariance estimation in VHDL design of second phase of the interference separation system.	49
4.15	Utilization of circular buffer in VHDL design of second phase of the interference separation system.	50
4.16	Illustration of state diagram for VHDL controller design in the second phase of the interference separation system.	52
5.1	Blind source separation setup.	54
5.2	Bounding Hyper-rectangle and Principal Hyper-ellipsoid.	57
5.3	Convergence steps as a function of number of samples in fixed-point algorithm.	63
5.4	SINR performance of fixed-point BCA algorithm as a function of number of samples.	63

5.5	SIR as a function of phase offset for SIR = -8dB and L = 50 case.	64
5.6	SIR as a function of phase offset for SIR = -4dB and L = 50 case.	65
5.7	SIR as a function of phase offset for SIR = -2dB and L = 50 case.	65
5.8	SIR as a function of phase offset for SIR = 0dB and L = 50 case.	66
5.9	SIR as a function of phase offset for SIR = 2dB and L = 50 case.	66
5.10	SIR as a function of phase offset for SIR = 4dB and L = 50 case.	67
5.11	SIR as a function of phase offset for SIR = 8dB and L = 50 case.	67
5.12	Convergence of BCA algorithm as a function of iterations.	68
5.13	Signal to Interference-Noise Ratio as a function of number of samples for SNR=15dB and SIR=0dB.	69
5.14	Signal to Interference-Noise Ratio as a function of number of samples for SNR=30dB and SIR=0dB.	69
5.15	Signal to Interference-Noise Ratio as a function of number of samples for SNR=15dB and SIR=-6dB.	70
5.16	Signal to Interference-Noise Ratio as a function of number of samples for SNR=15dB and SIR=6dB.	70
5.17	Illustration of block diagram for VHDL design of third phase of the in- terference separation system.	71
5.18	External control signals, data output signals and registers in VHDL de- sign of the third phase of the interference separation system.	72
5.19	Control and status signals in VHDL design of the third phase of the interference separation system.	73
5.20	Control and status signals in VHDL design of the third phase of the interference separation system.	73
5.21	Circuit blocks for BCA algorithm.	75
5.22	Illustration of state diagram for VHDL controller design in the third phase of the interference separation system.	78
5.23	Memory address allocation for input buffer.	79
5.24	Datapath design highlights of input buffer block.	80
5.25	Illustration of state diagram for VHDL controller design in input buffer block.	82
6.1	Alpha values are shown on the constellation diagram.	86
6.2	Mean based estimate of offset as a function of true offset.	87

6.3	Median based estimate of offset as a function of true offset.	88
6.4	Illustration of block diagram for VHDL design of the fourth phase of the interference separation system.	90
6.5	External control signals, data output signals and registers in VHDL design of the fourth phase of the interference separation system.	91
6.6	Control and status signals in VHDL design of the fourth phase of the interference separation system.	92
6.7	Control and status signals in VHDL design of the fourth phase of the interference separation system.	92
6.8	Design specifications of the ML algorithm.	93
6.9	Illustration of state diagram for VHDL controller design in fourth phase of the interference separation system.	96

CHAPTER 1

Introduction

In this thesis, we address the development of complete interference separation/cancellation approach for $\pi/2$ -BPSK receivers employing single antenna.

The interference cancellation in communication systems have been a deep grounded problem. The primary techniques in which the interference problem in communication systems have arised explicitly are located in military field. Utilization of jamming signals by enemy in an attempt to disrupt communication coincided in Second World War that was accepted as first electronic warfare. One of the noticeable examples in this subject was the transmission of the radio jamming by Germans for the purpose of preventing the listening of English BBC radio broadcasting from the continent of Europe, and the brochures which were airdropped by allies to inform about preparation of tuned loop antenna that was aimed to reduce interference against it [1].

The employment of antenna arrays and both boosting signals from a particular set of direction and reducing signals from other directions via antenna arrays were used in Second World War and in cold war era. Estimation algorithm studies on estimation of the desired signal and suppression of undesired signals has continued until today [2, 3]. The studies about the realization of estimation techniques in antenna arrays which were configured to direction of desired signal and the interference signal in practical system reach the 1960s when the adapting filtering systems were appeared [4].

Interference problem have been attracted widespread interest in academy since the 1980s and the 1990s when wireless communications with cellular networks and mobile phones

became irreplaceable in daily life. The essential principle in cellular communications is to divide the geographic area into adjacent cells and using of air as a transmission medium between cellular cells by this method. Fundamental assumption that based on this principle is the absence of interference between the signals of remote cells because of the transmitted power loss on transmission medium and thus to use same carrier frequencies in different cells without any interference between them [5]. Based on this assumption and with the utilization of dynamic power control algorithms, it was predicted to provide coverage for wireless communication via the method of dividing the wide geographic area into cells. However, the assumption that different cells with same frequency carrier may not interfere with each other will become invalid correlated with the reduction in cell area and increase in frequency reuse. In wireless cellular systems, the users using the same carrier frequency and time slots in different cells create an interference which is called co-channel interference.

There are numerous studies on co-channel interference in literature. It is feasible to divide these studies in two main subjects:

Up-link co-channel interference: In cellular networks, the transmission path from the mobile station to a base station is called up-link. The co-channel interference in this direction is generated by the signals on the receiver of base station by the simultaneously transmitting users on different cells which share the same frequency band.

Down-link co-channel interference: In cellular networks, the transmission path from the base station to the mobile station is called down-link. The co-channel interference in this link is arised from the interference signals which base stations in neighbouring cells that use the same frequency band and time windows generate on receiver of user terminal.

Most of the studies in literature on interference cancellation are done for up-link. As the receiver in this link is the base station, feasible methods to prevent interference

have greater extent. It stands to reason that the base station is suitable to place the directional antenna and/or antenna arrays, and its receiver is suitable to employ more complex algorithms in terms of computational complexity. One of the earliest techniques about interference cancellation in this context may be thought as deployment of smart antennas which was categorized as multibeam or adaptive array antennas at base stations [6]. Switched-beam smart antennas [7] can be considered the earliest work for the suppression of co-channel interference in up-link. It is aimed to select best received beams which were formed by using physically directive antenna in these systems.

Single-input multiple-output (SIMO) and multiple-input multiple-output (MIMO) based wireless communication systems appeared with the adaptation of antenna array methods to the analog cellular wireless communication in the late 1990s and 2000s. Related resources and approaches are presented in [8–10]. The main subject of this thesis is related to down-link interference cancellation. Suppression or reduction of interference in this link is realized on receiver of hand terminal for example on mobile phones. The deployment of antenna arrays is not suitable for these receivers because of the limitations due to the physical dimensions and the economic conditions. Therefore, the studies for down-link interference are restricted to single antenna receivers and the studies are classified under the common title “Single Antenna Interference Cancellation”.

The most significant study about interference cancellation with single antenna in industrial field is the “Feasibility Study About Single Antenna Interference Cancellation” which was realized in the frame of 3GPP [11]. Academic researches on single antenna interference cancellation have begun in the 1990s. One of the pioneer studies in this field is the study accomplished by Slock and Trigui for GSM standard based mobile communications by exploiting the real aspect of the symbol constellation after an appropriate reformulation of GMSK signal, leading to a two channel aspect caused by the

in-phase (I) and in-quadrature (Q) components of the receiver as in the two antenna utilization [12,13]. In this study, a significant interference reduction is provided through combining the phases of the received signal by a purely spatial filter. Exploiting the real aspect of the symbol constellation was also performed by Meyer et al in a similar approach [14].

The approaches that exploiting the real aspect of constellation sequences should not be applicable for the communication systems using modulation methods based on complex constellation sequences. Therefore, new techniques are required to generate multichannels except the I and Q components to apply similar linear filter based approach. A potential approach to obtain multichannels is oversampling, when it is infeasible to use antenna arrays. Mixtures of sub-samples may be modelled as outputs of desired multichannel structure with oversampling method. An approach in this direction is presented in [15]. In this study, multichannel output structure generated by oversampling technique for the communication system using MPSK modulation based on complex constellation are modelled as the mixtures of desired signal and interference signal. Independent component analysis (ICA) algorithm is applied to obtain original signals from these mixtures and thus desired and interference signals were successfully separated. The success of mentioned linear filtering based approach depends on availability of the excess bandwidth usage beyond the Nyquist Frequency. Besides the bandwidth usage, similarity factor of frequency responses of channels of desired and interference signals is a fundamental factor. Separation performance of the independent component analysis is positively correlated with the dissimilarity between the channel responses. Therefore, modulation methods used for desired signal and interference signal should be selected differently especially for the channels that are non-frequency selective.

As an alternative to linear filtering techniques, Decision Feedback Equalizer (DFE), which is nonlinear but requires less amount of operation is proposed to utilize for the

aim of interference cancellation [16]. In proposed method, detection of desired signal and interference signal are done separately, then the decisions from these detections feed back to the system to estimate desired signal and the interference signal. Approach in the study is proposed for antenna arrays, but may be adapted to single antenna nevertheless.

The most optimal approach to reduce symbol error rate in the case where true information about channel parameters for desired and interference signals are known for single-antenna receivers is Maximum Likelihood Sequence Estimation (MLSE). MLSE approach for vector channels were produced firstly in [17]. Pioneer studies on interference cancellation with this approach in wireless communication systems are posed in the years 90s. As an example, Joint-MLSE (JMLSE) algorithms by Grindar et al are presented in [18, 19].

In a similar manner, Ranta et al performed JMLSE algorithm on Time Division Multiple Access (TDMA) systems [20]. In this method, they assumed that base stations in the network were synchronously transmit their packets and the training sequences of desired signal and interference signal had low cross-correlation between them. They estimated channel parameters necessary for JMLSE via Least-Squares Method which uses training sequences [21].

In JMLSE, joint recovery of desired signal and the interference signal is intended with dynamic programming (Viterbi Algorithm). However, the computation complexity of the Viterbi Algorithm grows exponentially with channel impulse response length. Including the channel corresponding to interference user increases the computation complexity of the algorithm significantly.

Various approaches are presented on reduction of computation complexity of JMLSE. As an example, two-stage receiver algorithm which is much less complex than JMLSE is

proposed in [19]. Chen et al suggested an approach to adaptively truncate the channel taps with the weakest energy and the spatial diversity combining of signals from multiple antennas for accurate estimation of channel [22].

JMLSE was applied for the aim of separating the EDGE signals under GSM Standard, in a single antenna receiver in [23]. Delayed Decision Feedback Sequence Estimation (DDFSE) method was applied in [24] to reduce the complexity of the joint-estimation problem of desired and interference signals. In this approach, JMLSE algorithm based on the mixture of estimate of reduced number of states and decision feedback block to extract decisions from the best path leading to each state. Shortening of the memory channel is done to reduce the computation complexity of the DDFSE block [25]. It is important to emphasize that modulation methods based on real constellation samples [12, 14] are not feasible for EDGE signals which utilizes 8-PSK modulation method that is based on complex constellation samples.

Generalized version of DDFSE is presented in [26]. In this study, adaptive state allocation algorithm is generalized from one user [27] to a system with multiple interferer.

The critical point in algorithms which try to estimate desired signals and interference signals with methods based on Viterbi algorithm is the estimation of channel parameters. Although methods depending only on periodic burst of known training symbols by assuming ideal synchronization between the base stations [21] may be used, semi-blind methods may also be preferred which combine training-based criterion with a blind criterion [28, 29].

Aforementioned optimum or near-optimum JMLSE approaches provide joint estimation of the desired signals and interference signals. As an alternative, successive interference cancellation (SIC) method based on studies are designed for multi user communication systems in Information Theory [8, 30] and especially on Code Division Multiple Access

(CDMA) cellular systems. The principle in SIC is to detect desired user's signal which using a demodulator and then remove or cancel its effect from the other users through subtraction, and then subtraction of rest of the weaker signals successively. This method is used especially for high source to interference power levels. Application of this approach for single antenna interference cancellation is proposed in [31]. In this study, signal separability is obtained in the cases where there is exploitation of the relative timing delay between co-channel signals and examination of soft subtractions.

Studies on interference cancellation is not limited to wireless communication systems. In this topic, there are researches aim to increase coverage especially in DSL communication networks. In DSL networks even though communication is provided by data transmission on separate telephone lines, cross-talk may be occurred caused by electromagnetic coupling between the unshielded twisted pairs of lines. As an example, study on the cancellation of cross-talk is presented in [32].

The two main assumptions that included in many of the studies mentioned, information about the channel parameters of desired and interference signal and the coordination between the users for joint estimation of these parameters are invalid for the scenarios on the study in this thesis context. Therefore, because of the lack of information about channel parameters and the coordination between the desired and interference signals, the problem in this thesis is referred to blind source separation problem.

Numerous studies were presented on the blind source separation (BSS). A fundamental example pointed out to the pioneer and essential studies on BSS was Analytical Constant Modulus Algorithm by Vanderveen and Paulraj that aimed to separate signals in the channel [33]. Independent Component Analysis (ICA) which is the most common BSS approach was gained popularity in the years 1990s and 2000s [34]. There are numerous approaches and algorithms were presented in this field [35].

ICA approach is a proper candidate for the solution approach due to the success in different applications and the variety of algorithms in this field. However, more recently, a framework named Bounded Component Analysis (BCA) which was developed by group of researchers including Alper T. Erdoğan was proposed as a more feasible approach, especially in communication applications. Source boundedness as a practically valid assumption to be utilized for the separation of signals was posed firstly in that field in addition to the independence assumption proposed in ICA methods [36–39]. Following these studies, Cruces in [40] showed that the source boundedness side information may be used to replace the source independence assumption with a weaker assumption about domain separability and named the corresponding framework as Bounded Component Analysis (BCA). Alper T. Erdoğan proposed studies on Bounded Component Analysis (BCA), following his other studies based on the source boundedness. More recently, under the new framework of Bounded Component Analysis presented by Cruces, an algorithm and a convergence analysis for uncorrelated but dependent sources were proposed by Alper T. Erdoğan [41]. Family of algorithms were derived for the separation of dependent (even correlated) as well as independent sources [42]. The significant superiority of BCA approach is the separation performance improvement over the state of the art ICA based approaches, especially for short data records. In this field, a detailed comparison study is proposed in [43]. In this study, BCA algorithm is compared to ACMA algorithm [33] and state of art ICA algorithms, FASTICA and JADE [35].

1.1 Outline

In this thesis, we present a complete solution for the interference separation, targeting $\pi/2$ -BPSK communication receivers employing single antenna. Chapter 2 begins with the mathematical model corresponding to the single antenna interference problem. The

phases of the proposed interference separation approach are also outlined in this chapter. In Chapter 3, the algorithm block for estimating the first user's channel is presented. In Chapter 4, we present the algorithm block responsible for interference detection. This block is based on continuously monitoring the receive signal to determine the changes in the signal statistics. We propose alternative approaches to detect interference event via evaluating these statistics. We test the performance of the eigenvalue ratio based interference detection method for various Signal-to-Interference Ratio (SIR) and Signal-to-Noise Ratio (SNR) values. In Chapter 5, the algorithm block responsible for separating interference from the first user is presented. Bounded Component Analysis (BCA) algorithm is proposed as an efficient solution to 2×2 blind source separation problem. Separation capability of the proposed BCA algorithm is shown for different SIR scenarios. Additionally, performance of BCA algorithm over the state of the art algorithms are illustrated. In Chapter 6, the algorithm block responsible for detecting relative phase offset between sources as well as their symbols based on the Maximum Likelihood approach is proposed.

Finally, Chapter 7 is for concluding comments.

CHAPTER 2

Single Antenna Interference Cancellation Setup

2.1 Mathematical Model

We introduce the mathematical model corresponding to the single antenna interference problem. We assume a scenario where two identical transmitters, using same modulation schemes, transmit at the same frequency band. The following describes the components of the physical layer of the communication link:

- Modulation Scheme: We assume $\frac{\pi}{2}$ - BPSK modulation for the operation mode of the communication transceivers. For this modulation scheme, the continuous time modulated signal can be written as

$$s(t) = e^{j(\omega_c t + \phi(t))} \quad \text{where} \quad (2.1)$$

$$\phi(t) = \frac{\pi}{4} + \int_0^t f_M(\tau) d\tau \quad (2.2)$$

$$f_M(t) = \frac{\pi}{2T} \sum_k b_k \prod \left(\frac{t - (k + \frac{1}{2})T}{T} \right) \quad (2.3)$$

where T is the symbol period. Binary transmission sequence is represented with $b_k \in \{-1, 1\}$. In $\frac{\pi}{2}$ -BPSK modulation scheme, there is a $\pm\frac{\pi}{2}$ phase difference between each consecutive symbols. The sign of this phase difference is determined by the current data bit value.

- Received Signal: The baseband equivalent of the continuous time received signal (in interference free scenario) is given by

$$x(t) = H(e^{j\omega_c})e^{j\phi(t)} + n(t) \quad (2.4)$$

$$= he^{j\phi(t)} + n(t) \quad (2.5)$$

Here we assumed that the channel is non-frequency selective or flat in the bandwidth of the transmitted signal and $n(t)$ corresponds to receiver noise signal.

- Sampled Received Signal: Symbol-spaced sampled signal with correct phase can be written as

$$x(k) = x(kT) \quad (2.6)$$

$$= he^{j\phi(kT)} + n(kT) \quad (2.7)$$

$$= he^{j(\frac{\pi}{4} + \frac{\pi}{2} \sum_{l=1}^k b_l)} + n(k), \quad k = 0, 1, 2.. \quad (2.8)$$

Therefore

$$x(0) = he^{j\frac{\pi}{4}} + n(0) \quad (2.9)$$

$$x(1) = he^{j(\frac{\pi}{4} + (b_1)\frac{\pi}{2})} + n(1) \quad (2.10)$$

$$x(2) = he^{j(\frac{\pi}{4} + (b_1+b_2)\frac{\pi}{2})} + n(2) \quad (2.11)$$

- On the Phase of Samples: As stated in (2.6 - 2.11), if received signal is sampled at correct phase, $x(k)$ samples can be found by using (2.9 - 2.11). Phases of the constellation points are found with the equation $\mathbf{e}^{j\frac{\pi}{4}}\mathbf{e}^{j\frac{\pi\gamma_k}{2}}$ where $\gamma_k = \sum_{l=1}^k b_l$. As seen from Figure 2.1, if k is odd, phase of the constellation point is equal either $\mathbf{e}^{j\frac{3\pi}{4}}$ or $\mathbf{e}^{j\frac{7\pi}{4}}$. On the other hand, if k is even, phase of the constellation point is

equal either $e^{j\frac{\pi}{4}}$ or $e^{j\frac{5\pi}{4}}$.

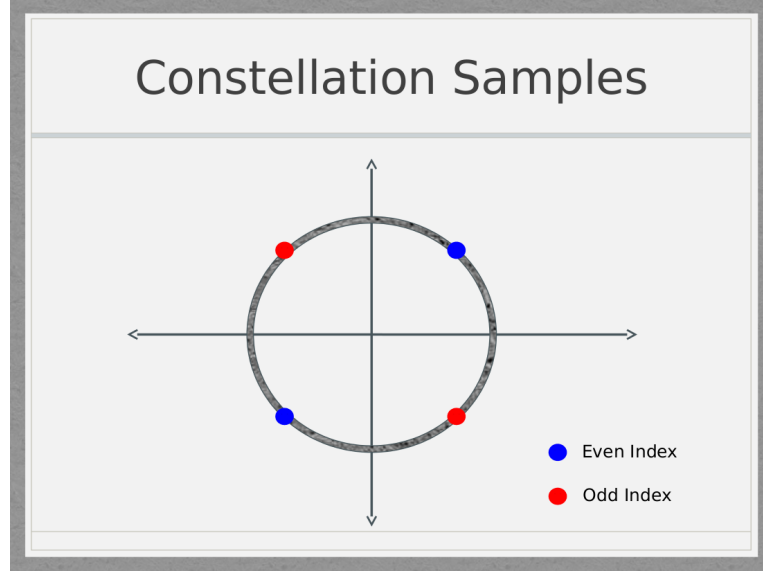


Figure 2.1: Constellation diagram where phases of the constellation points are shown in terms of indexes.

- Transformation to Real Axis: At that point, an operator is defined to transform the complex constellation points to the real axis. As seen from Figure 2.2, after multiplication with $e^{-j\frac{\pi}{4}}$ constellation points corresponding to odd indexes are transformed to imaginary axis of constellation diagram, while points corresponding to even indexes are transformed to real axis. Then, additional multiplication of $e^{-j\frac{\pi}{2}}$ is required to transform the points on the imaginary axis to real axis. These operations are defined in (2.12 - 2.14).

$$\mathbf{\Gamma}(q(k)) = \begin{bmatrix} \Re\{q(k)e^{-j\frac{\pi}{4}}\} \\ \Im\{q(k)e^{-j\frac{\pi}{4}}\} \end{bmatrix} \quad \text{k is even} \quad (2.12)$$

$$\mathbf{\Gamma}(q(k)) = \begin{bmatrix} \Re\{q(k)e^{-j\frac{3\pi}{4}}\} \\ \Im\{q(k)e^{-j\frac{3\pi}{4}}\} \end{bmatrix} \quad \text{k is odd} \quad (2.13)$$

$$\alpha(k) = \begin{cases} j^{\gamma(k-1)-1} & \text{k is odd} \\ j^{\gamma(k-1)} & \text{k is even} \end{cases} \quad (2.14)$$

In the true phase case if we define $\mathbf{y}(k) = \mathbf{\Gamma}(x(k))$ then we can write

$$\mathbf{y}(k) = \begin{bmatrix} \Re\{h\} \\ \Im\{h\} \end{bmatrix} \alpha(k) + \begin{bmatrix} \Re\{v(k)\} \\ \Im\{v(k)\} \end{bmatrix} \quad (2.15)$$

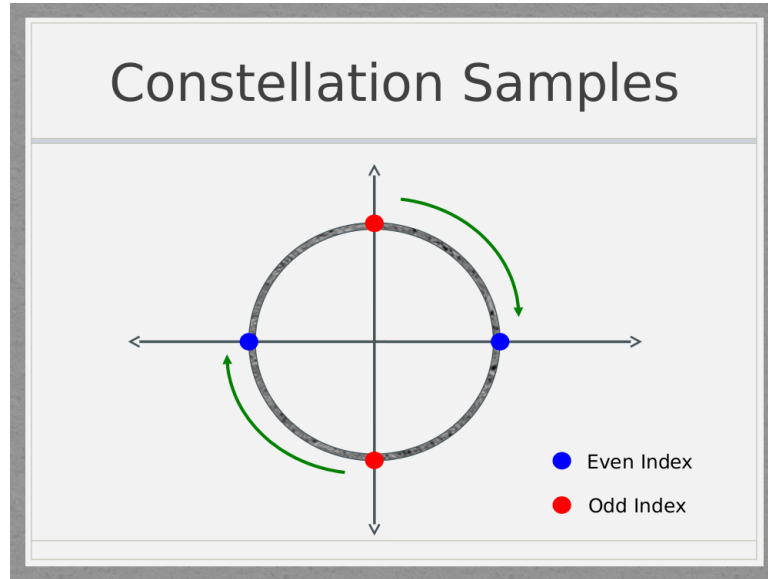


Figure 2.2: Constellation points after complex to real transformation.

- Interference Channel:

When two co-channel users exists and (unrealistically) assumed to have aligned phase at the receiver, then we can write the transformed signal as

$$\mathbf{y}(k) = \begin{bmatrix} \Re\{h_1\} & \Re\{h_2\} \\ \Im\{h_1\} & \Im\{h_2\} \end{bmatrix} \begin{bmatrix} \alpha_1(k) \\ \alpha_2(k) \end{bmatrix} + \mathbf{v}(k) \quad (2.16)$$

- Offset Problem

As stated earlier it is assumed that both signals have aligned phase at the receiver. However, it is possible for interference user to include at any moment. Instead, we may state that the sampling phase is synced to first user, whereas the interference user has the arbitrary relative phase with $\frac{T}{2M}$ granularity. For instance, if phase offset between the first user and the interference user is $\frac{\pi}{2}$, then phase irregularity can be solved by embedding the phase offset into the channel. Hence, both users have aligned phase at the receiver.

$$\mathbf{H}_1 = \begin{bmatrix} \Re\{h_1\} & \Re\{h_2\} \\ \Im\{h_1\} & \Im\{h_2\} \end{bmatrix} \quad (2.17)$$

$$\mathbf{H}_2 = \begin{bmatrix} \Re\{h_1\} & -\Im\{h_2\} \\ \Im\{h_1\} & \Re\{h_2\} \end{bmatrix} \quad (2.18)$$

Signal Energy to Interference Energy Ratio (SIR) were computed for each phase offset value. As shown in Figures 2.3 and 2.4, measure of phase offset is represented by the unit M . One symbol period between the first user and the interference user includes $2M$ sub-phases. Figure 2.3 indicates that performance is degraded for first user for the choice of $\mathbf{H}_1$ in the vicinity of $2M$. According to these figures, as the phase offset

become closer to $2M$, the employment of $\mathbf{H}_2$ is more accurate for the evaluation of the (Source to Interference Ratio) SIR expression.

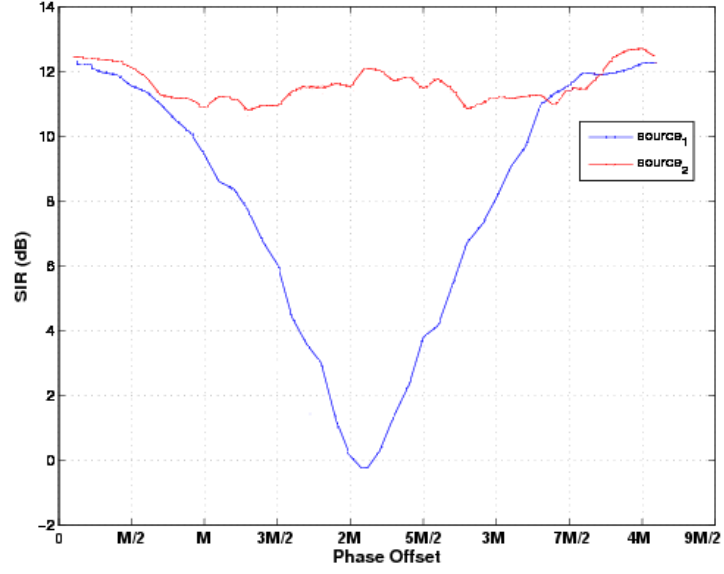


Figure 2.3: Signal to Interference Ratio as a function of phase offset for channel form I.

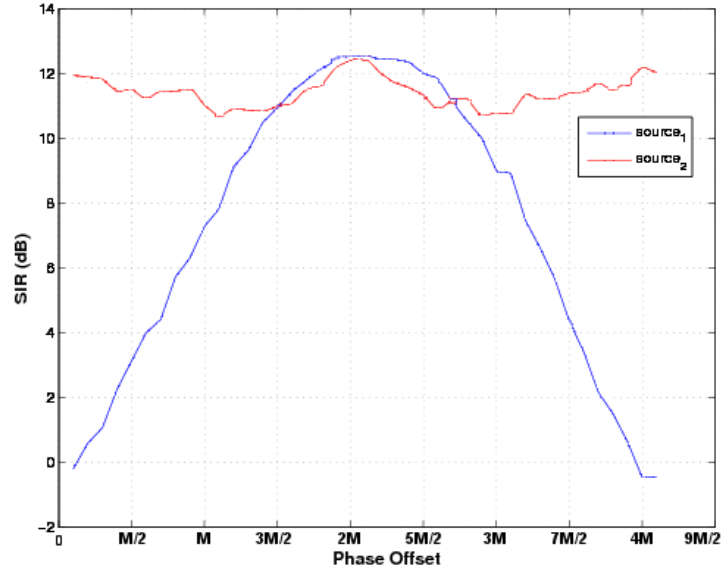


Figure 2.4: Signal to Interference Ratio as a function of phase offset for channel form II.

2.2 Simulation Model

In this section, we introduce the simulation model which was developed and tested in MATLAB environment. Figure 2.5 shows the equivalent baseband model at $2M \times Ts$ rates. $s_1(m)$ and $s_2(m)$ are the transmitted samples of first user and the interference user at $2M \times Ts$ rates. Phase offset block assign an arbitrary relative phase offset to the interference user with $\frac{T}{2M}$ granularity. We may exploit the additional phase information caused by this sampling rate to estimate relative phase offset. The problem is discussed in more details on Chapter 6. $\mathbf{h}_1$ and $\mathbf{h}_2$ are the mixing channel models for each user respectively. The mixtures are represented with $x(m)$ and down-sampled with decimation factor of $2M$. Then, complex-valued observations are transformed to two real channels $y_1(k)$ and $y_2(k)$.

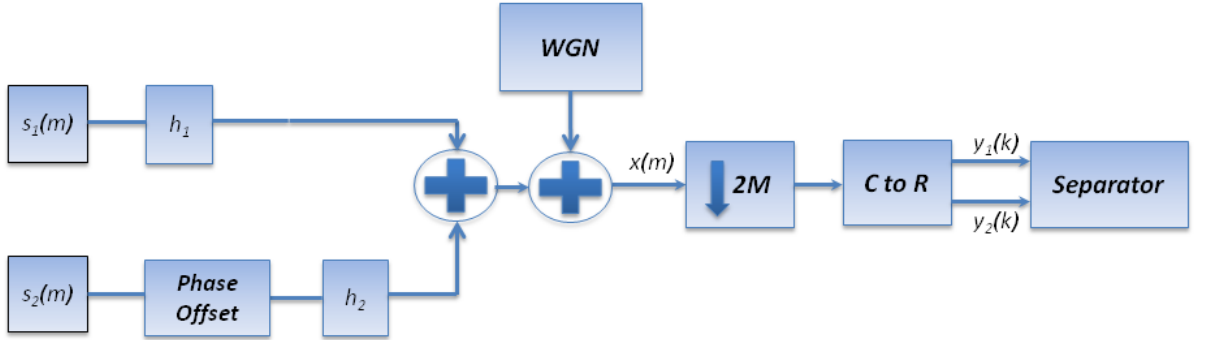


Figure 2.5: Simulation model developed in MATLAB.

2.3 VHDL Implementation Issues

2.3.1 Design Specifications

A digital system can be represented at the register transfer level (RTL). RTL design is composed of registers and combinational function blocks (e.g. adders and multiplexers) called the datapath and a finite state machine called the controller that controls the transfer of data through the function blocks and between the registers. In the setup phase, relatively simple RTL design is implemented to prepare the external data samples to the algorithm blocks. Block diagram is shown in Figure 2.6. The hardware model takes $x_1(m)$ and $x_2(m)$ samples as an external inputs at $2M \times T_s$ sampling rate. We aware that in the hardware realization, $x_1(m)$ and $x_2(m)$ are corresponded to the real and imaginary parts of the complex $x(m)$ mixtures. Therefore, complex $x(m)$ should be replaced with its real and imaginary parts in the fixed point simulation model for the convenience with hardware model. Frequency of *SamplerClk* clock is set to the referred sampling rate. Additionally, *MCCLK* clock is the processor clock which is set to the processor operation speed. This design block aims to manipulate the $x_1(m)$ and $x_2(m)$ samples to obtain $y_1(k)$ and $y_2(k)$ samples.

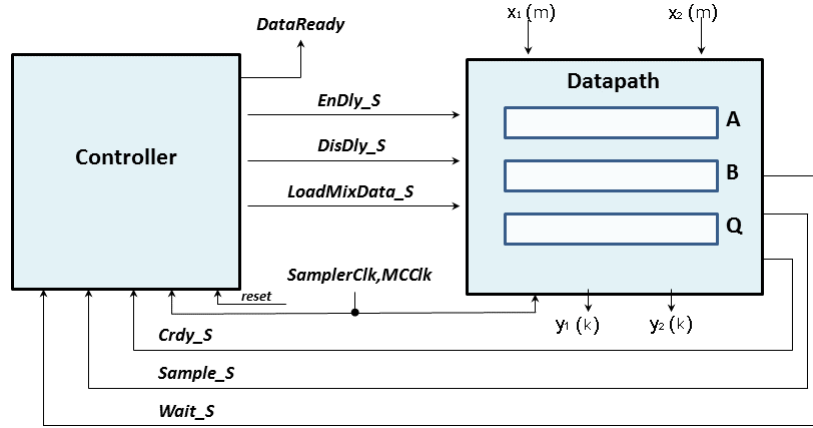


Figure 2.6: Illustration of block diagram for VHDL design of setup phase.

Datapath unit consists of three subblocks:

- Subblock **A**: *SamplerClk* is assumed to be synced to first user while receiving first data samples. Then, an up-counter unit increments a variable each time *SamplerClk* changes to its positive edge. When the count value reaches to $2M$, a status signal is sent to the controller to take permission for selection of the new samples. Then $x_1(m)$ and $x_2(m)$ samples corresponding to symbol-spaced samples are hold. Another variable is incremented for tracking the index of the symbol-spaced samples.
- Subblock **B**: Transformation operation is realized. By checking the least significant bit of the index variable of the symbol-spaced samples, correct rotation parameters are utilized.
- Subblock **Q**: Transformed samples are stored in register Q temporarily.

2.3.2 Table of Signals

External control input/output signals, external data output signals and registers are indicated in Figure 2.7.

Symbol	Function	Type
<i>reset</i>	Initialize registers to 0	Control input
<i>DataReady</i>	Indication that converted received samples are registered and ready to use	Control output
<i>DR</i>	BCD counter	7-Bit register
<i>MDR</i>	Parallel load register	10-Bit register

Figure 2.7: External control signals, data output signals and registers in VHDL design of setup phase.

Control and status signals connect controller unit and datapath unit. Figure 2.8 shows these signals with their meanings and which registers they alter.

Action or Status	Control or Status Signals	Meaning for Values
$DR \leftarrow 0$ $DR \leftarrow DR + 1$	$DisDly_S$ $EnDly_S$	1 : Reset DR to 0, 0 : No reset of DR 1 : BCD count up DR by 1, 0 : Hold DR value
$MDR \leftarrow B_output$	$LoadMixData_S$	1 : Enable load MDR register, 0 : Disable load MDR register
$DR = N, DR < N$	$Crdy_S$	1 : DR is equal to N , 0 : DR is less than N

Figure 2.8: Control and status signals in VHDL design of setup phase.

2.3.3 Controller Design

- **S1:** Just after asynchronous *reset*, controller is in state S1 in which the content of the register DR and MDR is clear.
- **S2:** Circuit proceeds to S2 to wait for $Sample_S$ signal. $Sample_S$ is ‘1’ when the unit **A** holds the received samples which correspond to symbol-spaced samples.
- **S3:** Circuit waits in this state until the operations of task **A** and **B** are done. Operation time of these circuits are known and compared to waiting time in S3 state which is stored in DR . Then an enable signal is asserted if they are equal. Circuit proceeds to S4.
- **S4:** The moore-type control signal is asserted to load MDR and state proceeds to S5 at the next positive edge of the clock.
- **S5:** *DataReady* output indicates that transformed receive samples are registered and ready for process in cascaded algorithm blocks.
- **S6:** Wait until next positive edge of $SamplerClk$. Then, circuit proceeds to S2 to wait for next symbol-spaced samples.

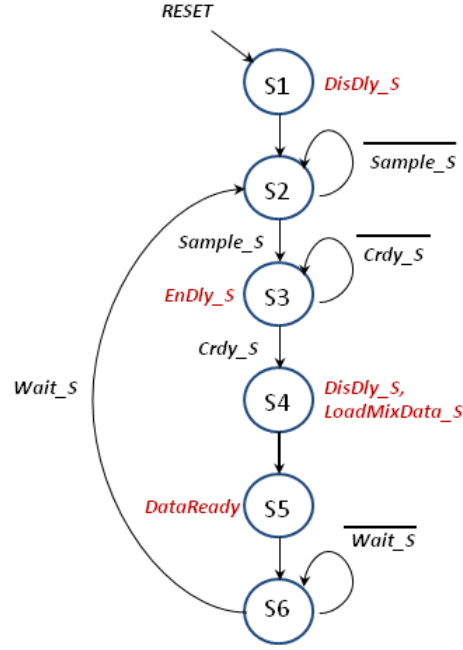


Figure 2.9: Illustration of state diagram for VHDL controller design in setup phase.

2.4 The proposed Interference Separation System



Figure 2.10: Phases of The Interference Separation Algorithm.

The proposed system involves cascade of different algorithm blocks for performing different tasks for achieving the solution for the single antenna interference cancellation problem. In order to construct this solution we assumed the following scenario:

- Initially the communication link is established by one of the users. We refer to this user as the “First User” in our algorithmic setting.
- and regard the other user as “the Second User” or interference.

Although this terminology based on the transmission start times can be replaced by the higher layers based on the priority of these users as the proposed solution reproduces both user’s signals.

Based on this assumption and the corresponding terminology, the proposed solution is formed by the following algorithmic steps shown in Figure 2.10:

- **Phase 1-*Estimation of the First User’s Channel and Noise Variance*** : In the initial transmission period, when only the first user is active the channel corresponding to this user and the receiver noise variance is estimated (based on the sync word). This stage is material for resolving the permutation ambiguity in the blind source separation process as well as determining the relative phase offset with the help of ML decoding process, which is critical for the two-phase channel model in linear approximation of the nonlinear data model. Furthermore, the noise variance is obtained to be used in the interference detection phase.
- **Phase 2-*Interference Event Detection***: Once the first user’s channel is estimated the proposed block enters into the interference event detection phase by continuously monitoring the receive signal to determine the changes in the signal statistics. The proposed solution is based on detecting the change in the second order statistics especially for the implementational convenience.

- **Phase 3-Source Separation Algorithm:** This is the key phase of the proposed solution block, where the blind source separation algorithm is applied to the transformed receiver samples. The resulting separator matrix is later used to estimate two different alternative channel estimates. We employ Bounded Component Analysis algorithm which has the desirable property of convergence with small amount of data samples relative to the state of the art methods such as ACMA, JADE and FASTICA algorithms.
- **Phase 4-Interference Offset and Symbol Detection:** Once the separator training is complete separator matrix is used to generate two channel matrix candidates to perform Maximum Likelihood detection for joint phase offset and symbol detection. This phase produces two different bit sequence candidates for each user which can be checked by the higher layers to select the correct bit sequence for each user.

CHAPTER 3

Estimation of The First User's Channel and Noise Variance

3.1 Mathematical Setup

Approximate solution to the first channel may be found by least squares method. As can be seen from (3.1), it is a linear least square problem. Γ operator in (3.1) is defined to transform the complex constellation samples to real axis. Let $S = \{s(1), s(2), \dots, s(N)\}$ denote the set of constellation samples at Ts rate. Minimizing the sum of squared differences between the actual value of the dependent variable and the value predicted by the model gives the best approximation. Solving the linear least square problem reveals that $\hat{\mathbf{H}}_{:,1}$ can be written at the form (3.2) and at (3.3) in a open form.

$$\underbrace{\begin{bmatrix} y_1(1) & y_1(2) & \dots & y_1(N) \\ y_2(1) & y_2(2) & \dots & y_2(N) \end{bmatrix}}_{\mathbf{Y}} = \underbrace{\begin{bmatrix} \Re\{h_1\} \\ \Im\{h_1\} \end{bmatrix}}_{\mathbf{H}_{:,1}} \underbrace{\begin{bmatrix} \Gamma(s(1)) & \Gamma(s(2)) & \dots & \Gamma(s(N)) \end{bmatrix}}_{\mathbf{g}^T} + \mathbf{v}(k) \quad (3.1)$$

Least Squares Estimate:

$$\hat{\mathbf{H}}_{:,1} = \frac{\mathbf{Y}^* \mathbf{g}}{\|\mathbf{g}\|_2^2} \quad (3.2)$$

$$\hat{\mathbf{H}}_{:,1} = \frac{\sum_{k=1}^N \mathbf{Y}_{:,k}^* g_k}{\sum_{k=1}^N |g_k|^2} = \frac{1}{N} \sum_{k=1}^N \mathbf{Y}_{:,k}^* g_k \quad (3.3)$$

Considering the data in (3.1) where $\mathbf{v}(k)$ is WGN with variance σ_v^2 is to be estimated. Maximum likelihood estimate can be found from

$$\frac{\partial \ln p(\mathbf{Y}; \sigma^2)}{\partial \sigma^2} = -\frac{N}{2\sigma^2} + \frac{1}{2\sigma^4} \sum_{k=1}^N (\mathbf{Y}_{:,k} - \mathbf{H}_{:,1} * g_k)^2 \quad (3.4)$$

Solving for σ^2 in (3.4) and using $\hat{\mathbf{H}}_{:,1} = \mathbf{H}_{:,1}$ produces

$$\hat{\sigma}_v^2 = \frac{1}{N} \sum_{k=1}^N (\mathbf{Y}_{:,k} - \hat{\mathbf{H}}_{:,1} * g_k)^2 \quad (3.5)$$

3.2 Fixed Point Implementation Issues

There are no critical fixed point implementation issues as the algorithm is relatively straightforward. Inputs are appropriately scaled integers. The only issue is the division by the norm of samples in the averaging process for both the channel estimate and the noise variance estimate. For the chosen number of samples, i.e. 64, this can be implemented by a 6 right shifts. We note that we used a separate scaling for the integer representation of the transmit symbols on the unit circle for better channel estimation accuracy.

3.3 VHDL Implementation Issues

3.3.1 Design Specifications

Block diagram in Figure 3.1 shows the external inputs, external outputs and the blocks that exist in design process of first phase of the interference separation system. In this phase, the main task is to design RTL to estimate the first user's channel and noise

variance which stated mathematically in (3.3) and (3.5). $y_1(k)$, $y_2(k)$ and $\Gamma(s(k))$ are symbol-spaced samples. We need N training samples for estimation of (3.3) and (3.5). A rom unit is initialized to $\Gamma(s(k))$ values. Two identical size distribution ram units are inferred to store $y_1(k)$ and $y_2(k)$ samples. N parameter in (3.3) and (3.5) must be smaller than the depth of these ram units to store all the training data in the same memory units. In the block diagram, we specify that r address lines are required to refer 2^r memory words. It is convenient to predefine N as an equal parameter to number of memory words. It is also convenient to select the parameter N as power of two to replace division operation with arithmetic shift right operation.

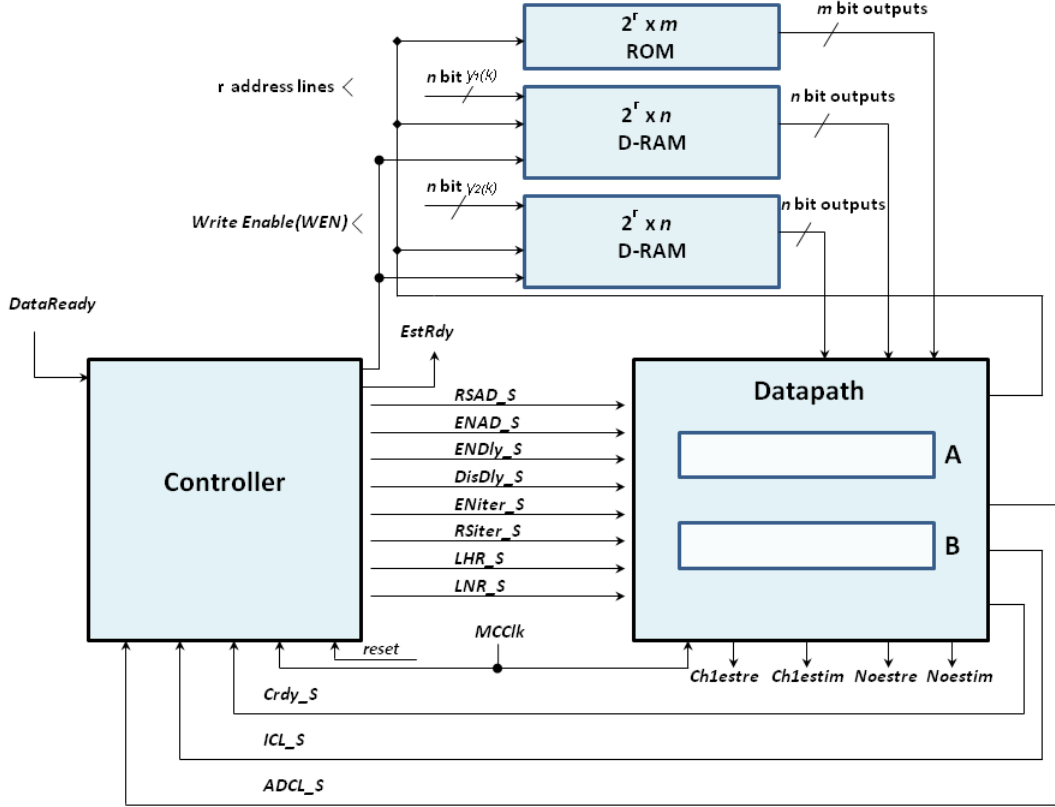


Figure 3.1: Illustration of block diagram for VHDL design of first phase of the interference separation system.

Two approaches are considered for the estimation process. In first approach, we may store all the training data, before permission signals to start estimation algorithms are send to the controller. In this technique, controller unit is simplified but waits N symbol period prior to the estimation. In second approach, we may read and process each sample as soon as it is written to the memory. It infers more complex controller circuit in return controller no longer waits for the receiving time of all the training data. We decided to use first approach. Channel and noise variance estimates are found in two iterations. At first iteration, $y_1(k)$, $y_2(k)$ and $\Gamma(s(k))$ samples are read from memory blocks with the permission of controller and send to datapath unit to be involved in the circuit which computes (3.3). As soon as channel estimates are found, address locations are reset. At second iteration, same set of samples read from memory, but be involved in the computation of (3.5).

3.3.2 Table of Signals

Some register types are used for the entire design blocks. *ADR*, *DR* and *ITR* registers are timer registers with count enable and asynchronous reset options. *ADR* register is a timer register needs to count up to point next address location in the memory block. *DR* register aims to adjust propagation delay of a specified circuit. Due to the complexity of the design, some circuits could not be designed in gate level so that it may not be possible to register their outputs within a clock cycle. Therefore, a timer register with a comparison circuit is required to compare the pre-defined propagation delay of the circuit(in terms of clock cycles) with the count of clock cycles and decide accordingly.

Symbol	Function	Type
<i>RESET</i>	Initialize registers to 0	Control input
<i>DataReady</i>	Indication of arrival of the transformed received samples	Control input
<i>WEN</i>	Enable the <i>write</i> input of the memory unit	Control output
<i>EstRdy</i>	Estimates of the first user's channel and noise variance is ready	Control output
<i>ADR</i>	BCD counter	6-Bit register
<i>DR</i>	BCD counter	7-Bit register
<i>ITR</i>	BCD counter	2-Bit register
<i>CHR</i>	Parallel load register	10-Bit register
<i>NR</i>	Parallel load register	13-Bit register

Figure 3.2: External control signals, data output signals and registers in VHDL design of first phase of the interference separation system.

Action or Status	Control or Status Signals	Meaning for Values
<i>ADR</i> $\leftarrow$ 0 <i>ADR</i> $\leftarrow$ <i>ADR</i> + 1	<i>RSAD_S</i> <i>ENAD_S</i>	1 : Reset <i>ADR</i> to 0, 0 : No reset of <i>ADR</i> 1 : BCD count up <i>ADR</i> by 1, 0 : Hold <i>ADR</i> value
<i>CHR</i> $\leftarrow$ <i>CHRnext</i> <i>NR</i> $\leftarrow$ <i>NRnext</i>	<i>LHR_S</i> <i>LNR_S</i>	1 : Enable load <i>CHR</i> register, 0 : Disable load <i>CHR</i> register 1 : Enable load <i>NR</i> register, 0 : Disable load <i>NR</i> register
<i>ITR</i> $\leftarrow$ 0 <i>ITR</i> $\leftarrow$ <i>ITR</i> + 1	<i>RSiter_S</i> <i>ENiter_S</i>	1 : Reset <i>ITR</i> to 0, 0 : No reset of <i>ITR</i> 1 : BCD count up <i>ITR</i> by 1, 0 : Hold <i>ITR</i> value
<i>DR</i> $\leftarrow$ 0 <i>DR</i> $\leftarrow$ <i>DR</i> + 1	<i>DisDly_S</i> <i>ENDly_S</i>	1 : Reset <i>DR</i> to 0, 0 : No reset of <i>DR</i> 1 : BCD count up <i>DR</i> by 1, 0 : Hold <i>DR</i> value
<i>DR</i> = <i>N</i> , <i>DR</i> < <i>N</i>	<i>CRDY_S</i>	1 : <i>DR</i> is equal to predefined threshold, 0 : <i>DR</i> is less than predefined threshold
<i>ITR</i> = "10", <i>ITR</i> = "01"	<i>ICL_S</i>	1 : <i>ITR</i> is equal to "10", 0 : <i>ITR</i> is equal to "01"
<i>ADR</i> < s.size -1, <i>ADR</i> = s.size, <i>ADR</i> > s.size	<i>ADCL_S</i>	00 : <i>ADR</i> is less than given sample size - 1, 01 : <i>ADR</i> is equal to sample size, 10 : <i>ADR</i> is greater than sample size

Figure 3.3: Control and status signals in VHDL design of first phase of the interference separation system.

3.3.3 Datapath Design

The circuits shown in Figure 3.5 and 3.6 are composed of combinational circuits and registers. Estimation algorithms reveal that until N parameter is reached in the summation unit, an accumulator circuit is required to keep summation results. When it is reached, an arithmetic shift register (ASR) circuit is used to shift accumulation result. However, in VHDL these circuits work parallel. At each cycle of combinational circuit, both ASR and accumulation circuit process its inputs. To prevent from unintended combinational circuit outputs, multiplexer is put to select one of the outputs of the mentioned circuits. Figure 3.4 show design blocks with an up-counter and a comparator unit which decide the selection inputs of the multiplexers. Finally, either accumulation result or the averaged result is registered at the positive edge of the clock with a condition that load enable signal is active before the edge of the clock.

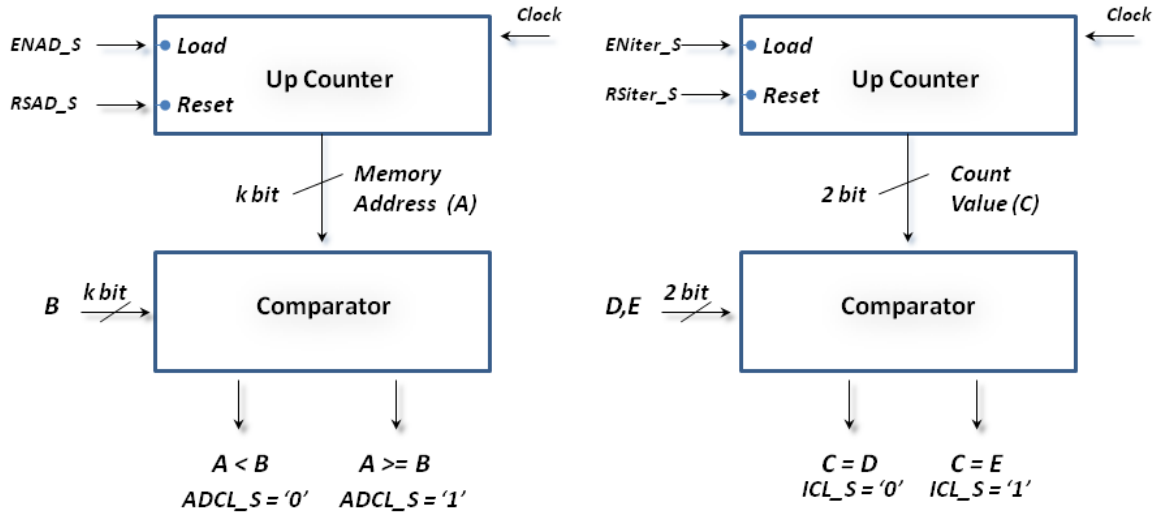


Figure 3.4: Counter and accumulator design blocks utilized in the first phase of the interference separation system.

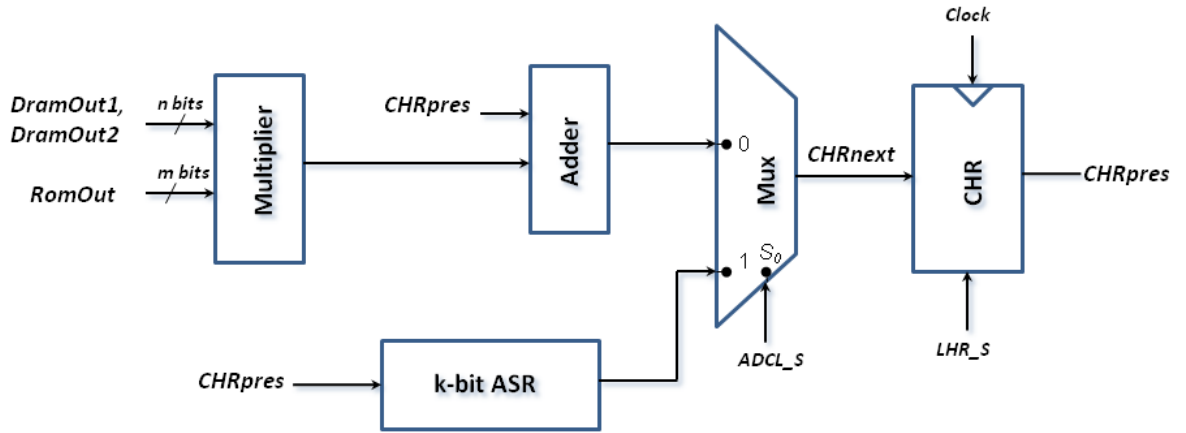


Figure 3.5: Circuit blocks for calculating the channel estimates in the first phase of the interference separation system.

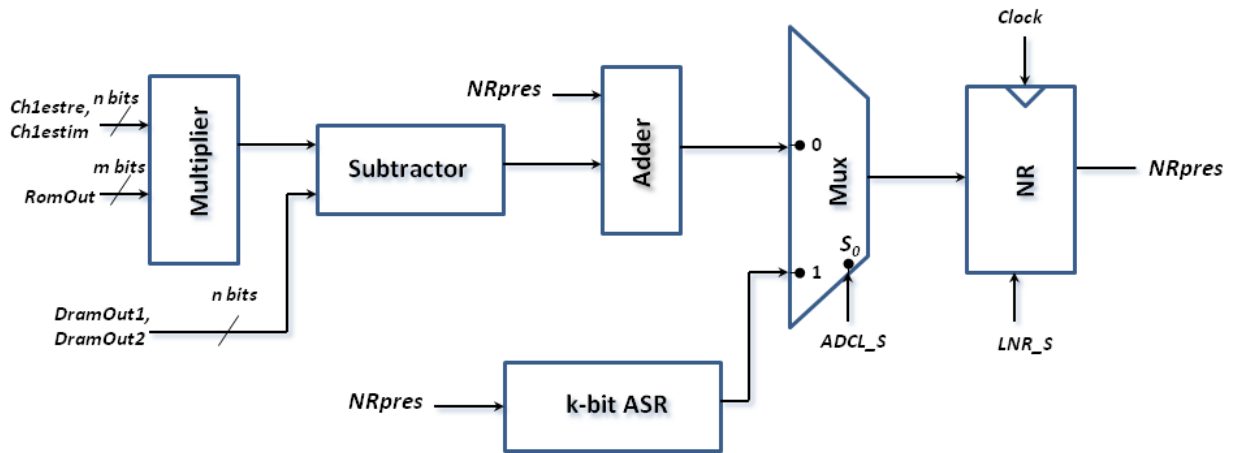


Figure 3.6: Circuit blocks for calculating the noise variance estimates in the first phase of the interference separation system.

3.3.4 Controller Design

- **S1:** Just after asynchronous *RESET*, controller is in state S1 in which the content of the address register *ADR*, the delay register *DR* and the iteration register *ITR* is clear.
- **S2:** Circuit proceeds to S2 to wait for *DataReady* = '1'. As long as *DataReady* = '0', the state remains at S2. *DataReady* = '1' causes a transition to state S3.
- **S3:** *WEN* is toggled to '1' to store $y_1(k)$ and $y_2(k)$ samples at the k 'th address location in two parallel distributed ram.
- **S4:** *ADR* is enabled to count up by 1 to reserve next address location for future sample. Circuit proceeds to S5 if all the training samples are buffered. A logic block controls the content of memory addresses after each write operation and status the *ADCLS* signal accordingly. If the address value that write address pointer addresses is smaller than $N-1$, circuit proceeds to S2 to wait for new samples. Otherwise it proceeds to S5.
- **S5:** As soon as circuit reaches to S5, we have entire data set stored in our memory blocks ready for computation. Address pointer must hold the first address location. Therefore, *ADR* is reset to clear the address content. Then *ITR* register is enabled by *ENiter_S* signal to increment the iteration index. We have two equations to deal with. Even though circuits run in parallel for computation, we can think of that at first iteration (3.3) result is computed and registered. Then at second iteration (3.5) is computed and registered. *ICLS* signal is assigned false when *ITR* timer is equals to integer 1. *ICLS* signal is assigned true when *ITR* timer equals to integer 2. In S6, controller decide which equation to deal with according to *ICLS* signal.

- **S6:** This state is the core state of the controller unit. In this state, the combinational circuits in Figure 3.5 and 3.6 process the training data. If ICL_S signal is false, CHR register is loaded at next state, otherwise NR register is loaded. However, propagation delays in Figure 3.5 and 3.6 take more than one clock cycle. To prevent wrong loading to the registers, DR register counts up until outputs of the combinational circuits become stable. Then $CRDY_S$ signal is send by the datapath unit for controller to proceeds S7 or S8.
- **S7:** In this state, load signal is enabled for CHR register. ADR register is enabled to read next samples from memory for next computations. If computation of (3.3) has not done yet, circuit proceeds to S6. Otherwise, it proceeds to S5.
- **S8:** In this state, load signal is enabled for NR register. ADR register is enabled to read next samples from memory for next computations. If computation of (3.5) has not done yet, circuit proceeds to S6. Otherwise, it proceeds to S9.
- **S9:** Both computations are done. Then $NRDY$ flag is toggled to ‘1’ for indication.

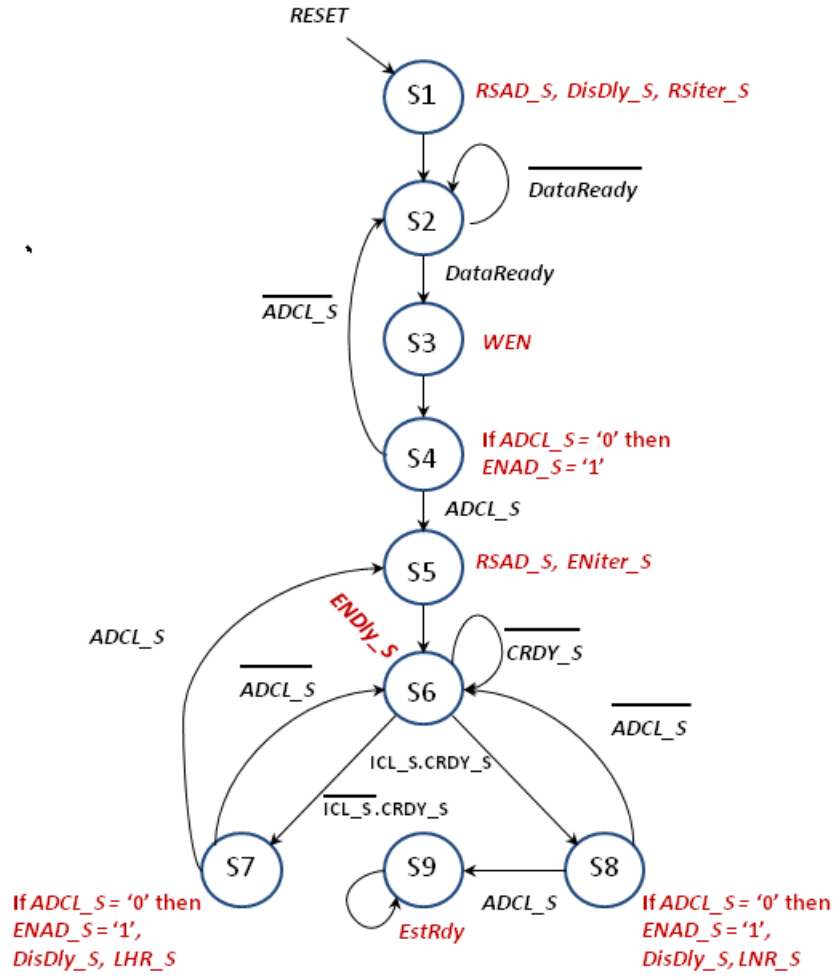


Figure 3.7: Illustration of state diagram for VHDL controller design of first phase of the interference separation system.

3.4 Conclusion

In this section, we presented linear least square method for estimating the channel corresponding to the first user. Then, we presented maximum likelihood method for estimating the receiver noise variances.

CHAPTER 4

Interference Event Detection

4.1 Mathematical Setup

We consider the following interference scenario consisting of two main phases:

- **No Interference Phase:** In this phase, only the initial user is active and the receiver is regularly receiving signals from this user. The received data model for this phase can be written as

$$\mathbf{y}^{(ni)}(k) = \underbrace{\begin{bmatrix} \Re\{h_1\} \\ \Im\{h_1\} \end{bmatrix}}_{\mathbf{H}_{:,1}} \Gamma(s_1(k)) + \mathbf{v}(k) \quad (4.1)$$

This expression assumes that the receiver is synced to the the initial user's signal at the correct phase. Assuming independence of the transmitter source samples and noise samples, the covariance of the received signal can be written as

$$\mathbf{R}^{(ni)} = \sigma_{\Gamma(s_1)}^2 \mathbf{H}_{:,1} \mathbf{H}_{:,1}^T + \sigma_v^2 \mathbf{I}. \quad (4.2)$$

We note that this covariance can be estimated based on the channel and noise variance estimates obtained during this phase as

$$\hat{\mathbf{R}}^{(ni)} = \hat{\sigma}_{\Gamma(s_1)}^2 \hat{\mathbf{H}}_{:,1} \hat{\mathbf{H}}_{:,1}^T + \hat{\sigma}_v^2 \mathbf{I}, \quad (4.3)$$

or directly through the sample covariance matrix

$$\hat{\mathbf{R}}^{(ni)} = \frac{1}{N} \sum_{k=1}^N (\mathbf{y}^{(ni)}(k) - \bar{\mathbf{y}}^{(ni)})(\mathbf{y}^{(ni)}(k) - \bar{\mathbf{y}}^{(ni)})^T, \quad (4.4)$$

where

$$\bar{\mathbf{y}}^{(ni)} = \frac{1}{N} \sum_{k=1}^N \mathbf{y}^{(ni)}(k). \quad (4.5)$$

- **Interference Phase:** In this phase, the interference user exists, in addition to the initial user. Unlike the initial user, however, we can not assume that the receiver is in sync with the interference transmitter's phase. Therefore, the data model for this phase can be written as

$$\begin{aligned} \mathbf{y}^{(int)}(k) &= \underbrace{\begin{bmatrix} \Re\{h_1\} \\ \Im\{h_1\} \end{bmatrix}}_{\mathbf{H}_{:,1}} \Gamma(s_1(k)) \\ &+ \underbrace{\begin{bmatrix} \Re\{h_2\} \\ \Im\{h_2\} \end{bmatrix}}_{\mathbf{H}_{:,2}} \Re\{\Gamma(s_2(k))\} \\ &+ \underbrace{\begin{bmatrix} -\Im\{h_2\} \\ \Re\{h_2\} \end{bmatrix}}_{\mathbf{H}_{:,3}} \Im\{\Gamma(s_2(k))\} + \mathbf{v}(k) \end{aligned} \quad (4.6)$$

$$\begin{aligned} &= \underbrace{\begin{bmatrix} \Re\{h_1\} & \Re\{h_2\} & -\Im\{h_2\} \\ \Im\{h_1\} & \Im\{h_2\} & \Re\{h_2\} \end{bmatrix}}_{\mathbf{H}} \begin{bmatrix} \Gamma(s_1(k)) \\ \Re\{\Gamma(s_2(k))\} \\ \Im\{\Gamma(s_2(k))\} \end{bmatrix} \\ &+ \mathbf{v}(k) \end{aligned} \quad (4.7)$$

Therefore, the corresponding covariance matrix can be written as

$$\mathbf{R}^{(int)} = \mathbf{H} \begin{bmatrix} \sigma_{\Gamma(s_1)}^2 & 0 & 0 \\ 0 & \sigma_{\Gamma(s_2)}^2 \cos^2(\phi_{s_2}) & 0 \\ 0 & 0 & \sigma_{\Gamma(s_1)}^2 \sin^2(\phi_{s_2}) \end{bmatrix} \mathbf{H}^T + \sigma_v^2 \mathbf{I} \quad (4.8)$$

4.2 Adaptive Algorithm

We first define a running estimate of covariance calculated based on the recursion

$$\hat{\mathbf{R}}_{\mathbf{y}}(k+1) = (1-\alpha)\mathbf{R}_{\mathbf{y}}(k) + \alpha\mathbf{y}(k)\mathbf{y}(k)^T \quad (4.9)$$

which is initialized at during the No-Interference Phase as $\hat{\mathbf{R}}_{\mathbf{y}}(0) = \hat{\mathbf{R}}^{(ni)}$. We considered following alternative approaches:

1. **Measurement of Deviation from Initial Covariance Matrix:** This approach is based on the fact that entrance of interference event will cause a change in the covariance matrix (from $\mathbf{R}^{(ni)}$ to $\mathbf{R}^{(int)}$). Therefore, one appropriate approach is to detect change in the running estimate of covariance $\hat{\mathbf{R}}_{\mathbf{y}}(k)$ relative to the initial covariance $\hat{\mathbf{R}}^{(ni)}$, which is calculated during the no interference phase. We define

$$\beta_1(k) = \frac{\|\hat{\mathbf{R}}_{\mathbf{y}}(k) - \hat{\mathbf{R}}^{(ni)}\|_F^2}{\|\hat{\mathbf{R}}^{(ni)}\|_F^2} \quad (4.10)$$

The detection event can be based on comparing $\beta_1(k)$ against a chosen threshold level τ_1 .

2. **Detection Based on the Eigenvalue Disparity of Covariance Matrix:**

Another practical approach is based on the observation to detect the interference

based on the change in the disparity of the covariance matrix. We note that the initial phase covariance matrix $\mathbf{R}^{(ni)}$ has large eigenvalue disparity due to the existence of only one source. Its minimum eigenvalue is equal to the noise variance. However, the covariance after the interference event, i.e., $\mathbf{R}^{(i)}$, is expected to have larger minimum eigenvalue, reflecting the level of the interference, especially when $\mathbf{H}$ is sufficiently far from rank deficiency.

The eigenvalues of the covariance estimate can be written as

$$\lambda_1(k) = \frac{\text{Tr}(\hat{\mathbf{R}}_{\mathbf{y}}(k)) + \sqrt{\text{Tr}(\hat{\mathbf{R}}_{\mathbf{y}}(k))^2 - 4 \det(\hat{\mathbf{R}}_{\mathbf{y}}(k))}}{2} \quad (4.11)$$

, which is the maximum eigenvalue, and

$$\lambda_2(k) = \frac{\text{Tr}(\hat{\mathbf{R}}_{\mathbf{y}}(k)) - \sqrt{\text{Tr}(\hat{\mathbf{R}}_{\mathbf{y}}(k))^2 - 4 \det(\hat{\mathbf{R}}_{\mathbf{y}}(k))}}{2} \quad (4.12)$$

which is the minimum eigenvalue. We can offer two options for detecting interference event:

- a. *Tracking the Minimum Eigenvalue* ($\lambda_2(k)$): As noted earlier, main effect of interference would be the increase in the minimum eigenvalue's value. Therefore, potentially we can check the variations in $\lambda_2(k)$ to detect the interference event. For this purpose, we can suggest comparing

$$\beta_{2,a}(k) = \frac{\lambda_2(k)}{\lambda_2(0)} \quad (4.13)$$

against a threshold $\tau_{2,a}$.

- b. *Tracking the Eigenvalue Ratio* ($\frac{\lambda_1(k)}{\lambda_2(k)}$): $\beta_{2,a}(k)$ function introduced in the previous section has the issue that if the overall signal is scaled (for example

by AGC) it would also increase the value of the minimum eigenvalue, which therefore can cause false alarm for the interference detection. In order to safeguard against such scalings, we can look at the ratio of the eigenvalues to detect the interference event. For this purpose, we introduce

$$\beta_{2,b}(k) = \frac{\lambda_1(k)}{\lambda_2(k)} \quad (4.14)$$

and the corresponding threshold, $\tau_{2,b}$. If the level $\beta_{2,b}(k)$ goes below the threshold $\tau_{2,b}$ then we declare that as the interference event.

4.3 Fixed Point Implementation Issues

We implemented the eigenvalue ratio algorithm with integers in MATLAB:

- We used the similar scaling factor that we used in the integer implementation of the BCA algorithm:

$$Scale = \frac{PS}{\hat{\sigma}_{\mathbf{y}}} \quad (4.15)$$

where P is the target integer range corresponding to the number of bits b used in the variable representation, i.e. $P = 2^{b-1} - 1$ and S is for the adjustment factor to adjust peak to average ratio for the corresponding variable. We can write $S = 10^{-PAR/20}$ where PAR is the target peak to average ratio in dB.

- Both sample covariance matrix $\hat{\mathbf{R}}_{\mathbf{y}}(k)$ and the eigenvalues $\lambda_{1,2}(k)$ are implemented based on the scaling based integer representation. However, for its convenience in division implementation we considered a near floating point implementation where we represent eigenvalues in terms of their exponents, which represents the

number of decimal digits after scaling: Let $\lambda_{m,int}(k)$ be the scaled version of the original eigenvalue $\lambda_m(k)$, i.e.,

$$\lambda_{m,int}(k) = Scale * \lambda_m(k), \quad m = 1, 2. \quad (4.16)$$

Therefore, in near-floating point implementation, we used

$$\lambda_{m,int}(k) \approx 10^{\lambda_{m,intexp}(k)} \quad m = 1, 2. \quad (4.17)$$

Therefore, $\beta_{2,b}(k)$ can be implemented as

$$\beta_{2,b}(k) = \lambda_{1,intexp}(k) - \lambda_{2,intexp}(k). \quad (4.18)$$

However, this exponent based representation works fine when SNR level is relatively high so that the initial eigenvalue ratio can be represented in order of magnitudes. One potential solution is to consider modified eigenvalue ratio

$$\frac{\lambda_1(k) - \gamma \hat{\sigma}_v^2}{\lambda_2(k) - \gamma \hat{\sigma}_v^2} \quad (4.19)$$

where $\gamma < 1$ is used to avoid zero in the numerator. Such an adjustment is expected to take care of the low SNR level issue.

- Square Root Implementation

An integer square root calculates $q = \sqrt{n}$ where n is the radicand, q is the root, and both n and q are integers. Algorithm steps are defined:

- Guess some value g for q .
- Compute n/g .
- If n/g is close enough to g , g is the square root.

- Otherwise, we try a better guess.
- α scaling used in the covariance updates are chosen as $1/2^n$ where $n \in \mathcal{Z}$ so that multiplication with them can be implemented as integers.

4.4 Simulation Results

We consider various scenarios to illustrate the effect of algorithm parameters:

α parameter in the covariance update recursion determines the nature of tracking with a clear trade off: if α parameter is chosen to have large value, then the covariance estimate will be largely dependent on the last observation vector, which increases the variance of the covariance estimate. Whereas, if we choose α to be small in this case the algorithm will be slow in updating variations in the covariance and therefore, in detecting the interference event.

We first consider a scenario where interference and the original signal has the same power level and interference enters between symbols 82-83.

- $\alpha = \frac{1}{10}$ **Case:** In Figure 4.1, the maximum eigenvalue $\lambda_1(k)$ is shown we can see that there is also noticable variation in the maximum eigenvalue, but not in a level to reliably detect the location of the interference event.

In Figure 4.2, the minimum eigenvalue $\lambda_2(k)$ is shown. There is a noticeable change (corresponding to an approximate scaling of 4.15). However, we should note that we assume that no AGC adjustments are assumed, therefore, there is no ambiguity in the power levels of signals in first and second phase. Despite the clear interference capturing behaviour of $\lambda_2(k)$, such AGC adjustments would make it an unrobust means of interference detection.

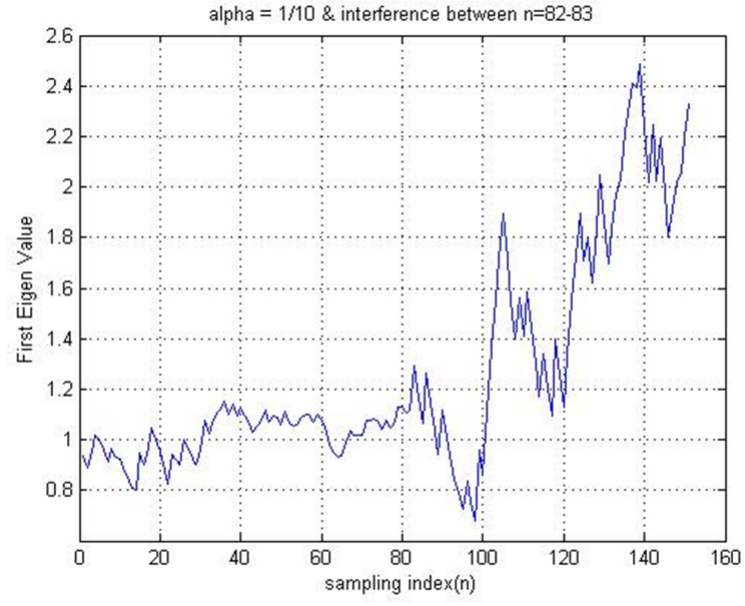


Figure 4.1: Maximum Eigenvalue as a function of sampling index for $\lambda_1(k)$: $SIR = 0dB$, $\alpha = \frac{1}{10}$.

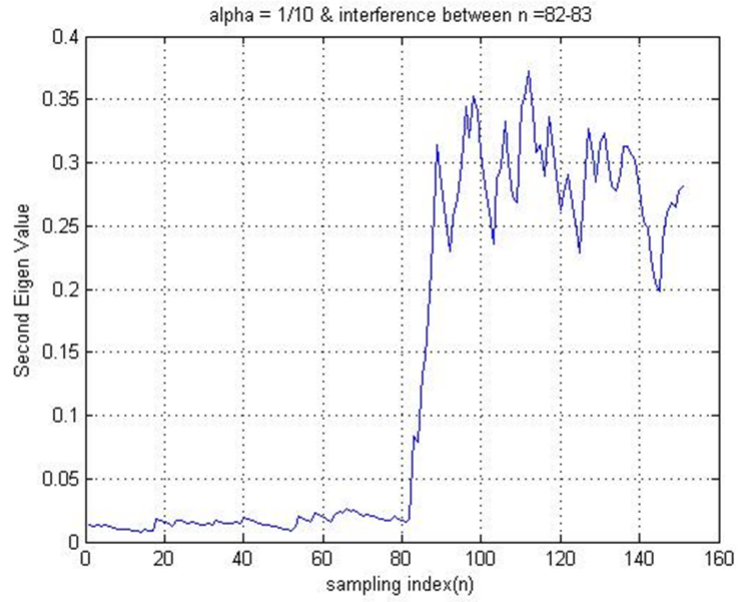


Figure 4.2: Minimum Eigenvalue as a function of sampling index for $\lambda_2(k)$: $SIR = 0dB$, $\alpha = \frac{1}{10}$.

Finally, in Figure 4.3, the variation in the norm of the Frobenius norm of the covariance matrix relative to its initial value, i.e., $\|\hat{\mathbf{R}}_{\mathbf{y}}(k) - \hat{\mathbf{R}}^{(ni)}\|_F$ is shown. Although there is a clear change in this indicator value, it is not as sharp as the minimum eigenvalue, which introduces lagging and ambiguity in detecting the interference event.

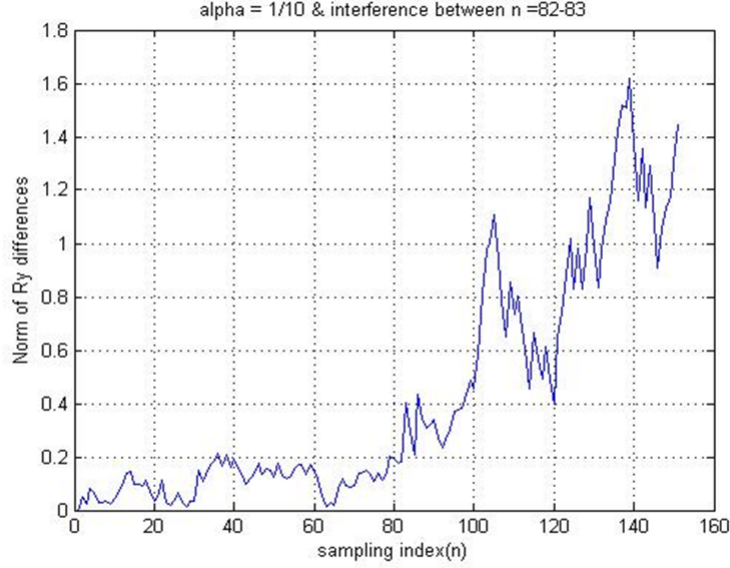


Figure 4.3: $\|\hat{\mathbf{R}}_{\mathbf{y}}(k) - \hat{\mathbf{R}}^{(ni)}\|_F$: $SIR = 0dB$, $\alpha = \frac{1}{10}$.

- $\alpha = \frac{1}{50}$ **Case:** In this case we reduce the α to a smaller value. Figure 4.4 shows the maximum eigenvalue in this case. The major observation is that the time variations are smoother due to decreased variance in the estimation of covariance and therefore its eigenvalues.

Figure 4.5 shows the minimum eigenvalue for this reduced α case. Despite a smoother curve compared to Figure 4.2 corresponding to larger α , it is less sharper and less precise for detecting the location of the interference event.

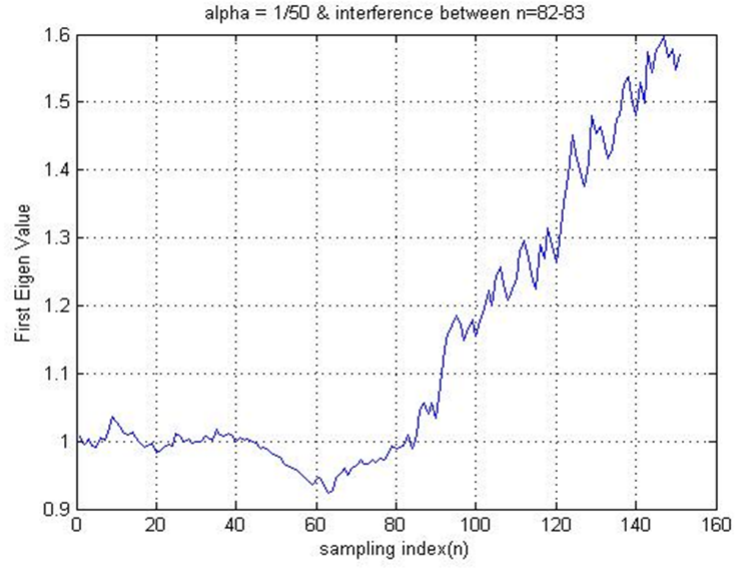


Figure 4.4: Maximum Eigenvalue as a function of sampling index for $\lambda_1(k)$: $SIR = 0dB$, $\alpha = \frac{1}{50}$.

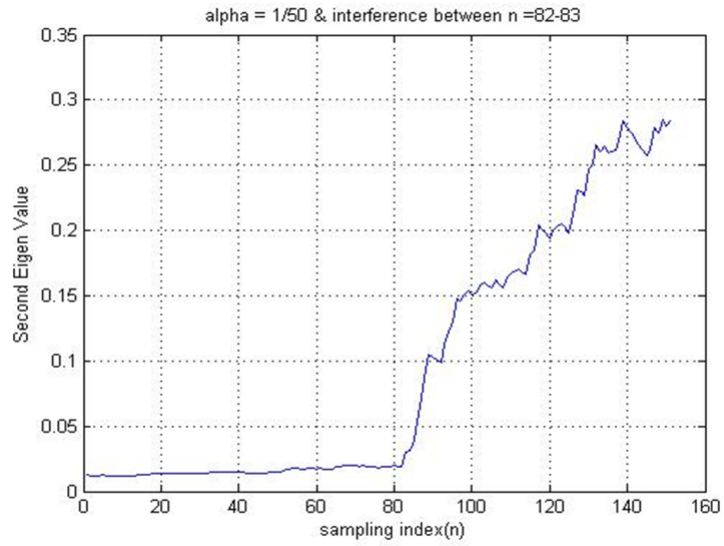


Figure 4.5: Minimum Eigenvalue as a function of sampling index for $\lambda_2(k)$: $SIR = 0dB$, $\alpha = \frac{1}{50}$.

Figure 4.6 shows the corresponding norm of difference, where again we see a smoother behavior as in the case of eigenvalues.

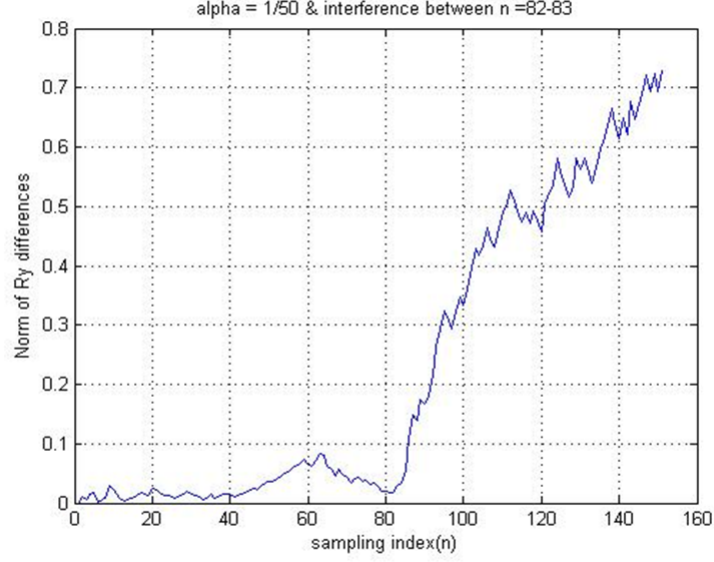


Figure 4.6: $\|\hat{\mathbf{R}}_{\mathbf{y}}(k) - \hat{\mathbf{R}}^{(ni)}\|_F$: $SIR = 0dB$, $\alpha = \frac{1}{50}$.

Finally, Figure 4.7 displays, the eigenvalue ratio $\beta_{2,b}(k) = \frac{\lambda_1(k)}{\lambda_2(k)}$. This seems to be the best indicator with an abrupt change in its value in the vicinity of the interference entrance point despite the fact that α value is relatively small.

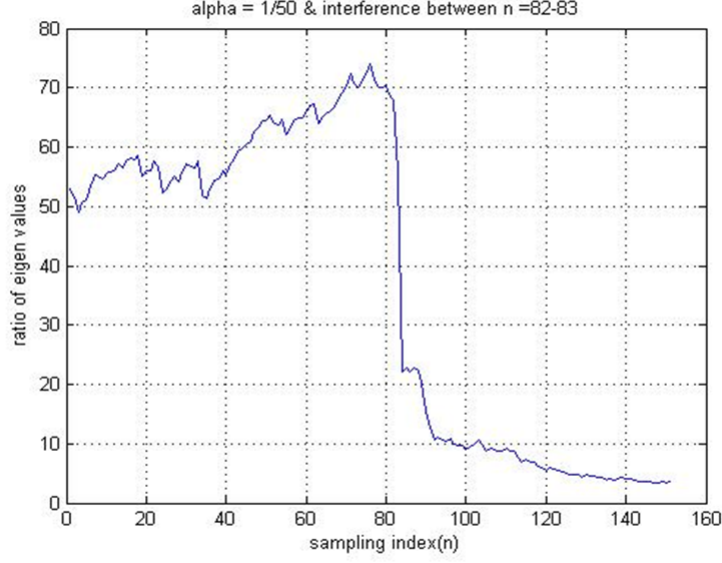


Figure 4.7: Ratio of Eigenvalues as a function of sampling index $\beta_{2,b}(k) = \frac{\lambda_1(k)}{\lambda_2(k)} : SIR = 0dB, \alpha = \frac{1}{50}$.

As a summary of the above simulations, we note that

- Smaller the α smoother the detection index curves, which is desirable. However, smaller α values also imply increased lag and ambiguity in determining the interference event.
- Among the introduced detection indexes, the best one seems to be the eigenvalue ratio $\beta_{2,b}(k)$, which has sharper behaviour around interference entrance point and more immune to lags and ambiguities caused by smaller α values. Furthermore, this index safeguards against random scalings potentially caused by the AGC adjustments.

Next, we tested the eigenvalue ratio based interference detection algorithm and its fixed point implementation's performance against various SIR and SNR values.

Figure 4.8 displays false alarm probability as a function of SNR in case of no interference. This is a relevant metric of interest to avoid unnecessary employment of source separation algorithm where the probability values can be further decreased by adjusting the threshold value $\tau_{2,b}$.

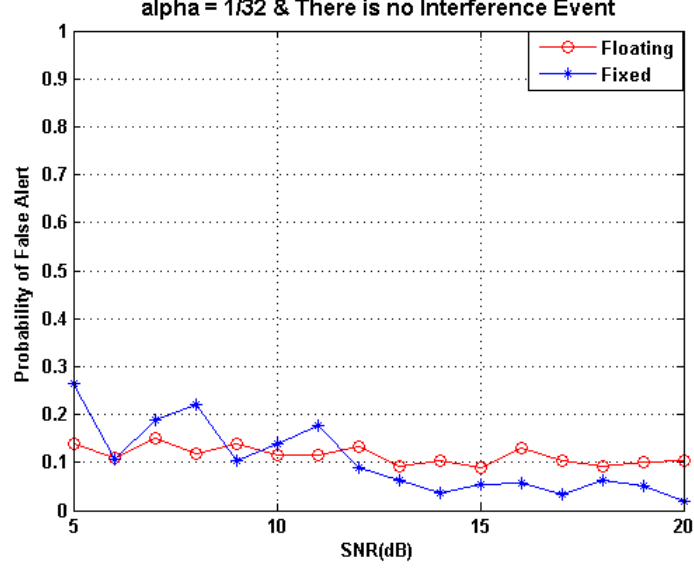


Figure 4.8: Probability of false alert as a function of SNR in case of no interference.

Figure 4.9 shows the missing probability (for detecting interference event) as a function of SIR and various SNR levels. We note that this probability is highly dependent on the level of interference relative to the noise floor. If the noise floor is sufficiently low, the interference event miss probability is relatively low even in low interference levels.

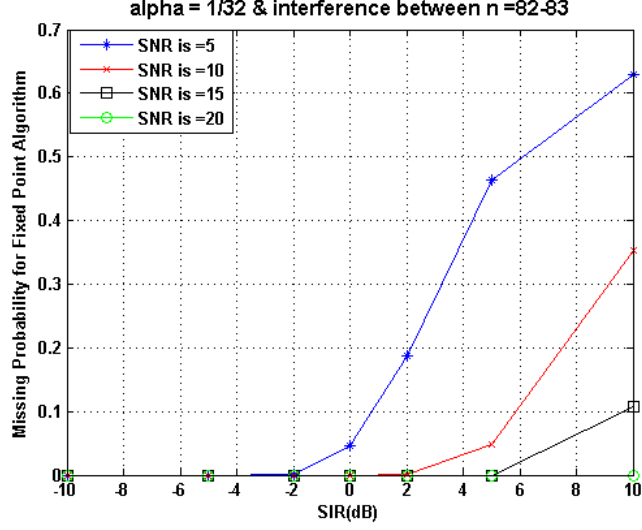


Figure 4.9: Probability of missing the interference event as a function of SIR for different SNR values.

Finally, Figure 4.10 shows the estimated average index locations (when the true index location is between 82-83) as a function of SIR for different SNR levels. As expected the performance drops, with decreasing interference level where it becomes harder to detect interference.

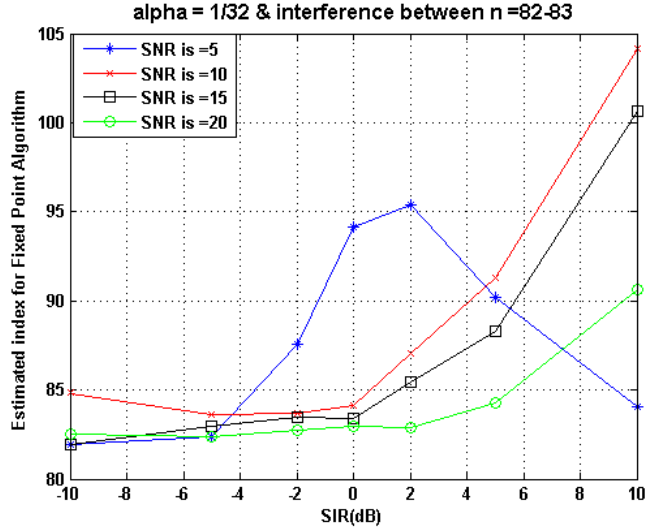


Figure 4.10: Averaged estimated index as a function of SIR for different SNR values.

4.5 VHDL Implementation Issues

4.5.1 Design Specifications

In this phase the goal is to design RTL to implement adaptive algorithm stated in (4.9) and tracking the eigenvalue ratio to detect interference event. As be seen from Figure 4.11, circular buffer is preferred as memory unit. Circular buffer has N elements, one write port and dual read ports. Reading from circular buffer is done asynchronously and depends only on read address pointer. Selection of buffer width, type and port type is not an ordinary choice. In the 4.5.3, it will be explained in details.

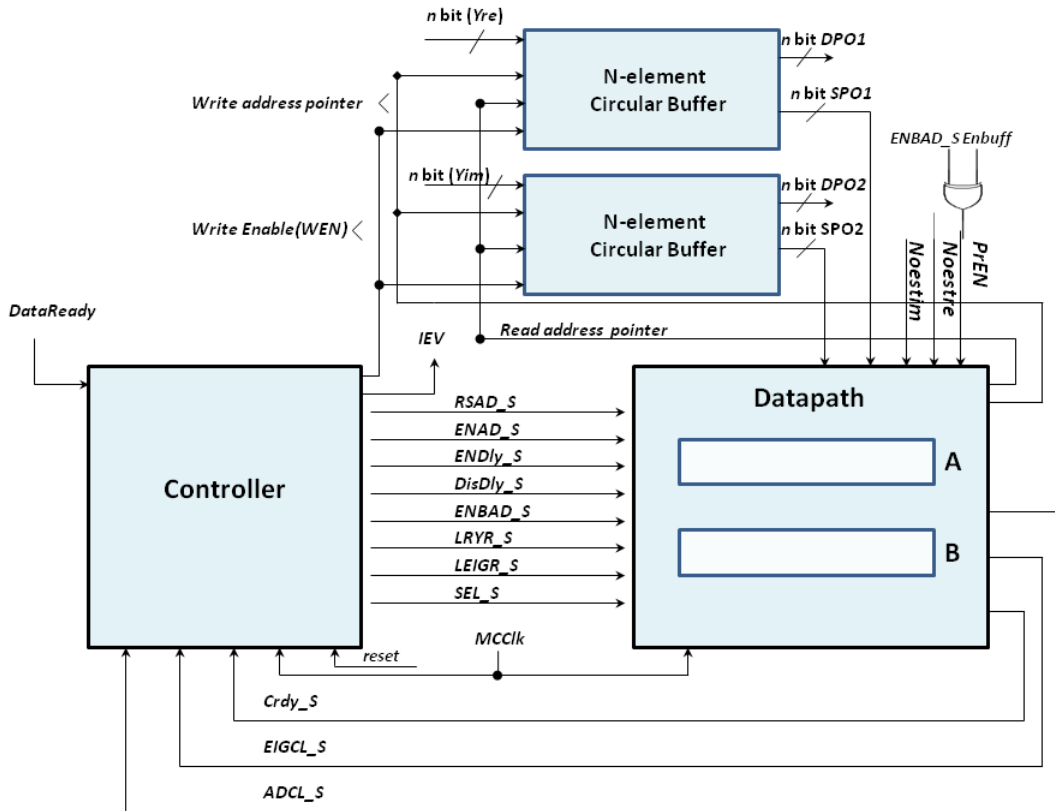


Figure 4.11: Illustration of block diagram for VHDL design of the second phase of the interference separation system.

4.5.2 Table of Signals

The external control input/output signals, external data output signals and registers are indicated in Figure 4.12.

Symbol	Function	Type
<i>RESET</i>	Initialize registers to 0	Control input
<i>DataReady</i>	Indication of arrival of the transformed received samples	Control input
<i>WEN</i>	Enable the <i>write</i> input of the memory unit	Control output
<i>IEV</i>	Indication that interference event is detected	Control output
<i>ADR</i>	BCD counter	6-Bit register
<i>DR</i>	BCD counter	7-Bit register
<i>RYR</i> (2x2 matrix each entry is 11-bit register)	Parallel load register	11-Bit register
<i>EIGR</i>	Parallel load register	13-Bit register

Figure 4.12: External control signals, data output signals and registers in VHDL design of the second phase of the interference separation system.

Action or Status	Control or Status Signals	Meaning for Values
$ADR \longleftarrow 0$ $ADR \longleftarrow ADR + 1$	<i>RSAD_S</i> <i>ENAD_S</i>	1 : Reset <i>ADR</i> to 0, 0 : No reset of <i>ADR</i> 1 : BCD count up <i>ADR</i> by 1, 0 : Hold <i>ADR</i> value
$RYR \longleftarrow RYR_{next}$ $EIGR \longleftarrow EIGR_{next}$	<i>LRYS_S</i> <i>LEIGR_S</i>	1 : Enable load <i>RYR</i> register, 0 : Disable load <i>RYR</i> register 1 : Enable load <i>EIGR</i> register, 0 : Disable load <i>EIGR</i> register
$DR \longleftarrow 0$ $DR \longleftarrow DR + 1$	<i>DisDly_S</i> <i>ENDly_S</i>	1 : Reset <i>DR</i> to 0, 0 : No reset of <i>DR</i> 1 : BCD count up <i>DR</i> by 1, 0 : Hold <i>DR</i> value
$EIGR \leq threshold$ $EIGR > threshold$	<i>EIGCL_S</i>	0 : <i>EIGR</i> is less than or equal to given threshold value 1 : <i>EIGR</i> is greater than given threshold value
$DR = N, DR < N$	<i>CRDY_S</i>	1 : <i>DR</i> is equal to predefined N value, 0 : <i>DR</i> is less than predefined N value
$ADR < s.size - 1, ADR = s.size - 1, ADR = s.size$ or $ADR = s.size + 1, o.w.$	<i>ADCL_S</i>	00 : <i>ADR</i> is less than given sample size – 1, 01 : <i>ADR</i> is equal to sample size-1, 10 : <i>ADR</i> is equal to sample size or sample size+1, 11 : Otherwise

Figure 4.13: Control and status signals in VHDL design of the second phase of the interference separation system.

4.5.3 Datapath Design

In VHDL design, adaptive algorithm (4.9) may be considered in two phases:

- Initialization of the adaptive algorithm: Initialization must be done at No-Interference Phase. Initialization is done parallel with the estimation of first users channel. In (4.4), we ignore the sample mean due to its small value (4.5) to reduce the algorithm complexity.
- Running Estimate of Covariance: (4.9) is updated as we receive new $y_1(k)$, $y_2(k)$ samples. The datapath is shown in Figure 4.14. First multiplexer selects between the output of one of the (4.4) and (4.9) at each iteration of combinational circuit. Controller unit decides the selection input of this multiplexer. Output of the first multiplexer is divided by N if first multiplexer selects (4.4) output. When the next state logic is ready, datapath send a status to the controller unit. Then controller assert load signals to register RYY .

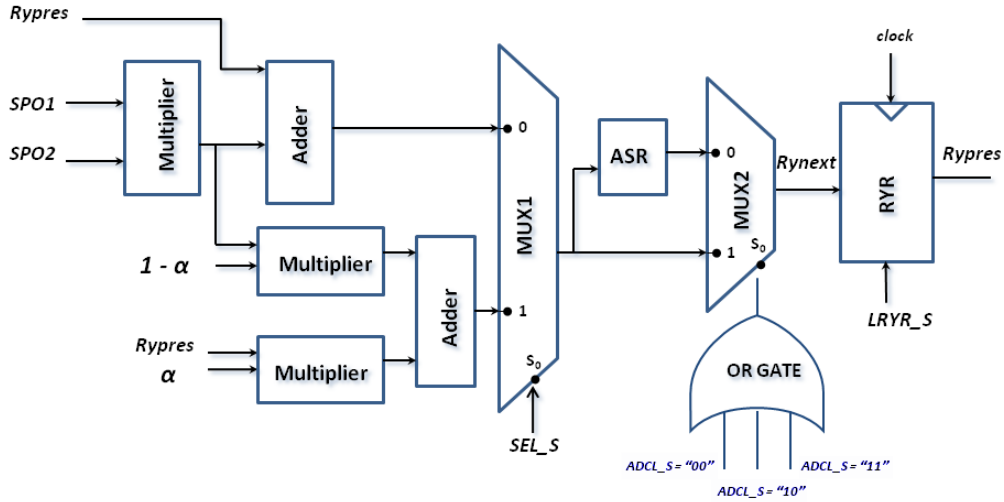


Figure 4.14: Circuit blocks for covariance estimation in VHDL design of second phase of the interference separation system.

Additionally, registers are required to keep last founded eigenvalues to compare next iteration values. If the eigenvalue ratio is bigger than the specified threshold value, a status signal is send to controller to raise a flag that indicates of interference event.

Figure 4.10 shows that estimated average index locations deviate from true index location for various SNR and SIR levels. According to this deviation, some symbols may be missed. Missing symbols can be recovered by using circular buffers. As be seen from Figure 4.15, an address counter determines the write and read pointer inputs of the circular buffer. Address counter increments the addresses of the pointers when Pr_EN signal is active at the load input of the counter. Pr_EN is activated in two situations. Firstly, to write and read $y_1(k)$, $y_2(k)$ values for algorithm purposes. Secondly, to read data to input buffer block which will be discussed in Chapter 5. The two situations are never occur at the same time. Pr_EN signal is the output of XOR gate as shown in Figure 4.11. Structure of the circular buffer is exploited to recover missing symbols. Read pointer follows the write pointer from one index behind. Therefore, read pointer always addresses the location with the oldest data stored in circular buffer. With an external status signal, reading of the content of the buffer provided successively.

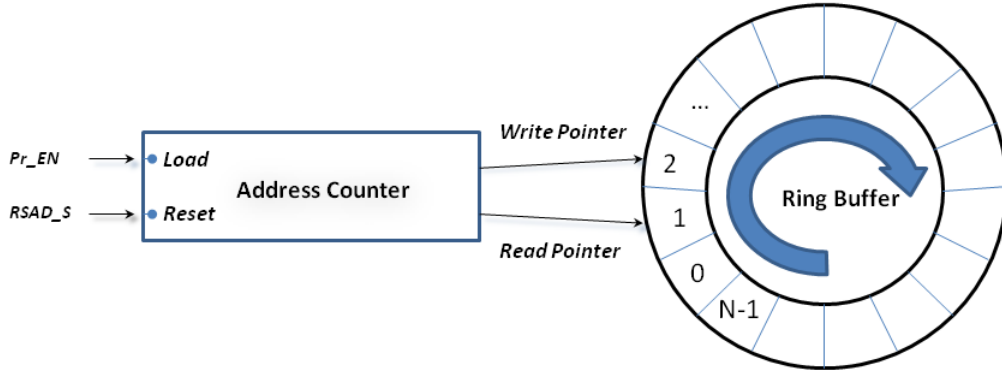


Figure 4.15: Utilization of circular buffer in VHDL design of second phase of the interference separation system.

4.5.4 Controller Design

The finite state machine diagram is given in Figure 4.16.

- **S1:** Just after asynchronous *RESET*, controller is in state S1 in which the address register *ADR* and delay register *DR* is reset to 0.
- **S2:** Circuit proceeds to S2 to wait for *DataReady* = '1'. As long as *DataReady* = '0', the state remains at S2. *DataReady* = '1' causes a transition to state S3.
- **S3:** $y_1(k)$, $y_2(k)$ samples are ready to process while controller is in this state. Then *WEN* are toggled to 1 to store the samples at k'th address location in circular buffers. If initialization is not ready circuit proceeds to S4, otherwise proceeds to S5.
- **S4:** First multiplexer in the datapath unit selects the output of initialization phase. *RYR* register is loaded at next state. However, propagation delay in Figure 4.14 must be evaluated. Combinational circuit must give stable outputs before controller enables the load registers. Likewise in first phase of the interference separation system, *DR* register counts up until outputs of the combinational circuits are stable. Then *CRDY_S* signal is sent by datapath unit for controller to proceed S6.
- **S5:** Difference of S5 from S4 is first multiplexer selects (4.9) outputs. When *CRDY_S* is enabled, circuit proceeds to S6.
- **S6:** *LRYR* is loaded. *ADR* is enabled to count up by 1 to reserve next address location for future samples. Address counter is checked again. If we are at initialization phase, circuit proceeds to S2 to wait for new samples. Otherwise it continues to S7.

- **S7:** Eigenvalues of $\hat{\mathbf{R}}_v(k)$ are calculated. When $CRDY_S$ becomes '1' circuit proceeds to S8.
- **S8:** Eigenvalue ratios are loaded to registers and compared to predefined threshold values. If threshold value is bigger, $EIGCL_S$ status '0' and next state is determined as S2. Otherwise, we can say that an interference event is detected and next state is determined as S9.
- **S9:** If circuit reaches S9, it indicates interference event is detected. Then IEV flag is toggled to '1' to alarm BCA block.

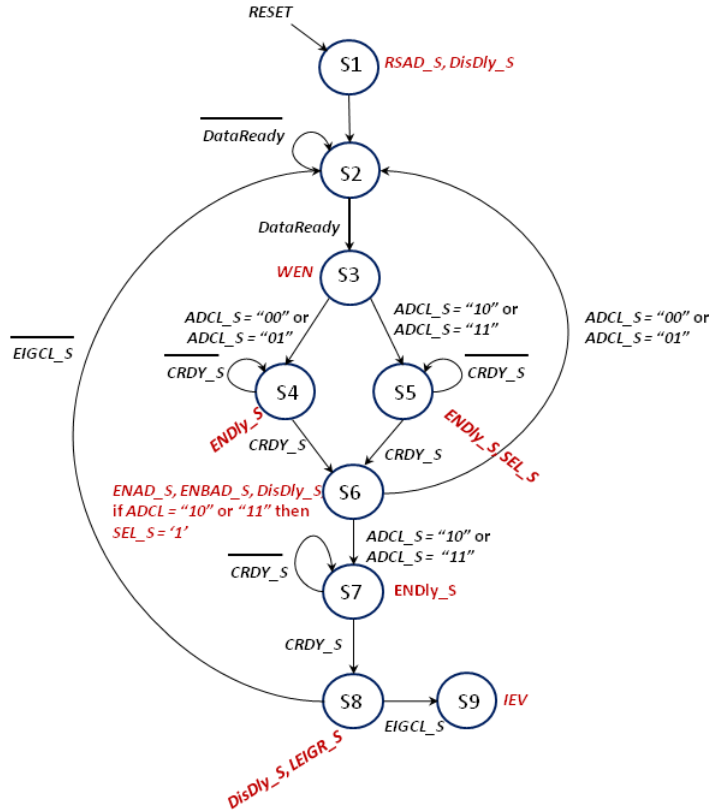


Figure 4.16: Illustration of state diagram for VHDL controller design in the second phase of the interference separation system.

4.6 Conclusion

In this chapter, we introduced a running estimate of covariance matrix corresponding to the data model for interference event detection problem. We proposed a detection method based on the eigenvalue disparity of covariance matrix. Specifically, eigenvalue ratio is presented as a best approach which has sharper behaviour around interference entrance point and more immune to lags due to slower variations in estimate algorithm. We illustrate the performance of eigenvalue ratio based interference detection algorithm for various SNR and SIR values. We noted that missing probability of the event and false alarm probability are affected by noise and interference floor. We showed that if the noise floor is sufficiently low, the interference event miss probability is relatively low even in low interference levels. We finally, investigated the estimated average index locations for various SIR and SNR values.

CHAPTER 5

Source Separation Algorithm

5.1 BCA Setup

BCA setup that was adapted to the scenario is summarized in Figure 5.1.

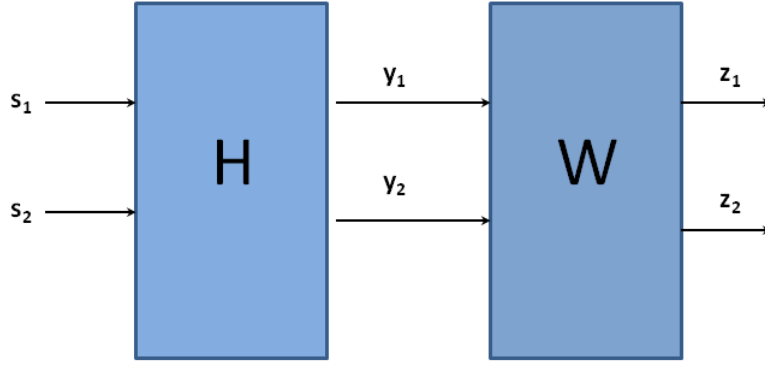


Figure 5.1: Blind source separation setup.

Indexing: i is used for (source, output) vector components, k is the sample index and t is the algorithm iteration index.

- We consider a setup consisting of 2 real sources which are represented by the vector $\mathbf{s} = [s_1 \ s_2]^T$. We assume that the sources are bounded.
- The sources are mixed by 2×2 mixing channel. Mixing channel is a linear and memoryless system which is represented by the matrix $\mathbf{H} \in \mathbb{R}^{2 \times 2}$.

- The mixtures are represented with $\mathbf{y} = [y_1 \ y_2]^T$ where the sources and the mixtures are related by $\mathbf{y} = \mathbf{H}\mathbf{s}$.
- $\mathbf{W} \in \mathbb{R}^{2 \times 2}$ is the separator matrix of the system which produces the outputs as $\mathbf{z} = \mathbf{W}\mathbf{y}$.
- The overall mapping from sources to separator is defined as $\mathbf{G} = \mathbf{W}\mathbf{H}$ where $\mathbf{G} \in \mathbb{R}^{2 \times 2}$. Therefore, the relation between the sources and the outputs can be written as $\mathbf{z} = \mathbf{G}\mathbf{s}$.
- $Y = \{\mathbf{y}(1), \mathbf{y}(2), \dots, \mathbf{y}(L)\}$ is the finite set consisting of observations of mixture samples. The main goal in BCA problem is to adapt the separator system based on these observations. We denote the corresponding set of source samples by $S = \{\mathbf{s}(1), \mathbf{s}(2), \dots, \mathbf{s}(L)\}$.

5.2 BCA Approach [43]

In this chapter, an essential summary of the deterministic instantaneous BCA approach introduced in [43] is provided. BCA framework based on two geometric objects:

- **Principal Hyperellipsoid** is the hyperellipse whose principal semi-axis directions are determined by the eigenvectors of the covariance matrix and whose principal semi-axis lengths are equal to principal standard deviations.
- **Bounding Hyperrectangle** corresponds to the box defined by the Cartesian product of the support sets of the individual components. This can be also defined as the minimum volume box containing all samples and aligning with the coordinate axes.

An example, for a case of 3-sources is provided in Figure 5.2. In this figure, a realization of separator output samples and the corresponding bounding hyper-rectangle and principal hyper-ellipsoid are shown. The approach is presented as maximization of the volume ratios of these two geometric objects.

The principal hyperellipsoid in the output domain is the image of the principal hyper-ellipsoid in the source domain. Transformation from source domain to output domain is determined by $\mathbf{G}$ matrix. We note that the instantaneous overall mapping causes $|\det(\mathbf{G})|$ scaling in the volume of principal hyperellipsoid. However, the image of the bounding hyperrectangle in the source domain is a hyperparallelopiped which is a subset of the bounding hyperrectangle in the output domain. It is proved that image of the bounding hyperrectangle is another hyperrectangle if $\mathbf{G}$ is referred as perfect separator. Then, the scaling in the volume of hyperrectangle depends on this criteria. $\mathbf{G}$ can be written as $\mathbf{G} = \mathbf{D}\mathbf{P}$, where $\mathbf{D}$ is a diagonal matrix with non-zero diagonal entries, and $\mathbf{P}$ is a permutation matrix. Optimization setting for that scenario is proposed as $\hat{\mathbf{R}}_{\mathbf{D}}$ represents the sample covariance matrix of the output samples in the set D and $\hat{\mathcal{R}}(D)$ includes the range values of the components of the output samples in the set D .

$$\text{maximize} \quad J_1(\mathbf{W}) = \frac{\sqrt{\det(\hat{\mathbf{R}}_{\mathbf{D}})}}{\prod \hat{\mathcal{R}}(D)}. \quad (5.1)$$

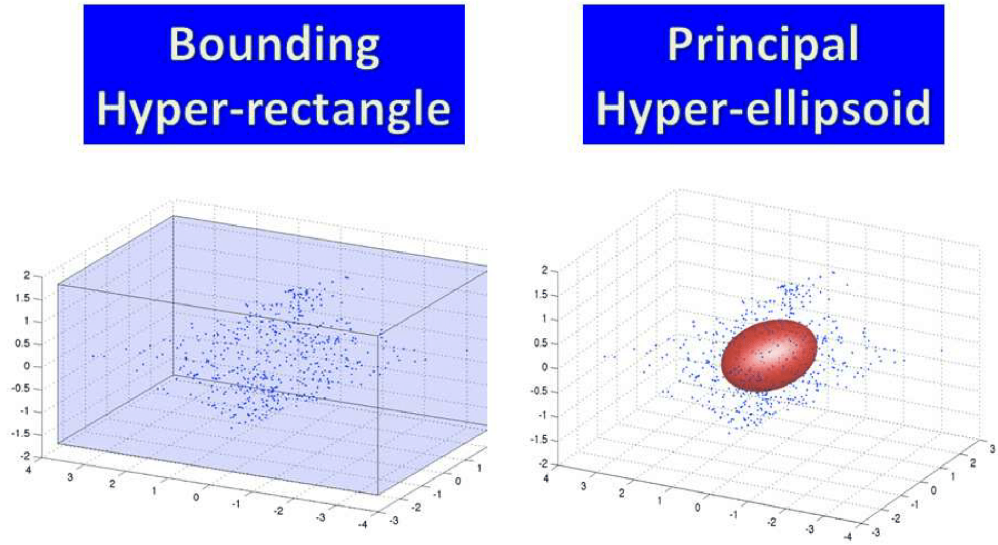


Figure 5.2: Bounding Hyper-rectangle and Principal Hyper-ellipsoid.

5.3 Adaptive Algorithms

In this section, we derive iterative adaptive algorithms to solve the corresponding optimization problems.

There are alternative BCA objective functions and the corresponding algorithm update rules:

- $J_1(\mathbf{W}) = \frac{\sqrt{\det(\hat{\mathbf{R}}_{\mathbf{z}})}}{\mathcal{R}(z_1)\mathcal{R}(z_2)}$ objective function, where

$$\hat{\mathbf{R}}_{\mathbf{z}} = \frac{1}{L} \sum_{k=1}^L \mathbf{z}_k \mathbf{z}_k^T \quad (5.2)$$

is the sample covariance matrix and $\mathcal{R}(\cdot)$ is the range operator which is the difference between the maximum and the minimum values of its argument. The adaptive algorithm corresponding to this objective function is given by

$$\begin{aligned} \mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} + \mu^{(t)} & \left(\mathbf{R}_z^{(t)-1} \mathbf{W}^{(t)} \mathbf{R}_y \sum_{i=1}^2 \right. \\ & \left. \frac{1}{\mathcal{R}(z_i^{(t)})} (\mathbf{y}(k_i^{(t)}, max) - \mathbf{y}(k_i^{(t)}, min))^T \right) \end{aligned} \quad (5.3)$$

- $J_{2,p} = \frac{\sqrt{\det(\hat{\mathbf{R}}_{\mathbf{z}})}}{(\mathcal{R}(z_1)^p + \mathcal{R}(z_2)^p)^{2/p}}$ is a family of objective functions for different $p > 1$ and the corresponding update rule for this family is given by

$$\begin{aligned} \mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} + \mu^{(t)} & \left(\mathbf{R}_z^{(t)-1} \mathbf{W}^{(t)} \mathbf{R}_y - \frac{n}{\|\mathcal{R}(\mathbf{z}^{(t)})\|_p} \sum_{i=1}^2 \right. \\ & (|z_i^{(t)}(k_i^{(t)}, max)|^{p-1} \mathbf{y}(k_i^{(t)}, max) - |z_i^{(t)}(k_i^{(t)}, min)|^{p-1} \\ & \left. \mathbf{y}(k_i^{(t)}, min))^T \right) \end{aligned} \quad (5.4)$$

where

- $\mathbf{W}^{(t)}$ is the separator matrix at the t^{th} iteration,
- $\mu^{(t)}$ is the step size at the t^{th} iteration,
- $(k_i^{(t)}, max)$ is the set of indexes where the i^{th} separator output reaches its maximum value at the t^{th} iteration,
- $(k_i^{(t)}, min)$ is the set of indexes where the i^{th} separator output reaches its minimum value at the t^{th} iteration.

In our simulations we used $J_{2,1}$ objective function and the corresponding update rule.

5.4 Fixed Point Implementation Issues

In a real system power level of the analog signal must be sufficient to quantized by an analog to digital converter. Assuming that ADC is used, quantization process naturally introduces an quantization error. If analog signal levels considered at simulations are fed to the higher ranged ADC, quantization errors become compatible with our weak signals. It causes greater mistakes in quantization. The gap between the analog power levels corresponding to each of the different levels of ADC must be adequate enough to decrease the chance of error in the quantization. For that purpose, automatic gain control (AGC) is placed before the ADC.

AGC raises the volume of the signal which is proper for analog to digital converter. The average or peak output signal level is used to adjust the gain to a suitable level, enabling the circuit to work with a greater range of input signal levels. AGC effectively reduces the volume if the signal is strong and raises it when it is weaker. In our situation it raises the signal level. There must be some scaling operation on real numbers in MATLAB to adequately assign these numbers to the ADC. If there is an overflow, upper and lower points must be pulled to the range of the ADC. Scaling operation is considered according to peak to average ratio (PAR) of the signal which is targeted as 15 dB. Scaling parameter is defined in (5.5).

$$Scale = \frac{PS}{\hat{\sigma}_y} \quad (5.5)$$

Likewise the definitions in interference event detection section, where P is the target integer range corresponding to the number of bits b used in the variable representation, i.e. $P = 2^{b-1} - 1$ and S is for the adjustment factor to adjust peak to average ratio for the corresponding variable. We can write $S = 10^{-PAR/20}$ where PAR is the target peak

to average ratio in dB.

Variables in MATLAB algorithms are scaled to the greater level and represented as integer numbers. It must be compatible with real system with ADC. For simplicity, the index number k indicating the sample index is not put i.e. $\mathbf{y}$ indicates $\mathbf{y}(\mathbf{k})$ where k is the sample index.

- Integer representation of $\mathbf{y}$ is estimated as:

$$\mathbf{y}^I = \mathbf{y} * Scale \quad (5.6)$$

- Seperator matrix $\mathbf{W}$ which was defined as 2×2 identity matrix is scaled by $Scale$ and represented in a variable $\mathbf{W}^I$.

$$\mathbf{W}^I = \begin{bmatrix} Scale & 0 \\ 0 & Scale \end{bmatrix} \quad (5.7)$$

Problem can arise when there are multiplication of represented integer variables. Each variable represented as integer number is multiplication of the unscaled form of it and the scaling factor. When two integer number is multiplied, scaling factor is multiplied twice. It causes a problem because the result must be the integer representation of unscaled form of that variable.

- Integer representation of $\mathbf{Z}$ is defined as:

$$\mathbf{W}^I \mathbf{Y}^I = \mathbf{WY} * Scale * Scale \quad (5.8)$$

$$\mathbf{Z} = \mathbf{WY} \quad (5.9)$$

$$\mathbf{Z}^I = \mathbf{Z} * Scale \quad (5.10)$$

- When above equations are considered together $\mathbf{W}^I \mathbf{Y}^I$ must be divided to *Scale* and rounded to integer value to exactly represent the $\mathbf{Z}^I$.

Arithmetic right shift and division operations have the same duty due to the binary representation of the numbers. Dividing the number with *Scale* is equivalent to shifting the number right by *Scale*.

$$[\mathbf{Z}^I = \mathbf{W}^I \mathbf{Y}^I \gg \text{Scale}] \quad (5.11)$$

Covariance calculations also includes multiplication of integer variables.

- $\mathbf{R}_y$ must be divided with *Scale* parameter.

Scale parameter depends on the value of $\text{std}(\mathbf{Y})$ and change by each observation. It can be rounded to the closest integer level that is represented by binary bits. Additionally, overflow in integer multiplication is controlled.

$$\mathbf{R}_y = \frac{1}{L-1} * \sum_{i=1}^2 (y_i - \bar{y})(y_i - \bar{y})^T \quad (5.12)$$

- Shift operation on covariance of $\mathbf{Y}$

$$\mathbf{R}_y^I = [\mathbf{R}_y \gg \text{Scale}] \quad (5.13)$$

- Within the iterative part of the BCA algorithm, range parameter $\mathcal{R}(\cdot)$ is determined according to separator outputs difference of reach its maximum value and minimum value. When separator outputs is represented as integer variable $\mathbf{Z}^I$, range is automatically found as an integer value.
- When the variables at the above equation is replaced with their integer values, overflow is inevitable. For this reason, variables must be multiplied two at a time and scaled properly. Integer representation of $\mathbf{R}_z$ is calculated in (5.14).

$$\mathbf{R}_z^{\mathbf{I}} = \lfloor \lfloor \mathbf{W}^{\mathbf{I}} \mathbf{R}_y^{\mathbf{I}} \gg Scale \rfloor \mathbf{W}^{\mathbf{I}^T} \gg Scale \rfloor \quad (5.14)$$

- There is a simplification in the algorithm given by

$$inv(\mathbf{R}_z^{\mathbf{I}}) \mathbf{W}^{\mathbf{I}} \mathbf{R}_y^{\mathbf{I}} = inv(\mathbf{W}^{\mathbf{I}^T}) \quad (5.15)$$

Therefore integer representation of $inv(\mathbf{R}_z^{\mathbf{I}})$ is not calculated.

- Remaining part of the update equation is modified to simplify hardware realization.

$$\mathbf{W}_{inv}^{\mathbf{I}} = inv(\mathbf{W}^{\mathbf{I}^T}); \quad (5.16)$$

$$Updint = \lfloor (Rangeint * \mathbf{W}_{inv}^{\mathbf{I}}) \gg Scale \rfloor - \lfloor (2 * \mathbf{O} * det(\mathbf{W}^{\mathbf{I}})) \gg Scale \rfloor \quad (5.17)$$

- Definition of *Rangeint* and $\mathbf{O}$ matrix will be explained in VHDL design part.

Representation of step size requires two steps:

- Norm value must be represented as integer number.
- According to iteration number, numerator of the stepsize equation must be modeled as fixed-point number as in Figure 5.3.

$$normint = (norm(Updint, 'fro')) \quad (5.18)$$

$$normint = \lfloor (normint \gg shift) / (Scale \gg shift) \rfloor \quad (5.19)$$

where shift is the integer after the first '1' bit in binary form of *normint* variable.

- Value at each interval determines the convergence step of the algorithm.

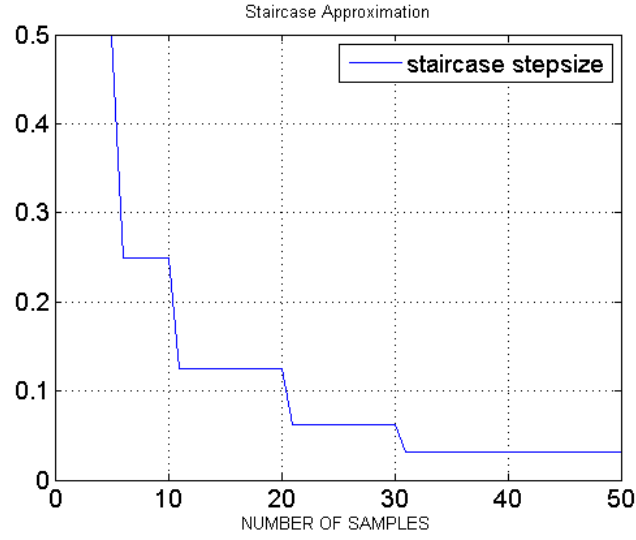


Figure 5.3: Convergence steps as a function of number of samples in fixed-point algorithm.

- According to represented numbers, fixed-point algorithm for BCA provides similar convergence characteristic with floating-point algorithm.

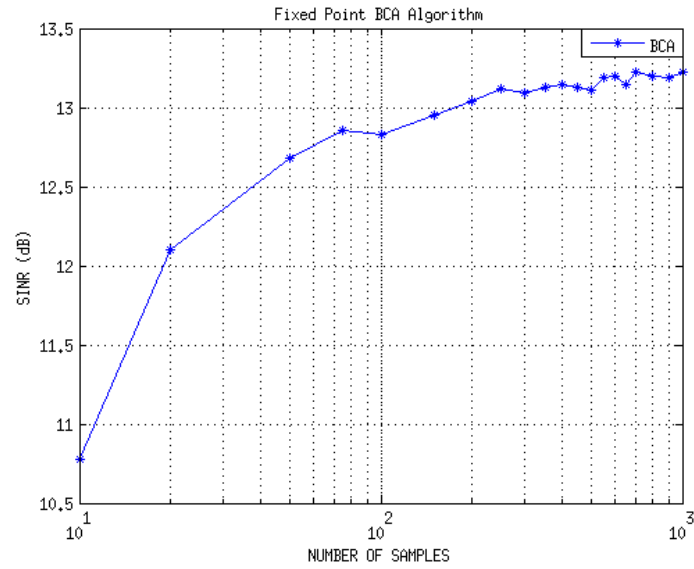


Figure 5.4: SINR performance of fixed-point BCA algorithm as a function of number of samples.

5.5 Simulation Results

Performance of the algorithm is tested for different (Source to Interference Ratio) SIR scenarios. SIR shows the total signal energy to total interference energy ratio for BCA algorithm. It is assumed that desired signal(*source_1*) power to noise power is 15 dB. By fixing the desired signal power, interference power adjusted at different values relative to desired signal power. Each scenario is simulated according to $L = 1000$ and $L = 50$. With these setup, adaptive algorithm in (5.4) is performed. $L = 50$ case provides closer results to $L = 1000$ case. Therefore, $L = 50$ is applied as number of observations.

- For SIR=-8dB case SIR graph is shown in Figure 5.5.

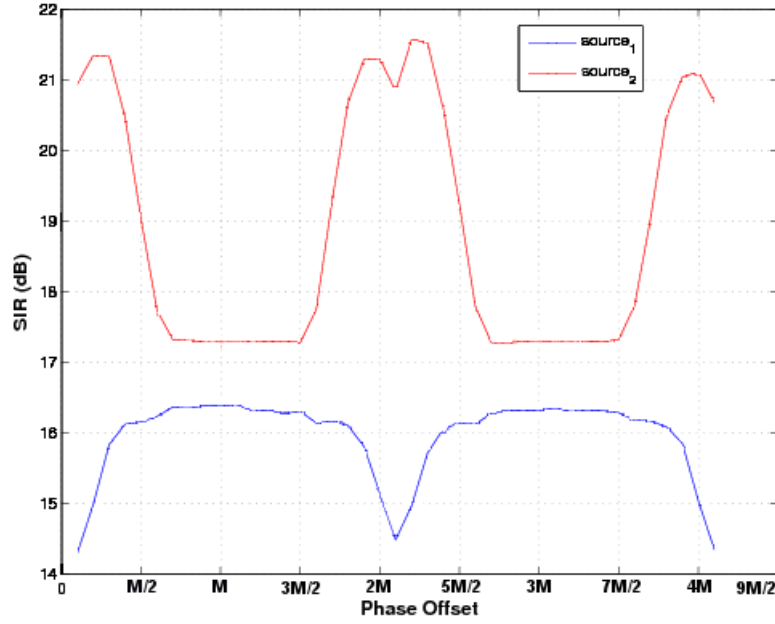


Figure 5.5: SIR as a function of phase offset for SIR = -8dB and $L = 50$ case.

- For $SIR = -4\text{dB}$ case SIR graph is shown in Figure 5.6.

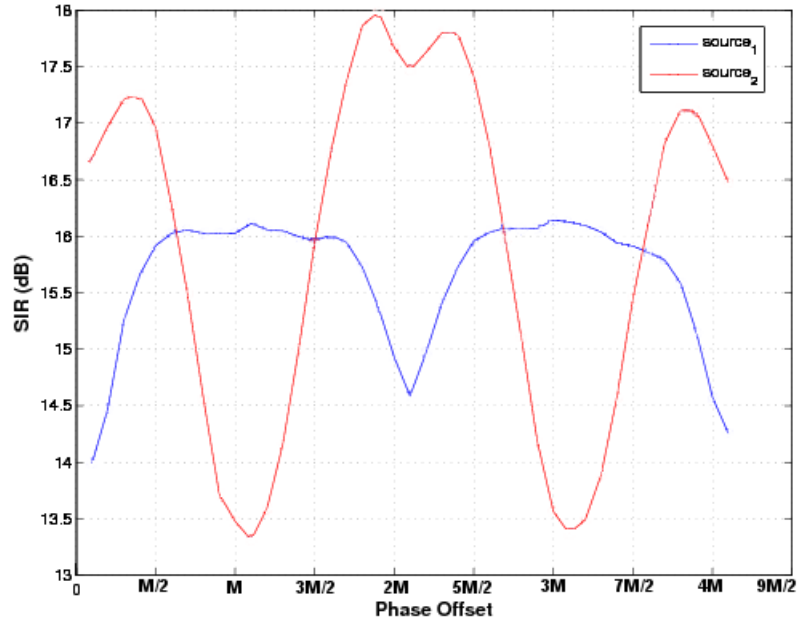


Figure 5.6: SIR as a function of phase offset for $SIR = -4\text{dB}$ and $L = 50$ case.

- For $SIR = -2\text{dB}$ case SIR graph is shown in Figure 5.7.

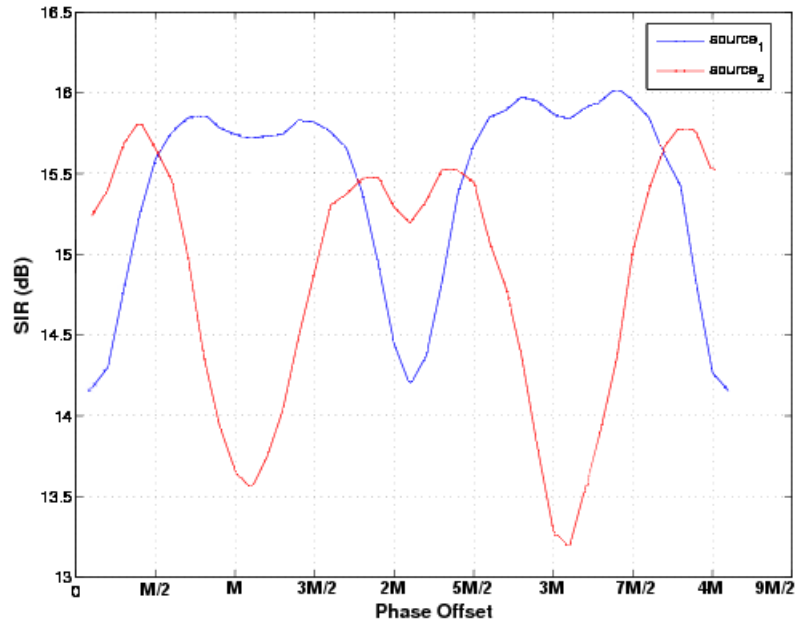


Figure 5.7: SIR as a function of phase offset for $SIR = -2\text{dB}$ and $L = 50$ case.

- For $SIR=0\text{dB}$ case SIR graph is shown in Figure 5.8.

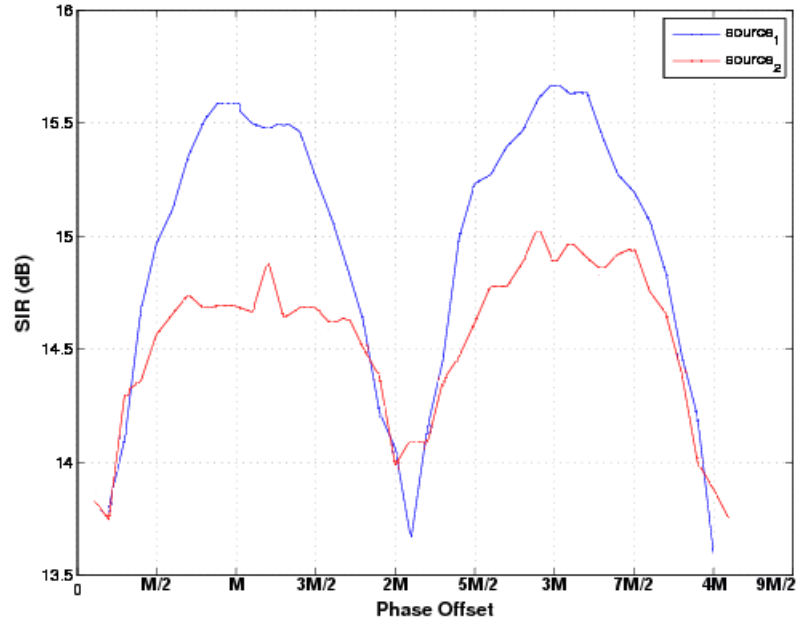


Figure 5.8: SIR as a function of phase offset for $SIR = 0\text{dB}$ and $L = 50$ case.

- For $SIR=2\text{dB}$ case SIR graph is shown in Figure 5.9.

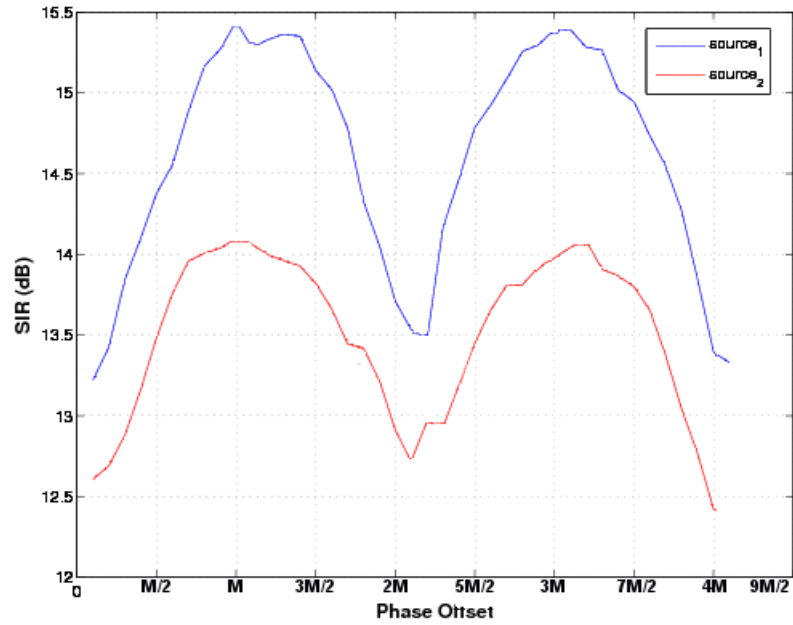


Figure 5.9: SIR as a function of phase offset for $SIR = 2\text{dB}$ and $L = 50$ case.

- For SIR=4dB case SIR graph is shown in Figure 5.10.

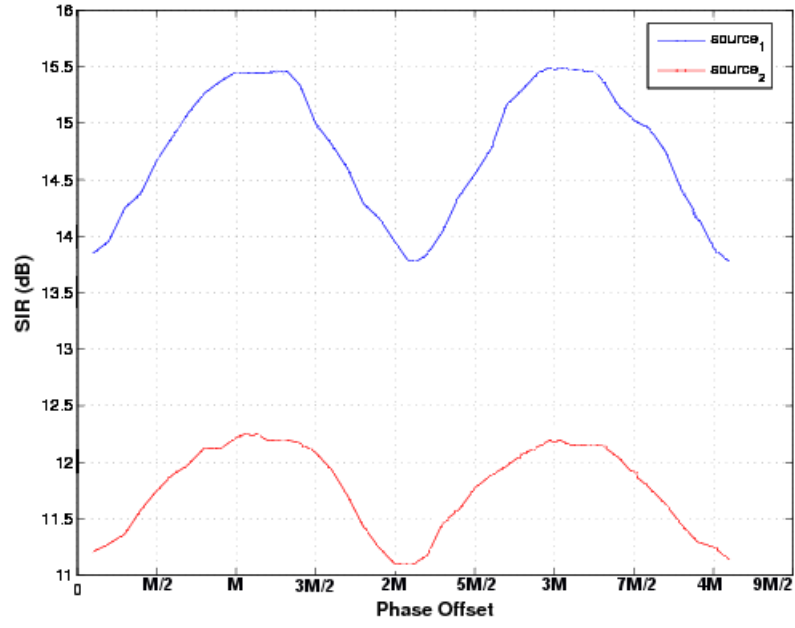


Figure 5.10: SIR as a function of phase offset for SIR = 4dB and L = 50 case.

- For SIR=8dB case SIR graph is shown in Figure 5.11.

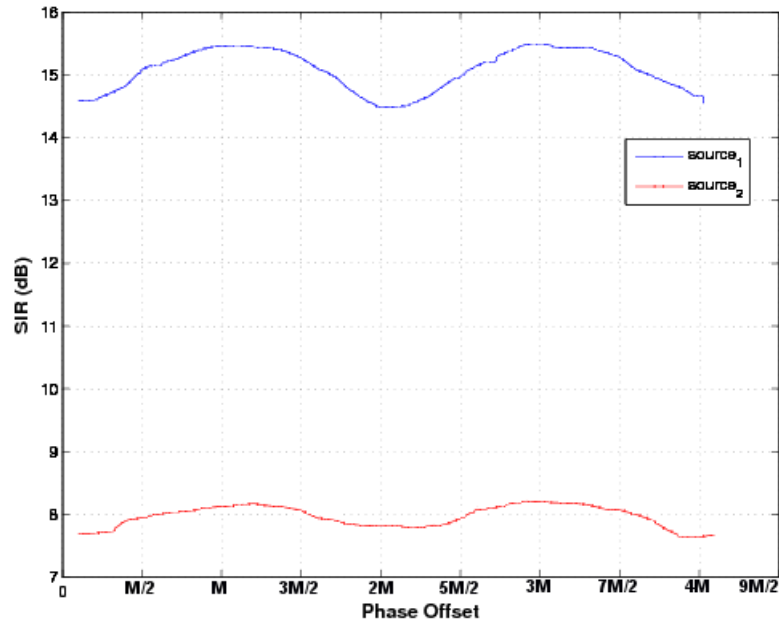


Figure 5.11: SIR as a function of phase offset for SIR = 8dB and L = 50 case.

5.5.1 Algorithm Convergence

In order to provide sense of convergence of the BCA algorithm, Figure 5.12 illustrates typical convergence curves for both outputs (for SIR=0dB case and 50 samples. According to these curves convergence occur before about 50th iteration.

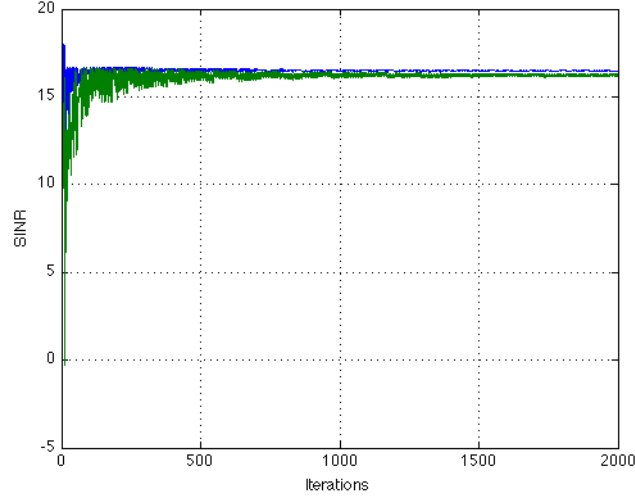


Figure 5.12: Convergence of BCA algorithm as a function of iterations.

5.5.2 Comparison of BCA with state of algorithms

- SNR is selected both 15 dB and 30 dB to compare SNR performances of BCA algorithm to the other adaptive algorithms.
- Iteration number is kept constant at $L = 50$. It is a sufficient number for algorithm to converge.

Power of the main signal and the interference signal is same at the receiver in Figure 5.13 and Figure 5.14. As can be seen from figures, BCA algorithm is optimum among the algorithms with relatively small number of samples. Increasing the number of samples

cannot make big fluctuation on the SINR curve for BCA algorithm especially at 30 dB SNR.

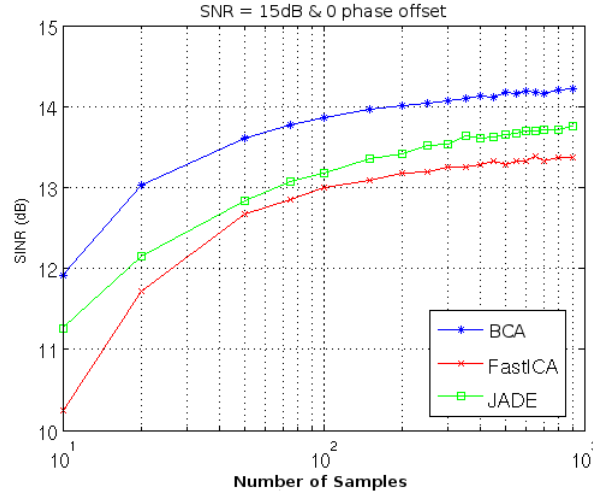


Figure 5.13: Signal to Interference-Noise Ratio as a function of number of samples for SNR=15dB and SIR=0dB.

- Performance of BCA is more effective for relatively small set of observations when SNR is 30 dB.

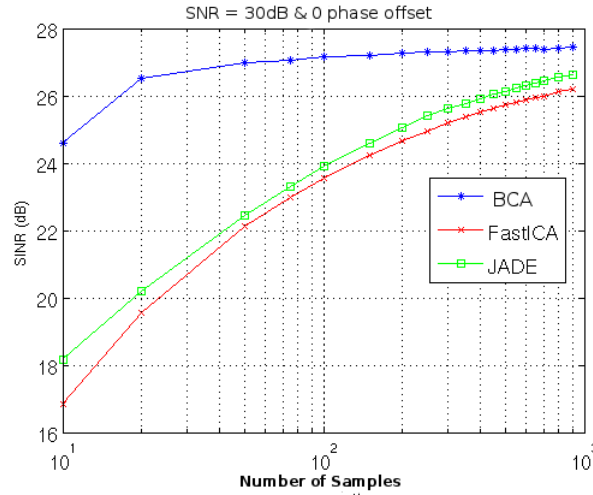


Figure 5.14: Signal to Interference-Noise Ratio as a function of number of samples for SNR=30dB and SIR=0dB.

- Power of the main signal is 6dB below the interference signal at the receiver in Figure 5.15.

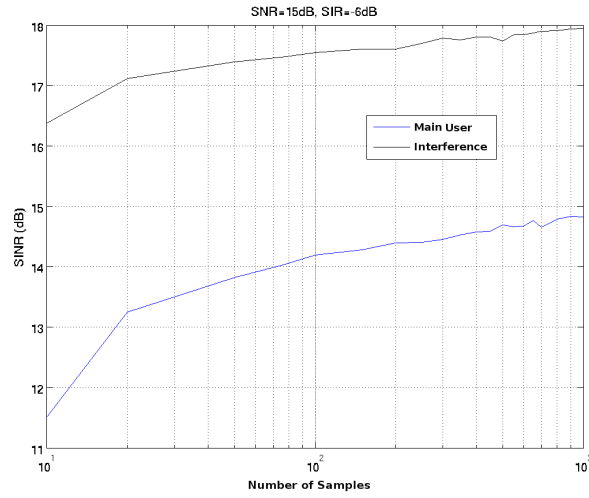


Figure 5.15: Signal to Interference-Noise Ratio as a function of number of samples for SNR=15dB and SIR=-6dB.

- Power of the main signal is 6dB above the interference signal at the receiver in Figure 5.16.

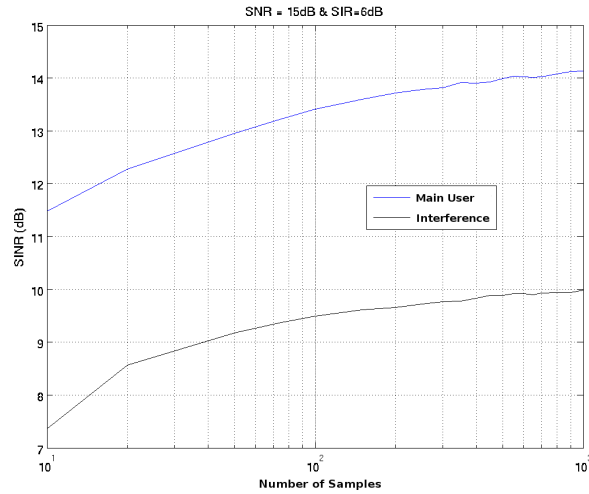


Figure 5.16: Signal to Interference-Noise Ratio as a function of number of samples for SNR=15dB and SIR=6dB.

5.6 VHDL Implementation Issues

5.6.1 Design Specifications

In this phase the goal is to design RTL to implement adaptive algorithm stated in (5.4). Once the interference event is detected the third block which is responsible for separating interference from the first user is activated. A finite set consisting of L observations of mixture samples are required to adapt the separator system based on these observations. Then, symbol-spaced samples $y_1(k)$ and $y_2(k)$ are received and stored for training purposes. Adaptive algorithm use same set of observations at each iteration. Like in the first phase, we store all the training data, before permission signals for operation is send to controller.

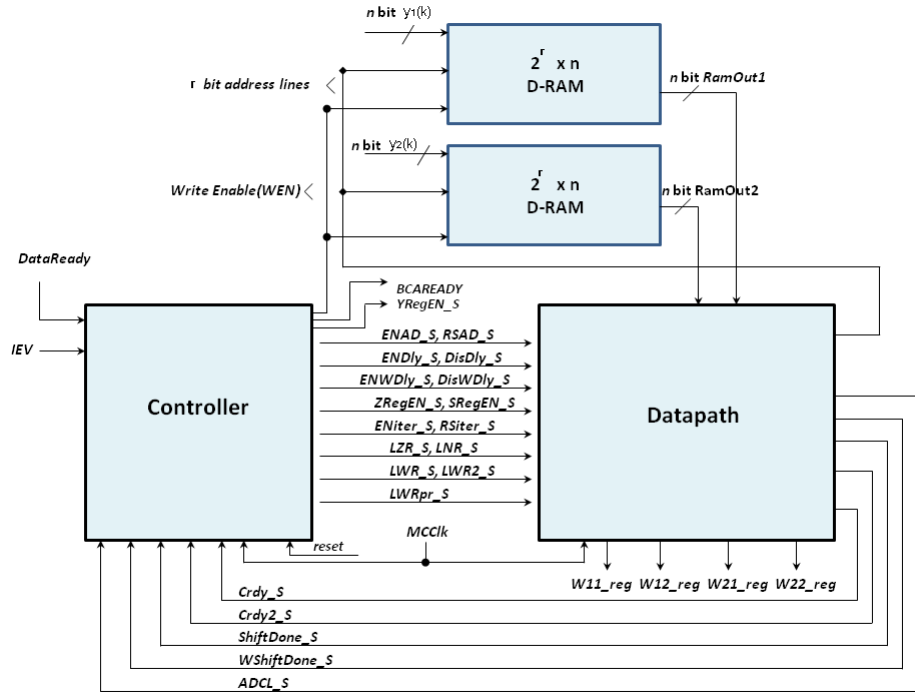


Figure 5.17: Illustration of block diagram for VHDL design of third phase of the interference separation system.

5.6.2 Table of Signals

The external control input/output signals, external data output signals and registers are indicated in Figure 5.18. Functionality of the registers are discussed in detail on state diagram.

Symbol	Function	Type
<i>RESET</i>	Initialize registers to 0	Control input
<i>DataReady</i>	Indication of arrival of the transformed received samples	Control input
<i>IEV</i>	Indication that interference event is detected	Control input
<i>WEN</i>	Enable the <i>write</i> input of the memory unit	Control output
<i>BCAREADY</i>	Indication that training of the separation algorithm is finished	Control output
<i>ADR</i>	BCD counter	6-Bit register
<i>DR,DWR</i>	BCD counter	7-Bit register
<i>ITR</i>	BCD counter	6- Bit register
<i>YR, ZR</i>	Parallel load register	16-Bit register
<i>SR, NR, RZR</i>	Parallel load register	16- Bit register
<i>LWR, LWR2, LWRpr</i>	Parallel load register	11-Bit register

Figure 5.18: External control signals, data output signals and registers in VHDL design of the third phase of the interference separation system.

Figure 5.19 and 5.20 shows these signals with their meanings and which registers they alter.

Action or Status	Control or Status Signals	Meaning for Values
$ADR \leftarrow 0$ $ADR \leftarrow ADR + 1$	$RSAD_S$ $ENAD_S$	1: Reset ADR to 0, 0: No reset of ADR 1: BCD count up ADR by 1, 0: Hold ADR value
$ITR \leftarrow 0$ $ITR \leftarrow ITR + 1$	$RSiter_S$ $ENiter_S$	1: Reset ITR to 0, 0: No reset of ITR 1: BCD count up ITR by 1, 0: Hold ITR value
$YR \leftarrow YR_{next}$ $ZR \leftarrow ZR_{next}$	$YRegEn_S$ $ZRegEn_S$	1: Enable load YR register, 0: Disable load YR register 1: Enable load ZR register, 0: Disable load ZR register
$SR \leftarrow SR_{next}$ $NR \leftarrow NR_{next}$	$SRegEn_S$ LNR_S	1: Enable load SR register, 0: Disable load SR register 1: Enable load NR register, 0: Disable load NR register
$RZR \leftarrow RZR_{next}$	LZR_S	1: Enable load RZR register, 0: Disable load RZR register
$DR \leftarrow 0$ $DR \leftarrow DR + 1$	$DisDly_S$ $ENDly_S$	1: Reset DR to 0, 0: No reset of DR 1: BCD count up DR by 1, 0: Hold DR value

Figure 5.19: Control and status signals in VHDL design of the third phase of the interference separation system.

Action or Status	Control or Status Signals	Meaning for Values
$DWR \leftarrow 0$ $DWR \leftarrow DWR + 1$	$DisWDly_S$ $ENWDly_S$	1: Reset DWR to 0, 0: No reset of DWR 1: BCD count up DWR by 1, 0: Hold DWR value
$DR = N$, $DR < N$	$CRDY_S$	1: DR is equal to predefined N value, 0: DR is less than predefined N value
$DWR = M$, $DWR < M$	$CRDY2_S$	1: DWR is equal to predefined M value, 0: DWR is less than predefined M value
$ITR < s.size - 1$ and $(ITR \bmod 10 \neq 0)$, $ITR < s.size - 1$ and $(ITR \bmod 10 = 0)$, otherwise	$ITCL_S$	00: ITR is less than given sample size – 1 and $\bmod 10$ of ITR is not equal to 0. 01: ITR is less than given sample size – 1 and $\bmod 10$ of ITR is equal to 0. 10: Otherwise
$ADR < s.size - 1$, $ADR = s.size - 1$, $ADR = s.size$ or $ADR = s.size + 1$, o.w.	$ADCL_S$	00: ADR is less than given sample size – 1, 01: ADR is equal to sample size-1, 10: ADR is equal to sample size or sample size+1, 11: Otherwise

Figure 5.20: Control and status signals in VHDL design of the third phase of the interference separation system.

5.6.3 Datapath Design

Adaptive algorithm stated in (5.4) is represented by design blocks in Figure 5.21. It is a complex process based on both sample index k and iteration index t . At each algorithm iteration, same set of observations are utilized.

$$D_{\mathbf{G}} = \{\mathbf{W}\mathbf{y}(1), \mathbf{W}\mathbf{y}(2), \dots, \mathbf{W}\mathbf{y}(L)\}$$

define the set of separator outputs where $\mathbf{z} \in D_{\mathbf{G}}$. L define the cardinality of the sample set. Sample index k is the index variable of received $y_1(k)$ and $y_2(k)$ samples. As defined in (5.3), separator output samples in 2-sources case is evaluated. In other words, each component of the separator outputs set $Z_{\mathbf{G}}$ consists of 2-sources which corresponds to 2 different separator output samples. The samples are corresponding to the k^{th} samples of $\mathbf{z}_1$ and $\mathbf{z}_2$ vectors. Range operator $\mathcal{R}(\cdot)$ takes the the difference of the maximum and the minimum values of each of these vectors. However, in the VHDL realization, separator output samples may not be stored in vectors. Instead at each k index of t iteration, $\mathbf{z}_{\mathbf{k}}^{(t)}$ and $\mathbf{z}_{\mathbf{k}}^{(t)}$ values are registered. Then, for the next k index, a circuit checks if the new separator output samples are the new maximum or minimum. If true, new values are registered. In other words, registers keep only the maximum and minimum values to update range operation. Process is repeated for each k index until operations are done with all of the samples.

Additionally, difference of $\mathbf{y}_{\mathbf{k}}^{(t)}$ vector values pointed by the set of indexes $(k_i^{(t)}, max)$ and $(k_i^{(t)}, min)$ are also registered for both sources. For that process, four registers are required. Register outputs correspond to the entries of the defined $\mathbf{O}^{(t)}$ output. Source index determines the row number of the difference vectors. At the end of the training set, range value R and $\mathbf{O}$ is fed to the update block as indicated in fixed-point section where some of the parameters are modified for simplification in hardware realization in BCA algorithm. Update block computes the (5.17) which requires bunch of multiplication

and addition circuits. Then frobenius norm of this parameter is computed.

According to iteration index and norm value, step size is set as figured in 5.4. Multiplication of step size with update block outputs requires division of these outputs by denominator of the step size value. Division operation is hard to implement. Instead, an alternative approach is applied. In this method, both the numerator and denominator are divided by 2(shifted right by one) at each clock cycle. Then this division value is registered. Registered value is compared to the constant vector in which only the least significant bit of the vector value is equal to '1'. Until denominator is equal to this vector value, numerator is shifted repeatedly. If both equal at some clock edge, a status signal *ShiftDone_S* is sent to controller to stop the division by disabling the registers.

All process repeats for each t iteration of BCA.

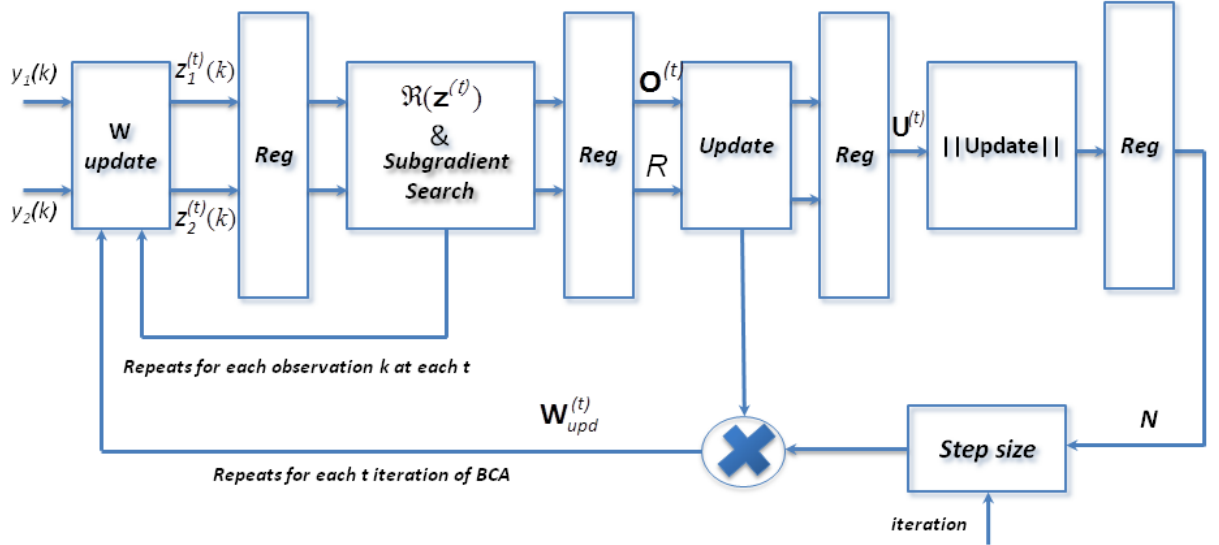


Figure 5.21: Circuit blocks for BCA algorithm.

5.6.4 Controller Design

- **S1:** Just after asynchronous *RESET*, controller is in state S1 in which the address register *ADR*, delay register *DR* and iteration register *ITR* is reset to 0.
- **S2:** Circuit proceeds to S2 if *IEV* is '1' to wait for *DataReady* = '1'.
- **S3:** $y_1(k)$ and $y_2(k)$ samples are registered.
- **S4:** *WEN* is toggled to '1' to store $y_1(k)$ and $y_2(k)$ to the k'th address location in two parallel distributed ram.
- **S5:** *ADR* is enabled to count up by 1 to reserve next address location for future sample. Circuit proceeds to S2 if number of training samples are not ready in ram blocks. A logic block controls the memory addresses after each write operation and status the *ADCLS* signal accordingly. If the memory pointer addresses the memory location with index smaller than the $N-1$, circuit proceeds to S2 to wait new sample. Otherwise it proceeds to S6.
- **S6:** This state is the indication that we finished to store components of the sample set in our memory blocks. Then, write address pointer returns to the first address location. Therefore, reset signal is asserted to clear the content of the *ADR* register. Then *ITR* register is enabled by *ENIS* signal to count the iteration number.
- **S7:** BCA algorithm starts. Controller waits for separator outputs $\mathbf{z}$ to compute. Then circuit proceeds to the S8.
- **S8:** Separator outputs are registered.
- **S9:** $\mathcal{R}(\cdot)$ and subgradient search block is processed.

- **S10:** Outputs of this block are registered. If block processed for each element of the observation set, controller proceeds to S11. Otherwise, proceeds to S7.
- **S11:** Each component of the update block is computed.
- **S12:** Outputs of the update block are registered.
- **S13:** Frobenius norm of these outputs are computed. Square root algorithm is utilized again.
- **S14:** Norm value is registered.
- **S15:** *ShiftDone_S* signal which is explained in Datapath section is required to proceed to S16.
- **S16:** According to the *ITCL_S* value which is the indication of iteration index, circuit proceeds to S17 or S18 or S22.
- **S17:** Indicates that $W^{(t+1)}$ is ready. Each entry of the updated separation matrix is registered in this state.
- **S18:** If circuit reaches this state, it means that mod 10 of the iteration index is 0.
- **S19:** In the case of these specific iteration index, BCA convergence is checked. Then, *WShiftDone_S* signal is asserted.
- **S20:** Updated separation matrix is registered.
- **S21:** Due to the differences of update rules depending on the iteration index, a master register for separation matrix is loaded.
- **S22:** If circuit reaches this state, it indicates the completion of last iteration. *BCARDY* signal is enabled.

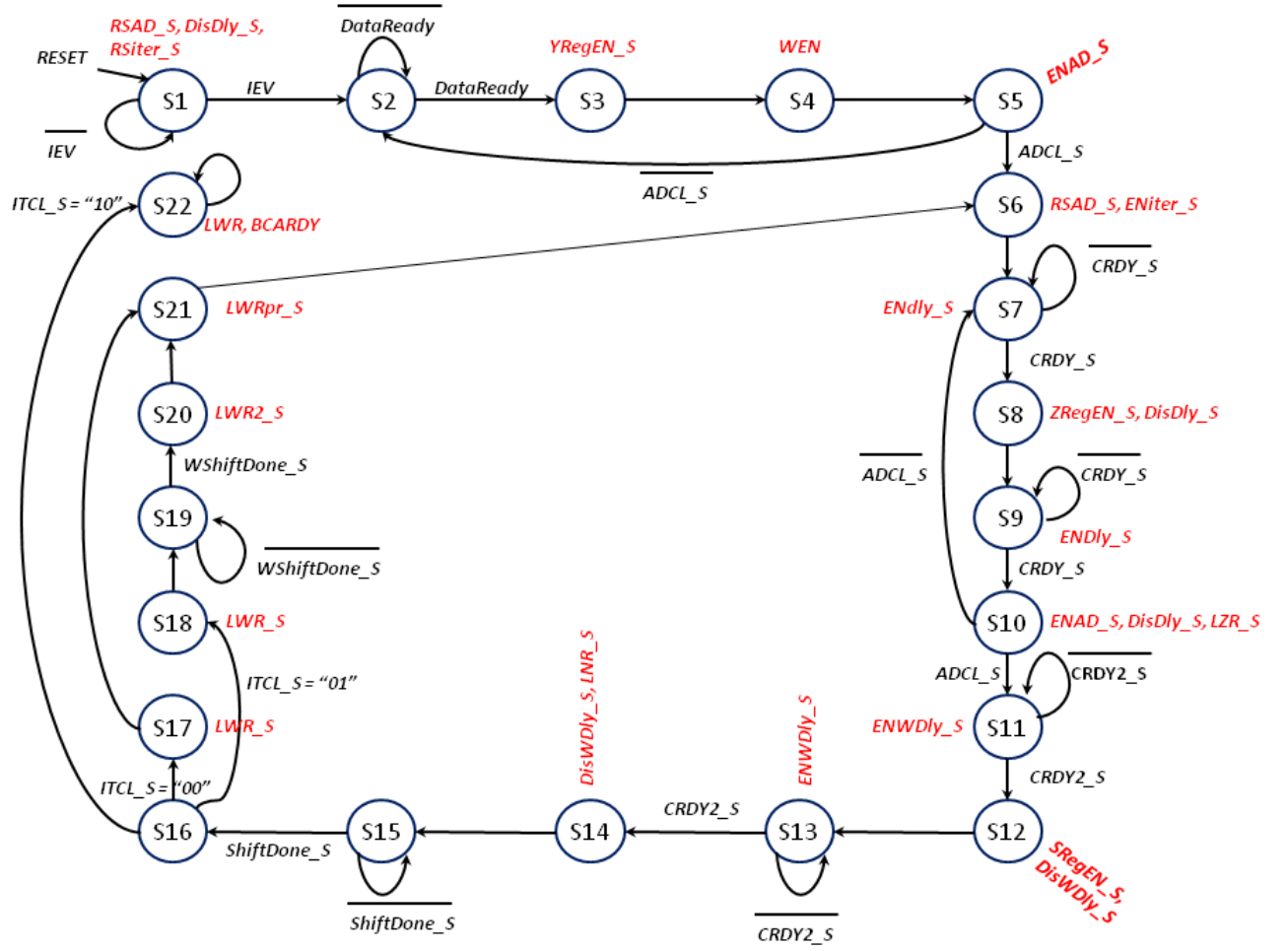


Figure 5.22: Illustration of state diagram for VHDL controller design in the third phase of the interference separation system.

5.7 Communication through Input Buffer Block-BCA Unit-ML Unit

As soon as interference event is detected, BCA and ML algorithm must be activated respectively. However, training of BCA and ML algorithms take some time and new $\mathbf{y}_k$

samples may be arrived during this time period. We need to keep these samples. For that purpose, we designed input buffer block which includes a circular buffer and an additional circuits.

- **Circular Buffer**



Figure 5.23: Memory address allocation for input buffer.

- **M1:** According to SNR and SIR values, event may happen before event detection algorithm decide an event. M1 is number of bits to allocate 2^{m1} addresses for data which are recovered because of the imprecise interference event detection. It depends on the depth of the circular buffer in interference event detection phase.
- **M2:** Number of bits to allocate 2^{m2} addresses to store BCA algorithm training data.
- **M3:** Number of addresses allocated to store received data during the training period of BCA algorithm.
- **M4:** Number of addresses allocated to store received data until ML algorithm successfully take and process the data in n1, n2, n3 and n4 locations.
- **M5:** If there is no addresses in this block, input buffer reallocate the n1 location for future received data. In this situation, there is no overlap in data reallocation

because ML algorithm finished the process of past data somewhere in n4 location.

Thus, the depth of input buffer should contain n1, n2, n3 and n4 locations at least.

Address location n1 in Figure 5.23 are composed of the data receiving from interference event detection circular buffer. By providing communication between the control units of these two phases, these data should be placed to address location n1 before any real-time sample received. The communication mechanism between input buffer block and interference event detection block is detailed on state diagram.

Address pointer shown in Figure 5.24 determines the write address pointer of the input buffer. If write address pointer reaches the buffer depth, it returns to the zeroth location of the input buffer. Address comparator unit compare the result of address counter with predefined integer. This integer is decided by the depth of the address location n1. A control unit signal takes values according to comparison result and this result decide either multiplexers outputs for address location n1 or n2.

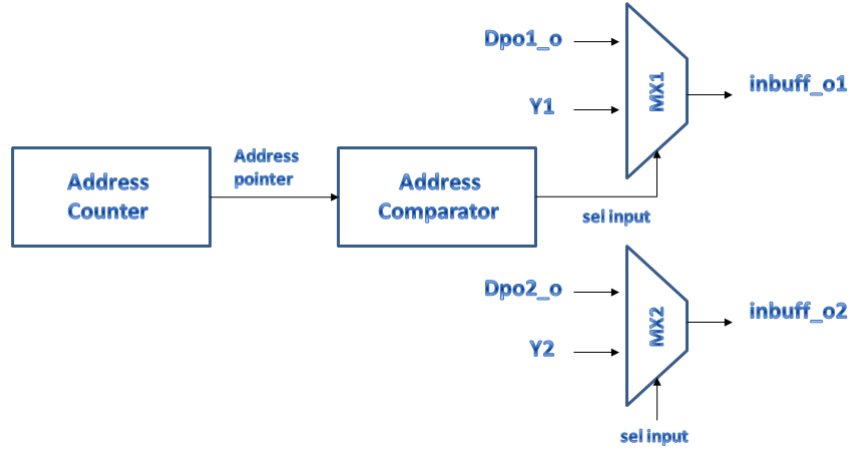


Figure 5.24: Datapath design highlights of input buffer block.

- **Controller for Input Buffer Block**

- **S1:** Just after asynchronous *reset*, controller is in state S1. Circuit proceeds to S2.
- **S2:** Controller wait for *IEV* signal. Before an interference event, input buffer block controller is at stand-by mode. When *IEV* becomes ‘1’, controller proceeds to S3.
- **S3:** Parallel to the state transition from S2 to S3, read pointer at interference event detection phase circular buffer locates the first data to be recovered. Then, *WEN* signal is asserted to buffer this data at the next clock cycle.
- **S4:** *ENAD_S* allocates the next location in input buffer memory block for memory write operation. *ENBAD_S* is the control signal send to interference event detection phase to increment the location of read address pointer there. Therefore, recover from interference event detection phase continues until all the samples are read. The status signal *SELS* decides this mechanism. If read is done, controller proceeds to S5.
- **S5:** We wait for next receive sample. *DataReady* indicates the arrival.
- **S6:** Write enable is asserted.
- **S7:** Next address location is allocated to write and controller proceeds to S5.

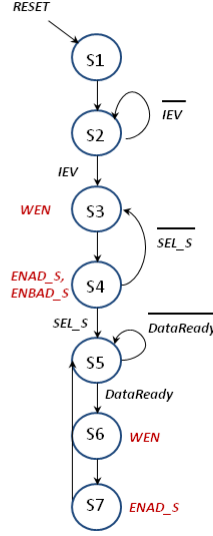


Figure 5.25: Illustration of state diagram for VHDL controller design in input buffer block.

5.8 Conclusion

In this section, we presented the instantaneous BCA approach which based on two geometric objects. We provide the iterative adaptive algorithm and the corresponding algorithm rules to solve optimization problem. We provide the following test scenarios to illustrate the separation capability of the adaptive algorithm. Desired signal power to noise power is fixed to 15 dB. We tested the performance of the algorithm for various SIR scenarios. Each scenario is simulated with $L = 1000$ and $N = 50$. We illustrated the separation performance as total Signal energy to total Interference energy Ratio (SIR) obtained for the BCA algorithm examples for these test scenarios. We proved that convergence of the algorithm occurs before 50th iteration. Moreover, The algorithm performance is compared to ACMA, FastICA and JADE algorithms, as representative ICA approaches for both SNR = 15 dB and SNR = 30 dB.

CHAPTER 6

Interference Offset and Symbol Detection

6.1 Mathematical Setup

In the previous solution block separator matrix was trained based on a finite set of mixture samples. In this block separator matrix will be used to generate two channel matrix candidates to perform Maximum Likelihood detection for joint phase offset and symbol detection. Joint phase offset will be discussed based on the realistic interference scenarios.

To understand the problem Figure 2.1 is considered. The circles show the location of constellation symbols on the diagram where there is a $\frac{\pi}{2}$ phase difference between each consecutive symbol. The symbols are sampled at Ts rate. If we assume that both source and interference signals are sampled simultaneously at indicated correct phases, there is no relative phase offset. Figure 2.2 posed the complex to real transformation of these constellation points. As mentioned in Chapter 2, after the transformation four points on Figure 2.1 are transformed to real axis which we called $\alpha_i(k)$ where i is used for users. This ideal case could not be provided in realistic interference scenarios because interference may include at the receiver with relative phase offset. To identify the phase offset, we exploit the $2M \times Ts$ sampling rate. Thus, $2M - 1$ additional points are constituted between each consecutive constellation symbols. There are $\frac{\pi}{4M}$ phase difference between these additional points. This time, complex to real transformation provides complex

$\alpha_2(k)$ values because additional to the points on Figure 2.1, interference signal may be on one of these points during the sampling. We define a parameter called g which is a phase offset index in which it has $\frac{\pi}{4M}$ phase difference between its two values. Thus, $g = 2M$ means there is $\frac{\pi}{2}$ phase difference between the source and the interference. With this notation, $\alpha_1(k)$ are true offset values on real axis and $\alpha_2(k)$ are unknown offset values on complex domain which have phase information $e^{\frac{j2\pi g}{4M}}$. Using these facts, set of α values and offset g is stated in (6.1), (6.2), (6.3).

$$\alpha_1(k) \in \{-1, 1\} \quad (6.1)$$

$$\alpha_2(k) \in \{e^{\frac{j2\pi g}{4M}}, e^{-\frac{j2\pi g}{4M}}\} \quad (6.2)$$

$$g \in \{0, \dots, M\} \quad (6.3)$$

Maximum Likelihood algorithm is proposed to estimate joint phase offset and symbol detection. h_1 estimate is used to figure out the permutation and scaling ambiguity in the BSS problem. To detail this, we note that for $J_{2,1}$ objective function, ideal $\mathbf{W}$ obtained by the minimization of this objective function has the property that

$$\mathbf{W}\mathbf{H} = d\mathbf{P} \quad (6.4)$$

where $\mathbf{P}$ is the permutation matrix corresponding to the permutation ambiguity, whereas d is the scaling ambiguity. Since the first column of $\mathbf{H}$ is known both d and $\mathbf{P}$ can be estimated. After that we can adjust, $\mathbf{W}$ to eliminate this scaling and permutation ambiguities:

$$\mathbf{W}_{\text{est}} = \frac{1}{d}\mathbf{P}\mathbf{W} \quad (6.5)$$

so that

$$\mathbf{W}_{\text{est}}\mathbf{H} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (6.6)$$

Therefore, an estimate for $\mathbf{H}$ can be obtained by

$$\mathbf{H}_{\text{est}} = \text{pinv}(\mathbf{W}_{\text{est}}) \quad (6.7)$$

In the case of phase offset in the interference source α_2 would be complex as illustrated in Figure 6.1. In such a case, real vector $\mathbf{y}$

$$\begin{aligned} \begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} &= \begin{bmatrix} \Re\{h_1\} & \Re\{h_2\} & -\Im\{h_2\} \\ \Im\{h_1\} & \Im\{h_2\} & \Re\{h_2\} \end{bmatrix} \begin{bmatrix} \alpha_1(k) \\ \Re\{\alpha_2(k)\} \\ \Im\{\alpha_2(k)\} \end{bmatrix} \\ &= \mathcal{H} \begin{bmatrix} \alpha_1(k) \\ \Re\{\alpha_2(k)\} \\ \Im\{\alpha_2(k)\} \end{bmatrix} \end{aligned} \quad (6.8)$$

Based on the estimate $\mathbf{H}_{\text{est}}$, we can estimate $\mathcal{H}$ in (6.8) as

$$\mathcal{H}_{\text{est}} = \begin{bmatrix} \mathbf{H}_{\text{est}}(1,1) & \mathbf{H}_{\text{est}}(1,2) & -\mathbf{H}_{\text{est}}(2,2) \\ \mathbf{H}_{\text{est}}(2,1) & \mathbf{H}_{\text{est}}(2,2) & \mathbf{H}_{\text{est}}(1,2) \end{bmatrix}. \quad (6.9)$$

Then, we can apply Maximum Likelihood(ML) detection for each k observation by the optimization

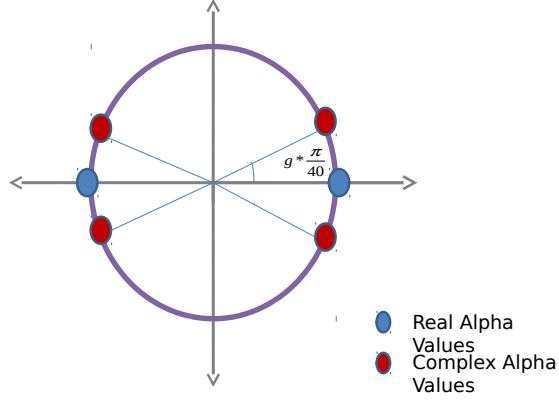


Figure 6.1: Alpha values are shown on the constellation diagram.

$$\text{minimize} \left\| \mathbf{y}(k) - \mathcal{H}_{est} \begin{bmatrix} \alpha_1(k) \\ \Re\{\alpha_2(k)\} \\ \Im\{\alpha_2(k)\} \end{bmatrix} \right\|_2^2 \quad (6.10)$$

for

$$\alpha_1(k) \in \{-1, 1\}, \alpha_2(k) \in e^{\frac{-j\pi g_k}{40}}, g_k \in \{0, \dots, M\} \quad (6.11)$$

As a result of minimization g_k values are found for each k . It is important to notice some details. Firstly, set of g values is defined from 0 to M . The reason is explained with an example. For instance, if g is 3, after the complex to real transformation, $\alpha_1(k)$ values must be either -1 or 1. $\alpha_2(k)$ can be one of the four phases. Blue circles indicates the possible $\alpha_1(k)$ values and red circles indicates the possible $\alpha_2(k)$ values in Figure 6.1. Location difference come from the initial modulation scheme which can be seen from Figure 2.1. However, all of them corresponds to g is 3. In this way, in algorithm,

set of the g is defined from 0 to M but $\alpha_2(k)$ values in minimization equation is searched in all location as can be seen from Figure 6.1. Secondly, as discussed at offset problem subsection if phase offset between the source and the interference is around $\frac{\pi}{2}$, separator matrix $\mathbf{G}$ is not symmetric due to this phase irregularity. This phase irregularity problem was solved in the simulation section by multiplying the second column of the channel with $-j$. In this way, both sources acts as they have correct phases. However **Hest** is found in (25),(26),(27) with nonsymmetric $\mathbf{G}$ so that phase irregularity could not be solved. Briefly, it is not possible to identify whether there is a phase irregularity in the channel or offset difference provides an irregularity. However, our algorithm search minimum error at four different phases and their $\frac{\pi}{2}$ shifted(due to phase irregularity) phases. Then g_{est} calculated as mean of offset estimates based on different samples

$$g_{est} = \frac{1}{L} \sum_{k=1}^L g_k, \quad (6.12)$$

or the median

$$g_{est} = median(g_1, \dots, g_L). \quad (6.13)$$

The results of both approaches are shown in Figure 6.2 and Figure 6.3 respectively.

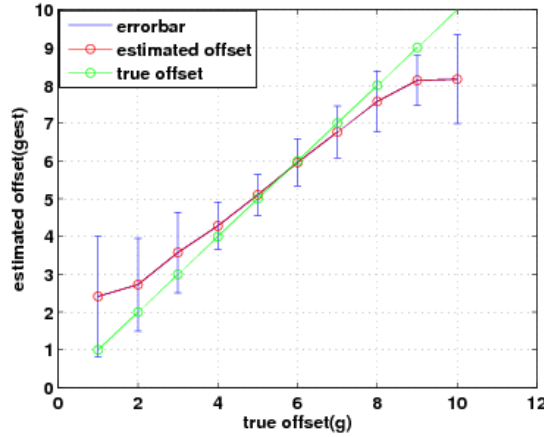


Figure 6.2: Mean based estimate of offset as a function of true offset.

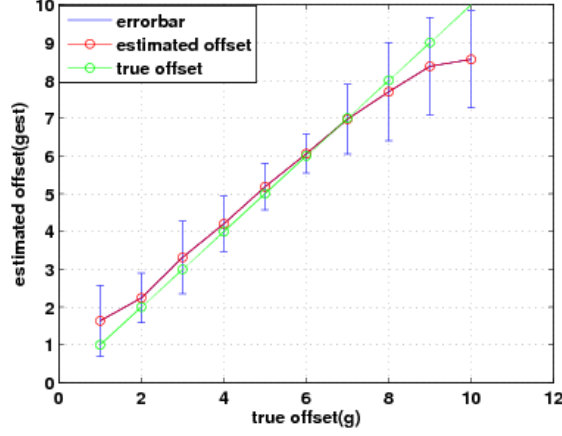


Figure 6.3: Median based estimate of offset as a function of true offset.

Since there is a phase ambiguity as an integer factor of $\frac{\pi}{2}$ for the second user's channel, we repeat the above procedure for the channel obtained by applying $\frac{\pi}{2}$ phase change to the second user's channel. Therefore, the algorithm offers two different decision sequences (and two different phase offsets) corresponding to these two channel alternatives. Furthermore, the complement of the provided sequences are also potentially candidates due to the sign ambiguity. These ambiguities can be resolved by the upper layer by using a side information such as checking the sync word.

$$\mathcal{H}_{est}^{(2)} = \begin{bmatrix} \mathbf{H}_{est}(1,1) & \mathbf{H}_{est}(2,2) & -\mathbf{H}_{est}(1,2) \\ \mathbf{H}_{est}(2,1) & \mathbf{H}_{est}(1,2) & \mathbf{H}_{est}(2,2) \end{bmatrix}. \quad (6.14)$$

6.2 Fixed Point Implementation Issues

We implemented the maximum likelihood (ML) algorithm with integers in MATLAB. We used the similar scaling factor that we used in the integer implementation of the BCA algorithm. Due to the complexity of the algorithm and the redundancy of repetition of same techniques used in previous phases, integer representation of the variables in ML algorithm is not presented. However, one critical issue about the algorithm is posed for convenience. In addition to the regular scaling factor, additional scaling parameter is used to represent $\Re\{\alpha_2(k)\}$ and $\Im\{\alpha_2(k)\}$ for better accuracy.

6.3 VHDL Implementation Issues

In this phase the goal is to design RTL to implement the algorithm stated in (6.10).

6.3.1 Design Specifications

Until BCA outputs are ready, Maximum Likelihood Algorithm is not initiated. After that ML algorithm is ready to start. The received data which were stored in Input Buffer until that moment must be read in sequence and the output of the algorithm must be stored in an output buffer. Block diagram in Figure 6.4 shows the external inputs, external outputs of Interference Offset and Symbol Detection Phase.

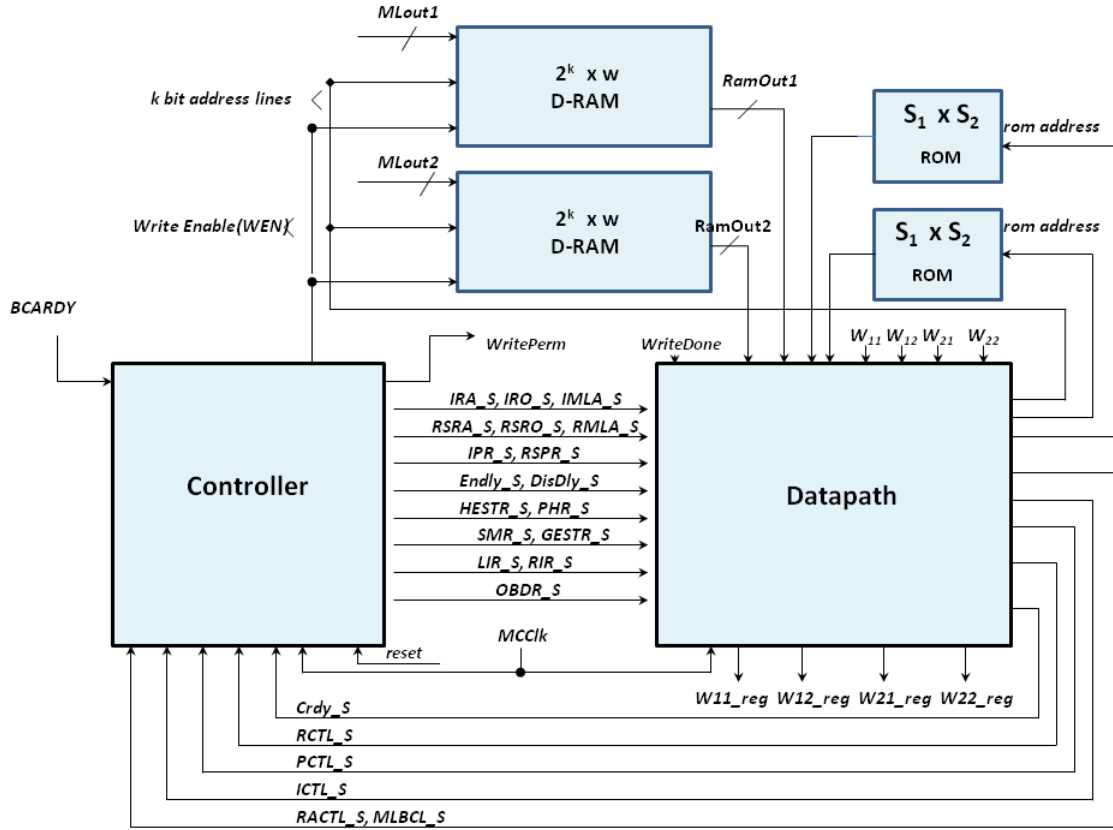


Figure 6.4: Illustration of block diagram for VHDL design of the fourth phase of the interference separation system.

6.3.2 Table of Signals

The external control input/output signals, external data output signals and registers are indicated in Figure 6.5. Functionality of the registers are discussed in detail on state diagram.

Symbol	Function	Type
<i>RESET</i>	Initialize registers to 0	Control input
<i>BCAREADY</i>	Indication that training of the separation algorithm is finished	Control input
<i>WritePerm</i>	Data write permission for output buffer block	Control output
<i>WEN</i>	Enable the <i>write</i> input of the memory unit	Control output
<i>BCAREADY</i>	Indication that training of the separation algorithm is finished	Control output
<i>ADR</i>	BCD counter	7-Bit register
<i>DR</i>	BCD counter	7-Bit register
<i>ROR, MLR</i>	BCD counter	4- Bit register, 6- Bit register
<i>ITR</i>	Parallel load register	2-Bit register
<i>HESR, APR</i>	Parallel load register	16- Bit register
<i>SRR, OBR</i>	Parallel load register	4-Bit register, 16-Bit register

Figure 6.5: External control signals, data output signals and registers in VHDL design of the fourth phase of the interference separation system.

Figure 6.6 and 6.7 shows these signals with their meanings and which registers they alter.

Action or Status	Control or Status Signals	Meaning for Values
$ADR \leftarrow 0$ $ADR \leftarrow ADR + 1$	$RSRA_S$ IRA_S	1: Reset ADR to 0, 0: No reset of ADR 1: BCD count up ADR by 1, 0: Hold ADR value
$ROR \leftarrow 0$ $ROR \leftarrow ROR + 1$	$RSRO_S$ IRO_S	1: Reset ROR to 0, 0: No reset of ROR 1: BCD count up ROR by 1, 0: Hold ROR value
$MLR \leftarrow 0$ $MLR \leftarrow MLR + 1$	$RMLA_S$ $IMLA_S$	1: Reset ROR to 0, 0: No reset of ROR 1: BCD count up ROR by 1, 0: Hold ROR value
$ITR \leftarrow 0$ $ITR \leftarrow ITR + 1$	RIR_S LIR_S	1: Reset ITR to 0, 0: No reset of ITR 1: BCD count up ITR by 1, 0: Hold ITR value
$APR \leftarrow 0$ $APR \leftarrow APR + 1$	$RSPR_S$ IPR_S	1: Reset APR to 0, 0: No reset of APR 1: BCD count up APR by 1, 0: Hold APR value
$DR \leftarrow 0$ $DR \leftarrow DR + 1$	$DisDly_S$ $ENDly_S$	1: Reset DR to 0, 0: No reset of DR 1: BCD count up DR by 1, 0: Hold DR value
$HESR \leftarrow HESR_{next}$ $PR \leftarrow PR_{next}$	$HESR_S$ PHR_S	1: Enable load $HESR$ register, 0: Disable load $HESR$ register 1: Enable load PR register, 0: Disable load PR register
$SRR \leftarrow SRR_{next}$ $OBR \leftarrow OBR_{next}$	SMR_S $OBDR_S$	1: Enable load SRR register, 0: Disable load SRR register 1: Enable load OBR register, 0: Disable load OBR register

Figure 6.6: Control and status signals in VHDL design of the fourth phase of the interference separation system.

Action or Status	Control or Status Signals	Meaning for Values
$DR = N, DR < N$	$CRDY_S$	1: DR is equal to predefined N value, 0: DR is less than predefined N value
$APR < s.size, APR = s.size$	$PCTL_S$	0: APR is less than given sample size, 1: APR is equal to given sample size
$MLR < s.size, MLR = s.size$	$MLBCL_S$	1: MLR is less than given sample size, 1: MLR is equal to given sample size
$ITR = 1, ITR > 1$	$ICTL_S$	0: ITR is at first iteration, 1: ITR is at other iterations
$ADR < s.size - 1, ADR = s.size - 1, ADR = s.size$ or $ADR = s.size + 1, o.w.$	$RACTL_S$	00: ADR is less than given sample size - 1, 01: ADR is equal to sample size-1, 10: ADR is equal to sample size or sample size+1, 11: Otherwise
$ROR < s.size - 1, ROR \geq s.size - 1$	$RCTL_S$	0: ROR is less than given sample size, 1: ROR is equal to given sample size - 1

Figure 6.7: Control and status signals in VHDL design of the fourth phase of the interference separation system.

6.3.3 Datapath Design

ML algorithm search minimum error at four different phases and their $\frac{\pi}{2}$ shifted (due to phase irregularity) phases for each k and g value. There are also two candidates for $\alpha_k^{(1)}$. Therefore, 16 different error candidates will be obtained for one iteration of algorithm. The complexity of the ML algorithm depends on the design structure. Serial process utilize one complex circuit block to estimate (6.10). It requires significant time considering the repetition of this for each k and g value. On the other hand, parallel process requires more area due to the parallel computation of each of minimum error candidate. By considering time-area trade-off, a system with 4 parallel blocks are designed. In this design, 16 computations are grouped into 4x4 parallel computations. It means that at one iteration, four parallel blocks are required for 4 computations. 4 multiplexers are designed to select from $\alpha_k^{(1)}$ and $\alpha_k^{(2)}$ candidates. Then, error detection blocks compare the errors in semi-parallel fashion to find the minimum at each k and g value.

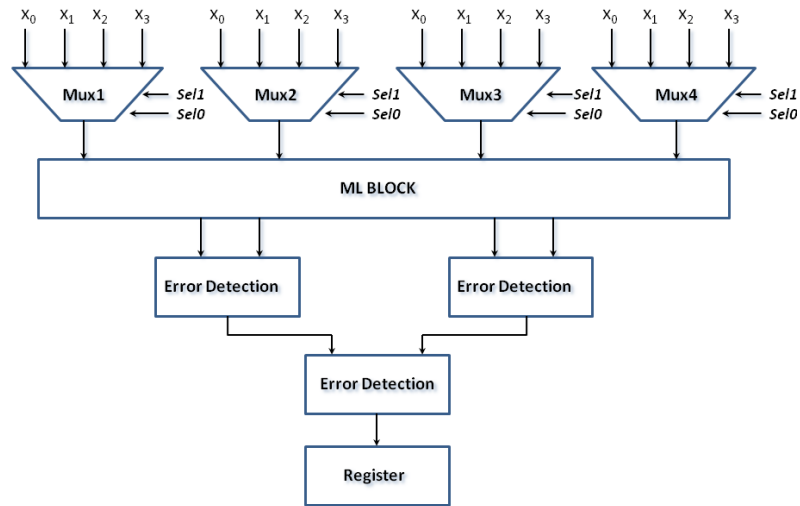


Figure 6.8: Design specifications of the ML algorithm.

6.3.4 Controller Design

The finite state machine diagram is given in Figure 6.9.

- **S1:** Just after asynchronous RESET, controller is in state S1 and content of the registers are cleared. Circuit proceeds to S2 if *BCARDY* signal is enabled.
- **S2:** A register is initialized to a value to indicate reading from a input buffer first time.
- **S3:** Circuit stays in S3 until channel estimates are computed.
- **S4:** Entries of the channel estimate matrix are registered.
- **S5:** At each arrival of this state, ML algorithm is computed for 16 phase values.
- **S6:** ML algorithm outputs are registered. Due to the 4x4 semi-parallel computation, a counter variable is incremented to count iterations of the computation blocks.
- **S7:** Variable to track g_k indexes is incremented. Phase register is cleared. Circuit proceeds to S5 if g_k is not done for processed sample index.
- **S8:** Register which stores g_k indexes are cleared. If initialization register output is '1' and the sample index is smaller then the cardinality of the sample set, then circuit proceeds to S9. It means that training of the ML algorithm continues.
- **S9:** A register stores minimum g_k value.
- **S10:** ML algorithm outputs for k sample index are stored in distributed ram units.
- **S11:** Phase rotation parameter are registered for potential symbol candidates.

- **S12:** Sample index is incremented. Circuit proceeds to S13 if ML estimation is complete with sample set.
- **S13:** Phase corrections are done for potential symbol candidates.
- **S14:** Symbol candidates are registered which will be received by output buffer.
- **S15:** A write permission signal is asserted to alert output buffer controller to write registered symbol candidates. When output buffer controller asserts the *WriteDone_S* signal to the block, circuit proceeds to S16.
- **S16:** Phase correction is repeated for each symbol index. If it is done for all symbols, circuit proceeds to S5 to wait for the computation of ML algorithm for new symbols.
- **S17:** Phase correction.
- **S18:** Symbol candidates are registered which will be received by output buffer
- **S19:** A write permission signal is asserted to alert output buffer controller to write registered symbol candidates. When output buffer controller asserts the *WriteDone_S* signal to the block, circuit proceeds to S20.
- **S20:** Circuit proceeds to S5.

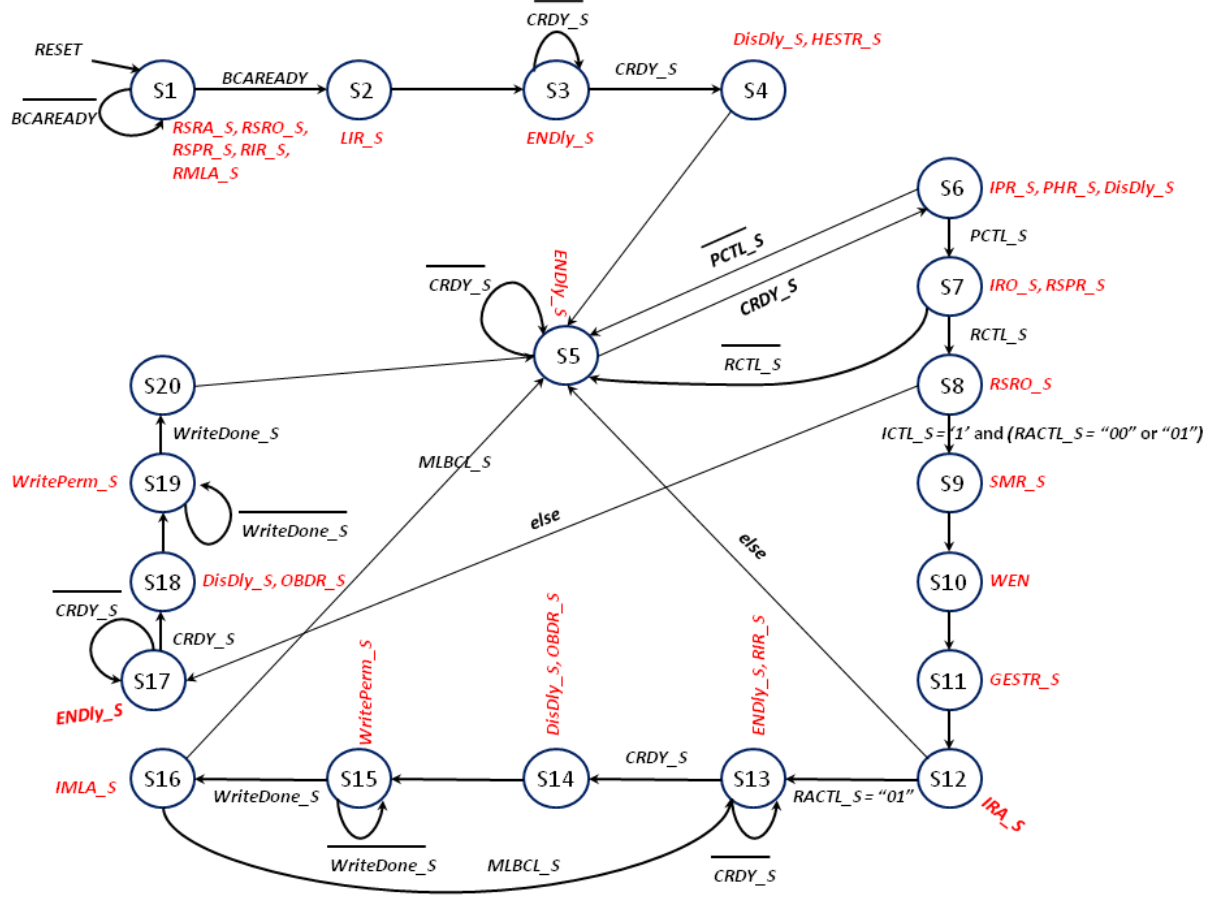


Figure 6.9: Illustration of state diagram for VHDL controller design in fourth phase of the interference separation system.

6.4 Conclusion

In this section, we estimated two different decision sequences and two different phase offsets corresponding to two channel alternatives defined in previous sections. We furthermore, consider the complement of the provided sequences as potential candidates due to the sign ambiguity. We stored these candidates for original symbol constellations in buffer units.

CHAPTER 7

Conclusion

This thesis has presented complete interference separation approach for $\pi/2$ -BPSK communication receivers employing single antenna. Algorithm has been divided into different blocks each proposed distinct solution to the algorithm. Algorithm is based on BCA algorithm which is proposed as an efficient solution to 2×2 blind source separation problem. We successfully implemented the algorithm blocks in both floating point and fixed point MATLAB. We also provided detailed VHDL based hardware implementation guidelines for each block.

Bibliography

- [1] Kraut sieve: a home-made indoor antenna intended to reduce the effect of German jamming stations. Dutch Resistance Museum. <http://www.verzetsmuseum.org/museum/en/alwayspresent.topstukken/moffenzeef>. 1
- [2] H.L. Van Trees, *Optimum Array Processing: (Detection, Estimation, Modulation Theory, Part IV)*, pp. 1, Wiley-Interscience, 2002. 1
- [3] Don H. Johnson, Dan E. Dudgeon, *Array Signal Processing: Concepts and Techniques*, pp. 1, Prentice Hall, 1993. 1
- [4] B. Widrow, et al, “Adaptive Antenna Systems”, vol. 47, pp. 1, *IEEE, Proceedings of IEEE*, vol. 55, pp. 2143-2159, 1967. 1
- [5] A.F. Molisch, *Wireless Communications*, pp. 1, Wiley, 2011. 2
- [6] J. Winters, “Smart Antennas for Wireless Systems”, *IEEE Personal Communications*, vol. 5, pp. 23-27, February 1998. 3
- [7] M.J. Ho, G.L. Stuber, M.D. Austin, “Performance of switched-beam smart antennas for cellular radio systems, *IEEE Trans. on Vehicular Technology*, vol. 47, pp. 10-19, February 1998. 3
- [8] Tse, David and P. Viswanath, *Fundamentals of Wireless Communication*, pp. 1, Cambridge University Press, 2005. 3, 6

- [9] T. Kaiser, et al, *Smart Antennas: State of the Art*, pp. 1, Hindawi Publishing Corporation, 2005. 3
- [10] A. Paulraj, R. Nabar and G. Dhananjay, *Introduction to Space-Time Wireless Communications*, pp. 1, Cambridge University Press, 2008.
- [11] TR, 3GPP TSG GERAN. “Draft Feasibility Study on Single Antenna Interference Cancellation (SAIC) for GSM Networks (Release 10), pp. 1, ETSI, 2011. 3
- [12] H. Trigui, D. Slock, “Co-channel Interference Cancellation within the Current GSM Standard”, *IEEE International Conference on Universal Personal Communications*, vol. 1, pp. 511-515, 1998. 3
- [13] H. Trigui, D. Slock, “Performance Bounds for Co-channel Interference Cancellation within the Current GSM Standard”, Elsevier, pp. 1, *Signal Processing*, vol. 80, pp. 1335-1346, July 2000. 4, 6
- [14] R. Meyer, et al, “A Single Antenna Interference Cancellation Algorithm for Increased GSM Capacity”, *IEEE Trans. on Wireless Commumications*, vol. 5, pp. 1-6, April 2006. 4
- [15] E.S. Warner, I.S. Proudler, “Single Channel Blind Signal Separation of Filtered MPSK Signals”, *IEEE Proc.-Radar Sonar Navig*, vol. 150, pp. 396-402, December 2003. 4, 6
- [16] M. Uesugi, S. Futagi and K. Homma, “Interference Cancellation Method Using DFE”, *IEEE Vehicular Technology Conference*, vol. 2, pp. 1190-1194, 1996. 4
- [17] W. Van Etten, “Maximum Likelihood Receiver for Multiple Channel Transmission Systems”, *IEEE Transactions on Communications*, vol. 24, pp. 276-283, Februry 1976. 5

- [18] K. Giridhar, et al, "Joint Estimation Algorithms for Co-channel Signal Demodulation", *IEEE International Conference on Communications*, vol. 3, pp. 1497-1501, 1993. 5
- [19] K. Giridhar, et al, "Nonlinear Techniques for the Joint Estimation of Co-channel Signals", *IEEE Transactions on Communications*, vol. 45, pp. 473-484, April 1997. 5
- [20] P.A. Ranta, A. Lappetelainen, Z.C. Honkasalo, "Interference Cancellation by Joint Detection in Random Frequency Hopping TDMA Networks", *IEEE International Conference on Communications*, vol. 1, pp. 428-432, 1996. 5, 6
- [21] P. Ranta, A. Hottinen and Z.C. Honkasalo, "Co-channel Interference cancelling receiver for TDMA Mobile Systems, *IEEE International Conference on Communications*, vol. 1, pp. 17-21, 1995. 5
- [22] J.T. Chen, et al, "Low Complexity Joint MLSE Receiver in the Presence of CCI", *IEEE International Conference on Communications*, vol. 2, pp. 704-708, 1998. 5, 6
- [23] A. Hafeez, D. Hui and H. Arslan, "Interference Cancellation for Edge via Two-user Joint Demodulation", *IEEE Vehicular Technology Conference*, vol. 2, pp. 1025-1029, 2003. 6
- [24] A. Duel-Hallen, C. Heegard, "Delayed Decision-feedback Sequence Estimation", *IEEE Transactions on Communications*, vol. 37, pp. 428-436, May 1989. 6
- [25] P.J.W. Melsa, R.C. Younce and C.E. Rohrs, "Impulse Response Shortening for Discrete Multitone Receivers", *IEEE Transactions on Communications*, vol. 44, pp. 1662-1672, December 1996. 6

- [26] P. Hoeher, et al, "Single Antenna Interference Cancellation (SAIC) for Cellular TDMA Networks by Means of Joint Delayed Decision Feedback Sequence Estimation", *IEEE Transactions on Wireless Communications*, vol. 5, pp. 1234-1237, June 2006. 6
- [27] H. Zamiri-Jafarian, S. Pasupathy, "Complexity Reduction of the MLSD/MLSDE Receiver Using the Adaptive State Allocation Algorithm", *IEEE Transactions on Wireless Communications*, vol. 1, pp 101-111, January 2002. 6
- [28] H. Schoeneich, P.A. Hoeher, "Single Antenna Interference Cancellation: Iterative Semiblink Algorithm and Performance Bound for Joint Maximum-Likelihood Interference Cancellation", *IEEE Global Telecommunications Conference*, pp. 1716-1720, 2003. 6
- [29] E. deCarvalho, D. Slock, "Blind and Semi Blind FIR Multichannel Estimation: (Global) Identifiability Conditions", *IEEE Transactions on Signal Processing*, vol. 52, pp. 1053-1064, April 2004. 6
- [30] R.H. Etkin, D.N.C. Tse, H. Wang, "Gaussian Interference Channel Capacity to Within One Bit", *IEEE Transactions on Information Theory*, vol. 54, pp. 5534-5562, December 2008. 6
- [31] H. Arslan, K. Molnar, "Co-channel interference Suppression with Successive Cancellation in Narrow Band Systems", *IEEE Communications Letters*, vol. 5, pp. 37-39, February 2001. 6
- [32] C. Zeng, et al, "Crosstalk Identification in xDSL Systems", *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 1488-1496, August 2001. 7
- [33] A.J. Van der veen, et al, "An Analytical Constant Modulus Algorithm", *IEEE Transactions on Signal Processing*, vol. 44, pp. 1136-1155, May 1996. 7

- [34] P. Comon, “Independent component analysis, A new concept?”, *Signal Processing*, vol. 36, pp. 287-314, April 1994. 7, 8
- [35] P. Comon and C. Jutten, *Handbook of Blind Source Separation: Independent Component Analysis and Applications*, Academic Press, 2010. 7
- [36] D.-T. Pham, “Blind separation of instantaneous mixtures of sources based on order statistics”, *IEEE Trans. on Signal Processing*, vol. 48, pp. 363-375, February 2000. 7, 8
- [37] A. T. Erdogan, “A simple geometric blind source separation method for bounded magnitude sources”, *IEEE Trans. on Signal Processing*, vol. 54, pp. 438-449, February 2006. 8
- [38] A. T. Erdogan, “Globally convergent deflationary instantaneous blind source separation algorithm for digital communication signals”, *IEEE Trans. on Signal Processing*, vol. 55, pp. 2182-2192, May 2007. 8
- [39] F. Vrins, J. A. Lee, and M. Verleysen, “A minimum-range approach to blind extraction of bounded sources”, *IEEE Trans. on Neural Networks*, vol. 18, pp. 809-822, May 2007. 8
- [40] S. Cruces, “Bounded component analysis of linear mixtures: A criterion for minimum convex perimeter”, *IEEE Trans. on Signal Process.*, vol. 58, pp. 2141-2154, April 2010. 8
- [41] A. T. Erdogan, “On the convergence of symmetrically orthogonalized bounded component analysis algorithms for uncorrelated source separation”, *IEEE Transactions on Signal Processing*, pp. 6058-6063, November 2012. 8

- [42] A. T. Erdogan, “A family of bounded component analysis algorithms”, *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1881–1884, March 2012. 8
- [43] A. Erdogan, “A class of bounded component analysis algorithms for the separation of both independent and dependent sources”, *Signal Processing, IEEE Transactions on*, vol. 61, pp. 5730–5743, Nov 2013. 8
- ix, 8, 55